QUANTUM INFORMATION & QUANTUM COMPUTATION

- A Quick Guide -

Huan Q. Bui

Colby College

PHYSICS & MATHEMATICS
Statistics

Class of 2021

May 17, 2020

Preface

Greetings,

This guide is based on *Quantum Computer Science, An Introduction* by N. David Mermin, and *Quantum Computation and Quantum Information* by Isaac Chuang and Michael Nielsen.

Enjoy!

Contents

Part 1

An Introduction

1.1 Introduction

1.1.1 Quantum computer

Quantum mechanics dictates most (if not all) physical interactions. Classical computers work *because* of quantum mechanics. What makes quantum computers "quantum" is the fact that the program *completely* controls *all* interactions in the physical system that makes up the computer. External, unaccounted for, interactions are destructive, resulting in what we call *decoherence*.

Decoherence occurs when the system could not be completely isolated from its own irrelevant interactions. Thus, it is out of the necessity to eliminate decoherence that individual bits must be microscopic systems such as quantum states of atoms.

1.1.2 Cbits

Cbits, Cbit states, Brakets & Vectors

A *bit* in a classical computer contains either a 0 or a 1. A classical computer stores this information in a physical system, such as a switch, which can be either ON or OFF. Such a two-state physical system is called a *Cbit*. The quantum generalization of a Cbit is called a *Qbit* (or *qubit*).

The Cbit can either be in the *state* $|0\rangle$ or $|1\rangle$. The state of five Cbits 11001, say, is given by $|1\rangle\,|1\rangle\,|0\rangle\,|0\rangle\,|1\rangle$, where putting the kets together like this implies the use of the *tensor product*. We won't worry too much about that for now.

Two Cbits can be in any of the following 2^2 states:

$$|1\rangle\,|1\rangle\,, \quad |1\rangle\,|0\rangle\,, \quad |0\rangle\,|1\rangle\,, \quad |0\rangle\,|0\rangle\,. \tag{1.1}$$

Similarly, three Cbits can be in any of their respective 2^3 states

$$|1\rangle\,|1\rangle\,|1\rangle\,, |1\rangle\,|1\rangle\,|0\rangle\,, \ldots, |0\rangle\,|0\rangle\,|0\rangle\,. \tag{1.2}$$

To simplify notations, we will just write $|A\rangle\,|B\rangle = |AB\rangle$ from now on. We can also shorten our notations to represent integers in various bases, but we won't pay much attention to that here.

So far we have been looking at *basis states* of the Cbits. These basis states are orthonormal vectors in some finite dimensional vector space. For example, in the single Cbit case, the vector space is two-dimensional (since there are exactly two basis states), spanned by

$$|0\rangle \equiv \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \quad |1\rangle \equiv \begin{pmatrix} 0 \\ 1 \end{pmatrix}. \tag{1.3}$$

A larger number of Cbits requires an exponentially larger vector space to describe. For example, the basis states for Cbits are

$$\begin{aligned} |11\rangle &\equiv |1\rangle \otimes |1\rangle \\ |10\rangle &\equiv |1\rangle \otimes |0\rangle \\ |01\rangle &\equiv |0\rangle \otimes |1\rangle \\ |00\rangle &\equiv |0\rangle \otimes |0\rangle \,. \end{aligned}$$

The funny $\otimes$ symbol denotes the tensor product. Actually *evaluating* tensor products is merely book-keeping. For example:

$$\begin{pmatrix} a \\ b \end{pmatrix} \otimes \begin{pmatrix} c \\ d \end{pmatrix} = \begin{pmatrix} a \begin{pmatrix} c \\ d \end{pmatrix} \\ b \begin{pmatrix} c \\ d \end{pmatrix} \end{pmatrix} = \begin{pmatrix} ac \\ ad \\ bc \\ bd \end{pmatrix} . \tag{1.4}$$

This generalizes to tensor products of matrices as well, as we will see. Note that in mathematics the tensor product is done in a slightly different order, but there is no fatal discrepancy to worry about. The truly remarkable aspect of the tensor product is that it works. It is not obvious at all how, say

$$|5\rangle_3 = |101\rangle = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \otimes \begin{pmatrix} 1 \\ 0 \end{pmatrix} \otimes \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} , \tag{1.5}$$

where the subscript 3 in $|\,\rangle_3$ denotes the number of Cbits, has the property that the number 5 in the 3-Cbit system is represented by a vector whose only nonzero entry is in the 5^{th} position. If it's not clear what's going on, I would suggest verifying that

$$|j\rangle_n \equiv \big(0 \ldots 1 \ldots 0\big)^\top \tag{1.6}$$

where the number 1 is at the j^{th} position, and the resulting vector lives in a 2^n-dimensional vector space over the finite field $\mathbb{F}_2 = \{0, 1\}$.

In general, the tensor product allows us to represent the state $|m\rangle_n$ as a 2^n column vector whose entries are zero except at the m^{th} entry where the entry is 1.

We can also turn this rule around and obtain a compact, product state representation of any number x where $0 \leq x < 2^n$. We won't go into the details here.

1.1.3 Reversible operations on Cbits

Most quantum operations are *reversible*, exceptor the *measurement*. Measurements are *irreversible*, but it is the only way to extract useful information about the Qbit. One can think of this as "observing the quantum state destroys the quantum state." The extraction of information from Cbits don't necessary destroy the Cbit states, and so we often don't have to worry about this issue in classical computing.

Classical computing has both reversible and irreversible operations, but we are only interested in the former, as only they can be relevantly transfered to the quantum computation landscape.

The NOT operator

The NOT, denoted as $\mathbf{X}$, operation on a Cbit is a reversible operation:

$$\begin{aligned} \mathbf{X}\,|0\rangle &= |1\rangle \\ \mathbf{X}\,|1\rangle &= |0\rangle\,. \end{aligned} \tag{1.7}$$

This operation is colloquially referred to as *bit-flipping*. The inverse of $\mathbf{X}$ is itself, as one can readily verify. We write this as

$$\mathbf{X} \circ \mathbf{X} = \mathbf{X}^2 = \mathbb{I} \equiv \mathbf{1} \tag{1.8}$$

where $\mathbb{I}$ and $\mathbf{1}$ denote the *identity* operator.

When we express the basis states $|0\rangle$ and $|1\rangle$ as vectors in a two-dimensional vector space spanned by the orthonormal basis $\begin{pmatrix}1 & 0\end{pmatrix}^\top$ and $\begin{pmatrix}0 & 1\end{pmatrix}^\top$ respectively, the NOT operator is given by the matrix

$$\boxed{\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}} \tag{1.9}$$

The astute reader immediately recognizes that $\mathbf{X}$ is the first of the Pauli matrices $\boldsymbol{\sigma}_x$, hence the name $\mathbf{X}$. We can readily check $\mathbf{X}\begin{pmatrix}1 & 0\end{pmatrix}^\top = \begin{pmatrix}0 & 1\end{pmatrix}^\top$ and so on. It might also be re-assuring that things work by checking that $\mathbf{X}^2 = \mathbb{I}_2$, which is the 2×2 identity matrix.

The SWAP operator

Things get more interesting when we go to higher dimensions (as they do). Permutations among the possible states are some of many reversible operations on a number Cbits. For example, the SWAP operator $\mathbf{S}_{ij}$ for a pair of Cbits (which has a total of 4! possible permutations) interchanges the states i and j:

$$\mathbf{S}_{10}\,|xy\rangle = |yx\rangle\,. \tag{1.10}$$

One can readily verify that in the $0-1$ orthonormal basis, the SWAP matrix takes the form:

$$\mathbf{S}_{10} = \mathbf{S}_{01} = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}. \tag{1.11}$$

Staring at this matrix for a bit and we will see that it swaps $|01\rangle \leftrightarrow |10\rangle$ but keep $|00\rangle$ and $|11\rangle$ the same. It is also obvious that $\mathbf{S}_{ij}^2 = \mathbb{I}$.

The cNOT operator

We also consider the *controlled-NOT*, or cNOT gate, denoted as $\mathbf{C}_{ij}$, where i is the *control Cbit* and j is the *target Cbit*. If the control Cbit i is $|0\rangle$, then j is unchanged. Else, j is flipped via a NOT gate (hence cNOT). The action of the cNOT gate can be describe via the following two equations:

$$\begin{aligned} \mathbf{C}_{10} |xy\rangle &= |x\rangle |y \oplus x\rangle \\ \mathbf{C}_{01} |xy\rangle &= |x \oplus y\rangle |y\rangle , \end{aligned} \tag{1.12}$$

where the funny $\oplus$ denotes additional modulo 2. We note that $\mathbf{C}_{ij} \neq \mathbf{C}_{ji}$.

In matrix form,

$$\boxed{\mathbf{C}_{10} = \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix}, \quad \mathbf{C}_{01} = \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix}} \tag{1.13}$$

When in doubt as to which $\mathbf{C}_{ij}$ is which matrix, a few test cases will do the trick. Just remember that in $\mathbf{C}_{ij}$, i is the control bit, and j is the target bit. It also turns out that $\mathbf{S}_{ij} = \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij}$, which can readily be checked with matrix multiplication.

In practice, we should not worry too much about what the operations in matrix form are like (because matrices require bases, while state vectors exist by themselves in their vector spaces). It is more common to know how operators act on states, and how they are constructed from other operators, provided they could be so. 2-Cbit operators can be formed by taking the tensor product of two 1-Cbit operators:

$$\boxed{(\mathbf{a} \otimes \mathbf{b}) |xy\rangle = (\mathbf{a} \otimes \mathbf{b}) |x\rangle \otimes |y\rangle = \mathbf{a} |x\rangle \otimes \mathbf{b} |y\rangle} \tag{1.14}$$

It isn't clear how the tensor product works this way on first glance. It is actually even more magical once you plug in some vectors and matrices and verify that equality holds. Of course, this is because the tensor product is constructed so

that it behaves exactly like this, and that the mathematics behind the tensor product makes sure it is as nice as we want it to be.

From here, we can guess (and then check, of course) that

$$\boxed{(\mathbf{a}\otimes\mathbf{b})(\mathbf{c}\otimes\mathbf{d}) = (\mathbf{ac})\otimes(\mathbf{bd})} \tag{1.15}$$

See? As nice as we want it to be.

Just a word of caution: not all 2-bit operators can be written a tensor product of two 1-bit operators. We will see this more when we go to the quantum case. The key/buzzword here is *entanglement*.

And so we see that the whole purpose of the tensor product of operators is such that we can express, as a single operator, an operation on a multi-bit system that acts on individual bits differently, with or without the bit operations interfering each other. For example,

$$\mathbb{I}\otimes\mathbb{I}\otimes\mathbf{a}\otimes\mathbb{I}\otimes\mathbf{b}\otimes\mathbb{I} \tag{1.16}$$

is a 6-Cbit operator which does nothing to the bits except applying $\mathbf{a}$ to the fourth bit and $\mathbf{b}$ to the second bit *from the right*. In practice, however, we write this operator more elegantly (and in a more self-explanatory manner) as

$$\mathbb{I}\otimes\mathbb{I}\otimes\mathbf{a}\otimes\mathbb{I}\otimes\mathbf{b}\otimes\mathbb{I} = \mathbf{a}_3\mathbf{b}_1 = \mathbf{b}_1\mathbf{a}_3 \tag{1.17}$$

where the Cbits are indexed 0 to 5 from the right.

1.1.4 Manipulating operations on Cbits

The *number* operator

We now introduce the 1-Cbit *number operator* $\mathbf{n}$:

$$\mathbf{n}\,|x\rangle = x\,|x\rangle \tag{1.18}$$

where $|x\rangle$ is the eigenstate $|0\rangle$ or $|1\rangle$ of $\mathbf{n}$. $\mathbf{n}$ projects any state onto the state $|1\rangle$, and so it makes sense that $|1\rangle$ gets mapped to itself and $|0\rangle$, which is orthogonal to $|1\rangle$, is mapped to zero. We also define its complementary operator:

$$\tilde{\mathbf{n}} = \mathbb{I} - \mathbf{n}, \tag{1.19}$$

which projects any state onto $|0\rangle$. In matrix form (under the 0-1 basis, of course),

$$\mathbf{n} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}, \quad \tilde{\mathbf{n}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}. \tag{1.20}$$

These operators are special cases of *idempotent operators*. Roughly speaking, they are projections because their respective eigenspaces are orthogonal to each other. It follows from these properties that

$$\begin{aligned}\mathbf{n}^2 &= \mathbf{n}\\ \tilde{\mathbf{n}}^2 &= \tilde{\mathbf{n}}\\ \mathbf{n}\tilde{\mathbf{n}} &= \tilde{\mathbf{n}}\mathbf{n} = \mathbf{0}\\ \mathbf{n} + \tilde{\mathbf{n}} &= \mathbb{I}.\end{aligned} \tag{1.21}$$

The last property is a special case of what's called *resolution of identity*. Many properties of idempotents and matrices are covered in my matrix analysis notes.

We also have

$$\begin{aligned}\mathbf{n}\mathbf{X} &= \mathbf{X}\tilde{\mathbf{n}}\\ \tilde{\mathbf{n}}\mathbf{X} &= \mathbf{X}\mathbf{n}.\end{aligned} \tag{1.22}$$

This makes intuitive sense but deserves a check nevertheless.

While there are no good physical interpretations for the action of $\mathbf{n}$ and $\tilde{\mathbf{n}}$, they are incredibly useful as building blocks of other fundamental operators. For example, the SWAP operator can be written as

$$\boxed{\mathbf{S}_{ij} = \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + (\mathbf{X}_i\mathbf{X}_j)(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)} \tag{1.23}$$

where all of the "multiplication" in the formula above are tensor products. They are not normal matrix multiplications because $\mathbf{S}_{ij}$ is a 4×4 matrix while the building blocks are actually 2×2 matrices.

Now, we can't believe this equality until we at least see how to get there, which is via the cNOT operators $\mathbf{C}_{ij}$. Recall that

$$\mathbf{S}_{ij} = \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij}. \tag{1.24}$$

The cNOT operators can actually be written in terms of the projections and NOT operators

$$\boxed{\mathbf{C}_{ij} = \tilde{\mathbf{n}}_i + \mathbf{X}_j\mathbf{n}_i} \tag{1.25}$$

This expression requires some understanding of how the tensor product works, so I will rewrite $\mathbf{C}_{ij}$ in a more straightforward manner as

$$\boxed{\mathbf{C}_{ij} = \tilde{\mathbf{n}}_i \otimes \mathbb{I}_j + \mathbf{n}_i \otimes \mathbf{X}_j} \tag{1.26}$$

The two expressions are equivalent, but once emphasizes the fact that the j operators act only on the j bit and i on i (and hence writing $\mathbf{X}$ or $\mathbf{n}$ does not matter provided we include proper subscripts). Writing this way also allows us to directly verify in matrix form that the formula makes sense. This is left as an exercise to the reader.

The Z operator

We will also define a new operator $\mathbf{Z}$ by

$$\boxed{\mathbf{Z} = \tilde{\mathbf{n}} - \mathbf{n} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}} \tag{1.27}$$

The astute reader will immediately recognize that this is the third of the Pauli matrices $\boldsymbol{\sigma}_z$, hence the name $\mathbf{Z}$. One of the well-known properties of the Pauli matrices is the fact that $\mathbf{Z}$ and $\mathbf{X}$ *anticommute*, i.e.

$$\mathbf{ZX} = -\mathbf{XZ} \iff \{\mathbf{Z}, \mathbf{X}\} = 0 \tag{1.28}$$

where the $\{\ ,\ \}$ is called (within our scope) the Poisson bracket, or the *anticommutator*.

Now, we have $\mathbf{n} + \tilde{\mathbf{n}} = \mathbb{I}$ and $\mathbf{Z} = \tilde{\mathbf{n}} - \mathbf{n}$, solving these two equations for $\mathbf{n}$ and $\tilde{\mathbf{n}}$ we find:

$$\boxed{\mathbf{n} = \frac{1}{2}(\mathbb{I} - \mathbf{Z}) \quad \tilde{\mathbf{n}} = \frac{1}{2}(\mathbb{I} + \mathbf{Z})} \tag{1.29}$$

from which we can write

$$\begin{aligned} \mathbf{C}_{ij} &= \tilde{\mathbf{n}}_i \otimes \mathbb{I}_j + \mathbf{n}_i \otimes \mathbf{X}_j \\ &= \frac{1}{2}(\mathbb{I}_i + \mathbf{Z}_i) \otimes \mathbb{I}_j + \frac{1}{2}(\mathbb{I}_i - \mathbf{Z}_i) \otimes \mathbf{X}_j \\ &= \frac{1}{2}\mathbb{I}_i \otimes (\mathbb{I}_j + \mathbf{X}_j) + \frac{1}{2}\mathbf{Z}_i \otimes (\mathbb{I}_j - \mathbf{X}_j), \end{aligned} \tag{1.30}$$

where the third equality follows from the fact that $[\mathbf{Z}_i, \mathbf{X}_j] = 0$. The notation in the textbook is more compact as it does not have to deal with the ordering of operators (we don't *have* to, technically), but I like to keep things a bit more organized.

Alas, the textbook formula gives us the ability to illustrate the next point, so I will include it anyway:

$$\boxed{\mathbf{C}_{ij} = \frac{1}{2}(\mathbb{I} + \mathbf{Z}_i) + \frac{1}{2}(\mathbb{I} - \mathbf{Z}_i) = \frac{1}{2}(\mathbb{I} + \mathbf{X}_j) + \frac{1}{2}\mathbf{Z}_i(\mathbb{I} - \mathbf{X}_j)} \tag{1.31}$$

which tells us that if we were to interchange $\mathbf{X}$ and $\mathbf{Z}$ (and being careful with interchanging i and j as well), we can move from one formula to the other. This means interchanging $\mathbf{X}$ and $\mathbf{Z}$ has the effect of switching with Cbit is the control and which is the target, i.e. $\mathbf{C}_{ij} \leftrightarrow \mathbf{C}_{ji}$.

The Hadamard operator

The key operator among all these interchanging of operators and bits is the *Hadamard transformation* (or *Hadamard operator*, or *Walsh-Hadamard transformation*), given by

$$\boxed{\mathbf{H} = \frac{1}{\sqrt{2}}(\mathbf{X} + \mathbf{Z}) = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}} \tag{1.32}$$

This operator is of fundamental importance in quantum computation. Now, we will study its properties and how see it is responsible for $\mathbf{Z} \leftrightarrow \mathbf{X}$ change, as well as the $\mathbf{C}_{ij} \leftrightarrow \mathbf{C}_{ji}$ change.

It is easy to check that

$$\mathbf{H}^2 = \mathbb{I}, \tag{1.33}$$

which means $\mathbf{H} = \mathbf{H}^{-1}$, i.e. it is involutory, i.e. it is both self-adjoint (Hermitian) and unitary.

We also notice that $\mathbf{H}$ is also Hermitian and unitary, and $\mathbf{Z}$ and $\mathbf{X}$ are similar under the $\mathbf{H}$, i.e.

$$\boxed{\mathbf{HXH} = \mathbf{Z}, \quad \mathbf{HZH} = \mathbf{X}} \tag{1.34}$$

With this, we can comfortably interchange $\mathbf{X}$ and $\mathbf{Z}$ in $\mathbf{C}_{ij}$. From what we had before, we can show that

$$\boxed{\mathbf{C}_{ij} = (\mathbf{H}_i \otimes \mathbf{H}_j)\mathbf{C}_{ij}(\mathbf{H}_i \otimes \mathbf{H}_j)} \tag{1.35}$$

or

$$\boxed{\mathbf{C}_{ij} = (\mathbf{H}_i\mathbf{H}_j)\mathbf{C}_{ji}(\mathbf{H}_i\mathbf{H}_j)} \tag{1.36}$$

for short. Note that the order in which $\mathbf{H}_i$ and $\mathbf{H}_j$ appear does not matter here because their matrix forms are identical. This formula will be crucial later on.

Now, while it is true that the interchanging between control and target bits could be done using only the SWAP operator:

$$\boxed{\mathbf{C}_{ij} = \mathbf{S}_{ij}\mathbf{C}_{ji}\mathbf{S}_{ij}} \tag{1.37}$$

the same action using the Hadamard operators are far superior because it is formed via a tensor product of two 1-Cbit operators, while the SWAP gate cannot be written as a tensor product of two 1-Cbit operators.

On single Cbit states, we have

$$\boxed{\mathbf{H}\,|0\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \quad \mathbf{H}\,|1\rangle = \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)} \tag{1.38}$$

We will discuss the significance of this when we work with the quantum cases.

The $\mathbf{C}^Z$ operator

The next important operator is called the *controlled-Z* operator, denoted as $\mathbf{C}^Z_{ij}$. If the control bit i is $|0\rangle$, then cZ does nothing to j Else, cZ lets $\mathbf{Z}$ act on j. So, we have that

$$\boxed{\mathbf{C}^Z_{10}\,|xy\rangle = |xy\rangle \;\text{ unless }\; |x\rangle = |y\rangle = 1, \quad \mathbf{C}^Z_{10}\,|11\rangle = -\,|11\rangle} \tag{1.39}$$

The action of cZ is symmetric in the two Cbits, so we have

$$\boxed{\mathbf{C}^Z_{ij} = \mathbf{C}^Z_{ji}} \tag{1.40}$$

Now, the cNOT and cZ operators are actually similar under $\mathbf{H}$ as well:

$$\boxed{\mathbf{H}_j\mathbf{C}_{ij}\mathbf{H}_j = \mathbf{C}^Z_{ij} = \mathbf{C}^Z_{ji} = \mathbf{H}_i\mathbf{C}_{ji}\mathbf{H}_i} \tag{1.41}$$

where the $\mathbf{H}_k$'s here have been promoted to the 4×4 versions via tensoring with $\mathbb{I}_2$. But note that this equality is just (1.36).

To complete the introductory picture in Cbits, we introduce one more operator:

$$\boxed{\mathbf{Y} = i\mathbf{X}\mathbf{Z} = \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix}} \tag{1.42}$$

which is the last Pauli matrix $\boldsymbol{\sigma}_y$. With this, we will try to write the SWAP operator in terms of only $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$, and the identity. We start with substituting (1.29) into (1.23) to get

$$\begin{aligned} \mathbf{S}_{ij} &= \frac{1}{2}(\mathbb{I}_4 + \mathbf{Z}_i \otimes \mathbf{Z}_j) + \frac{1}{2}(\mathbf{X}_i \otimes \mathbf{X}_j)(\mathbb{I}_4 - \mathbf{Z}_i \otimes \mathbf{Z}_j) \\ &= \frac{1}{2}\left(\mathbb{I}_4 + \mathbf{X}_i \otimes \mathbf{X}_j + \mathbf{Z}_i \otimes \mathbf{Z}_j - (\mathbf{X}_i\mathbf{Z}_i) \otimes (\mathbf{X}_j\mathbf{Z}_j)\right) \\ &= \frac{1}{2}\left(\mathbb{I}_4 + \mathbf{X}_i \otimes \mathbf{X}_j + \mathbf{Z}_i \otimes \mathbf{Z}_j + i(\mathbf{X}_i\mathbf{Z}_i) \otimes i(\mathbf{X}_j\mathbf{Z}_j)\right). \end{aligned} \tag{1.43}$$

It is now easy to see that

$$\mathbf{S}_{ij} = \frac{1}{2}\left(\mathbb{I}_4 + \mathbf{X}_i \otimes \mathbf{X}_j + \mathbf{Y}_i \otimes \mathbf{Y}_j + \mathbf{Z}_i \otimes \mathbf{Z}_j\right) \tag{1.44}$$

or

$$\mathbf{S}_{ij} = \frac{1}{2}\left(\mathbb{I}_4 + \mathbf{X}_i\mathbf{X}_j + \mathbf{Y}_i\mathbf{Y}_j + \mathbf{Z}_i\mathbf{Z}_j\right) \tag{1.45}$$

or even more compactly

$$\boxed{\mathbf{S}_{ij} = \frac{1}{2}\left(\mathbb{I}_4 + \vec{\boldsymbol{\sigma}}^{(i)} \cdot \vec{\boldsymbol{\sigma}}^{(j)}\right)} \tag{1.46}$$

where we have defined the "dot-tensor-product"

$$\boxed{\vec{\boldsymbol{\sigma}}^{(i)} \cdot \vec{\boldsymbol{\sigma}}^{(j)} = \vec{\boldsymbol{\sigma}}_x^{(i)} \otimes \vec{\boldsymbol{\sigma}}_x^{(j)} + \vec{\boldsymbol{\sigma}}_y^{(i)} \otimes \vec{\boldsymbol{\sigma}}_y^{(j)} + \vec{\boldsymbol{\sigma}}_z^{(i)} \otimes \vec{\boldsymbol{\sigma}}_z^{(j)}} \tag{1.47}$$

A good way to remember when to use the ordinary multiplication or the tensor product is the following when we have two operators multiplying in some arbitrary way: If the operators act on different bits (or vector spaces), use the tensor product. Else, it is the ordinary multiplication. So in the definition above, because the Pauli matrices in each summand act on different bits (i and j respectively) the "product" in the "dot-product" is the tensor product.

Some properties of Pauli matrices

Here are some important properties of Pauli matrices that might come in handy later. First, all Pauli matrices square to identity:

$$\boldsymbol{\sigma}_x^2 = \boldsymbol{\sigma}_y^2 = \boldsymbol{\sigma}_z^2 = \mathbb{I}. \tag{1.48}$$

They anticommute:

$$\{\boldsymbol{\sigma}_x, \boldsymbol{\sigma}_y\} = 0. \tag{1.49}$$

Product of any two is related to the third in a cyclic fashion:

$$\boldsymbol{\sigma}_x \boldsymbol{\sigma}_y = i\boldsymbol{\sigma}_z \tag{1.50}$$

$$\vdots$$

All of these properties can be summed up in a single statement: For $\vec{a}, \vec{b} \in \mathbb{R}^3$,

$$\boxed{(\vec{a} \cdot \vec{\boldsymbol{\sigma}})(\vec{b} \cdot \vec{\boldsymbol{\sigma}}) = (\vec{a} \cdot \vec{b})\mathbb{I} + i(\vec{a} \times \vec{b}) \cdot \vec{\boldsymbol{\sigma}}} \tag{1.51}$$

Together with the identity matrix $\mathbb{I}_2$, the Pauli matrices form a basis for the 4-dimensional algebra of two-dimensional matrices of complex numbers: any such matrix is a unique linear combination of these four. Also, because all are Hermitian, any two-dimensional Hermitian matrix $\mathcal{A}$ of complex numbers must have the form

$$\mathcal{A} = a_0 \mathbb{I}_2 + \vec{a} \cdot \boldsymbol{\sigma} \tag{1.52}$$

where $a_0 \in \mathbb{R}$ and $\vec{a} \in \mathbb{R}^3$.

1.1.5 Qbits & their states

Without further ado, the general state of a Qbit is any arbitrary linear combination of the basis states of the Cbit

$$|\psi\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle \equiv \begin{pmatrix} \alpha_0 \\ \alpha_1 \end{pmatrix} \tag{1.53}$$

where $\alpha_0, \alpha_1 \in \mathbb{C}$ and satisfy the normalization condition

$$|\alpha_0|^2 + |\alpha_1|^2 = 1. \tag{1.54}$$

We say $|\psi\rangle$ is a *superposition* of the states $|0\rangle$ and $|1\rangle$, with amplitudes α_0 and α_1, respectively.

We can extend this definition for a system of multiple Qbits. For example, the quantum state of two Qbits is a normalized superposition of the four orthonormal basis states of the 2-Cbit system:

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle \equiv \begin{pmatrix} \alpha_{00} \\ \alpha_{01} \\ \alpha_{10} \\ \alpha_{11} \end{pmatrix} \tag{1.55}$$

and so on for n Qbits:

$$\boxed{|\Psi\rangle \sum_{0 \le x < 2^n} \alpha_x |x\rangle_n} \tag{1.56}$$

where of course the normalization condition must be satisfied at all times:

$$\sum_{0 \le x < 2^n} |\alpha_x|^2 = 1. \tag{1.57}$$

The set of 2^n classical basis states generated by the tensor products of n individual Qbits states $|0\rangle$ and $|1\rangle$ is called the *computational basis*, or *classical basis*.

Given two arbitrary Qbits

$$\begin{aligned} |\psi\rangle &= \alpha_0|0\rangle + \alpha_1|1\rangle \\ |\Phi\rangle &= \beta_0|0\rangle + \beta_1|1\rangle, \end{aligned} \tag{1.58}$$

the multi-Qbit system is created (once again) via the tensor product:

$$\begin{aligned} |\Psi\rangle &= |\psi\rangle \otimes |\Phi\rangle \\ &= (\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle) \\ &= \alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle \\ &= \begin{pmatrix} \alpha_0\beta_0 \\ \alpha_0\beta_1 \\ \alpha_1\beta_0 \\ \alpha_1\beta_1 \end{pmatrix}, \end{aligned} \tag{1.59}$$

which is nothing out of the blue if you think about how we have been creating multiple-Cbit states from single-Cbit states.

Just as how not all multiple-Cbit operators can be written as a tensor product of singe-Cbit operators, not all multi-Qbit states can be written as a tensor product of single-Qbit states as we will see very soon. Such nonproduct states are called *entangled* states.

1.1.6 Reversible operations on Qbits

The only nontrivial reversible operation a classical computer can perform on a single Cbit is the **X**, or the bit-flip. On the quantum computers, however, the possibilities are infinite. Reversible operations on quantum computers not only are invertible but they also must preserve the normalization condition of the state vector. This means that any such single-Qbit operator is *unitary*, which satisfies the condition:

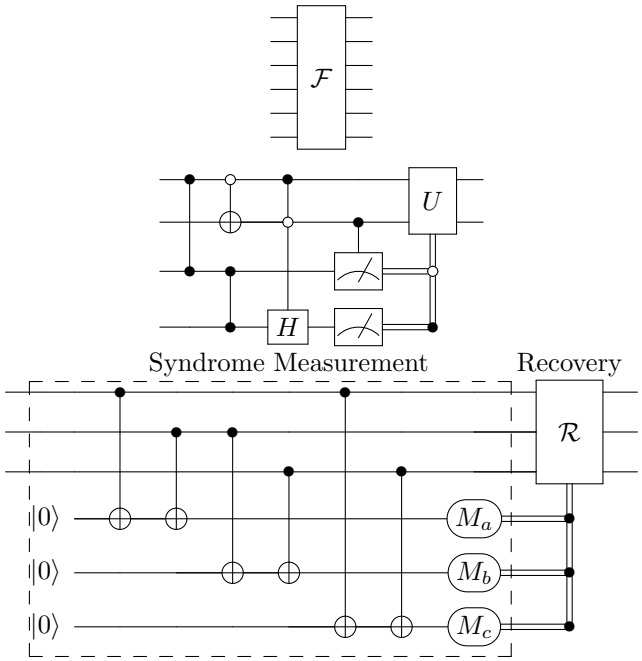
$$\mathbf{u}^\dagger \mathbf{u} = \mathbb{I}. \tag{1.60}$$

In higher dimensions, the same story goes, and we end up with only unitary operators in 2^n-dimensional vector spaces as well:

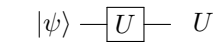
$$\mathbf{U}^\dagger \mathbf{U} = \mathbb{I}_{2^n}. \tag{1.61}$$

1.1.7 Circuit diagrams

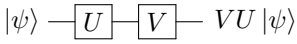
Here are some visual examples of what quantum circuits:



Circuit diagrams follow the flow of time (conventionally left-to-right). For example:



However, when writing the outcome of a quantum circuit, one must be careful to follow the rule of function composition. For example:



1.1.8 Measurement gates and the Born rule

A single Cbit can either be in the state $|0\rangle$ or $|1\rangle$. On the other hand, a single Qbit can be in an infinite number of superpositions of the states $|0\rangle$ and $|1\rangle$. Further, because of this property, Qbits can be transformed under a much richer set of transformations. Thus, it can seem that quantum computers would be vastly more powerful than classical computers. However, the catch lies in observing/measuring the states of the physical bit. For classical bits, simply looking at the physical bit can tell us the state of the Cbit. The Cbit remains intact after the observation. When we have n Qbits in a superposition of computational basis states, there is nothing we can do to observe the amplitudes without projecting them onto one of the basis states and destroying the Qbit state in the process.

Making a measurement consists of performing a certain test on each Qbit, the outcome of which is either 0 or 1, which some probability. If the state of n Qbits is

$$|\psi\rangle_n = \sum_{0 \leq x \leq 2^n} \alpha_x |x\rangle_n \tag{1.62}$$

then the probability that the zeros and ones resulting from measurements of all the Qbits will give the binary expansion of the integer x is:

$$p(x) = |\alpha_x|^2 \tag{1.63}$$

This is known as the *Born rule*. An n-Qbit *measurement gate* is depicted schematically as follows

$$|\psi\rangle_n = \sum \alpha_x |x\rangle \mathbf{M}_n \to x |x\rangle_n$$

In contrast to unitary gates, which are invertible and hence produce a unique output for each input, the state of the Qbits emerging from a measurement gate is only statistically determined by the state of the input Qbits. This action cannot be undone, and hence **measurement is irreversible**. The action of the measurement gate is also NOT linear.

n-Qbit measurement gates can be realized by applying 1-Qbit measurement gates to each of the n Qbits. A measurement reduces the initial quantum state to one of the computational basis states. This change of state is called the *reduction* or the *collapse* of the quantum state.

There is a possibility of misunderstanding the Born rule by asserting that when a single Qbit to be in superposition its "actual state" is either $|0\rangle$ with probability $|a_0|^2$ or $|1\rangle$ with probability $|a_1|^2$, say. Here's why the Qbit could not have been in either states initially. Consider

$$|\psi\rangle = \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle) . \tag{1.64}$$

The Qbit is initially in superposition. Now applying the Hadamard transformation to it

$$\mathbf{H}\,|\psi\rangle = |0\rangle\,. \tag{1.65}$$

According to the Born rule, if we measure a Qbit in the state $|\psi\rangle$, we get $|0\rangle$ with probability 1. Now, if the Qbit were in either $|0\rangle$ or $|1\rangle$ with equal probability initially, then the Hadamard gate would put it in either the superposition $(1/\sqrt{2})(|0\rangle + |1\rangle)$ or the superposition $(1/\sqrt{2})(|0\rangle - |1\rangle)$, which directly contradicts the result $\mathbf{H}\,|\psi\rangle = |0\rangle$.

The Born rule is often stated in terms of inner products or projections operators. The probability of a measurement giving the result $0 \leq x \leq 2^n$ is

$$\boxed{p_\psi(x) = |\alpha_x|^2 = |\langle x|\psi\rangle|^2} \tag{1.66}$$

In terms of projection operators:

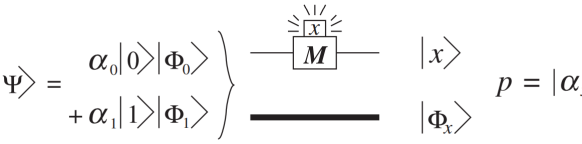
$$\boxed{p_\psi(x) = \langle x|\psi\rangle\,\langle\psi|x\rangle = \langle x|\,\mathbf{P}_\psi\,|x\rangle} \tag{1.67}$$

or

$$\boxed{p_\psi(x) = \langle\psi|x\rangle\,\langle x|\psi\rangle = \langle\psi|\,\mathbf{P}_x\,|\psi\rangle} \tag{1.68}$$

where $\mathbf{P}_\psi = |\psi\rangle\,\langle\psi|$ is the projection operator on the state ψ, and $\mathbf{P}_x = |x\rangle\,\langle x|$ on the state $|x\rangle$.

1.1.9 The generalized Born rule



The stronger form of the Born rule applies when one measures only a single one of $n+1$ Qbits by sending it through a standard 1-Qbit measurement gate. The general form of the $(n+1)$ Qbits is given by

$$|\Psi\rangle_{n+1} = \sum_{x=0}^{2^{n+1}-1} \gamma(x)\,|x\rangle_{n+1}\,, \qquad \sum_{x=0}^{2^{n+1}-1} |\gamma(x)|^2 = 1 \tag{1.69}$$

from which we write explicitly as a combination of the measured Qbit and the other n Qbits:

$$|\Psi\rangle = \alpha_0\,|0\rangle\,|\Phi_0\rangle_n + \alpha_1\,|1\rangle\,|\Phi_1\rangle_n \tag{1.70}$$

where

$$|\Phi_0\rangle_n = \frac{1}{\alpha_0}\sum_{x=0}^{2^n-1}\gamma(x)\,|x\rangle_n\,; \quad |\Phi_1\rangle_n = \frac{1}{\alpha_1}\sum_{x=0}^{2^n-1}\gamma(2^n+x)\,|x\rangle_n \tag{1.71}$$

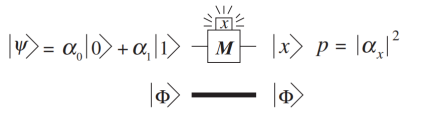
and of course to normalize things

$$\alpha_0^2 = \sum_{x=0}^{2^n-1}|\gamma(x)|^2, \quad \alpha_1^2 = \sum_{x=0}^{2^n-1}|\gamma(2^n+x)|^2. \tag{1.72}$$

Note that $|\Phi_0\rangle_n$ and $|\Phi_1\rangle_n$ are not necessarily orthogonal.

In plain English, the rule asserts that if one measures only the single Qbit whose state symbol is explicitly separated out from the other n Qbits, then the measurement gate will produce x (0 or 1) with probability $|\alpha_x|^2$. The resulting state vector will be the product state $|x\rangle\,|\Phi_x\rangle_n$.

If the Qbit being measured is initially unentangled with the other n Qbits, then the action of the measurement gate on the measured Qbit is just that specified by the Born rule. The unmeasured Qbits are as if they are not there and remain in their original states throughout the process.



Applying the generalized Born rule n times to successive 1-Qbit measurements of each of n Qbits initially in $|\psi\rangle_n$, one can show that the final state is x with probability $|\alpha_x|^2$. Hence, there is only a single primitive piece of measurement hardware: the 1- Qbit measurement gate.

Even more generally, suppose we have the general state of $m+n$ Qbits:

$$|\psi\rangle_{m+n} = \sum_{x=0}^{2^m}\alpha_x\,|x\rangle_m\,|\Phi_x\rangle_n \tag{1.73}$$

where $\sum_x |\alpha_x|^2 = 1$ and the states $|\Phi_x\rangle$ are normalized but not necessarily orthogonal. Applying the Born rule m times to m Qbits we see that if just m Qbits ($|x\rangle_m$) are measured, then with probability $|\alpha_x|^2$ the result will be x, and the resulting state vector will be the product state $|x\rangle_m\,|\Phi_x\rangle_n$.

1.1.10 Measurement gates and state preparation

The measurement gate is useful for producing states with definite states. A measurement that registers x outputs a state in the classical basis $|x\rangle_n$. If we

then **X** each Qbit that registered a 1 in the measurement and do nothing to the Qbits that registered a 0 then the resulting Qbits will be in the state $|0\rangle_n$. Most quantum-computational algorithms take this state as its input.

Measurement gates therefore plays two role: *state preparation* and extracting information from the Qbits.

1.1.11 Constructing arbitrary 1- and 2-Qbit states

We will first consider the case for a single Qbit. Let $|\psi\rangle$ be any 1-Qbit state, and let $|\phi\rangle$ be an orthogonal state, i.e. $\langle\psi|\phi\rangle = 0$. The state $|\phi\rangle$ is unique up to an overall phase. Because $|0\rangle$ and $|1\rangle$ are linearly independent, there is a unique linear transformatin taking them into $|\psi\rangle$ and $|\phi\rangle$. Also, because $|\phi\rangle$ and $|\psi\rangle$ are orthonormal, this linear transformation must be unitary. We will call it **u**, where

$$|\psi\rangle = \mathbf{u}\,|0\rangle\,. \tag{1.74}$$

By showing the existence of such a unitary gate **u**, we're done.

Any arbitrary unentangled 2-Qbit state, which is just a fancy way to say it is a tensor product of two 1-Qbit states, can be constructed out of $|00\rangle$ by applying a tensor product of two unitaries to each of the $|0\rangle$ individual states. This follows from the previous argument. To sum this up, we say that an unentangled state requires an unentangled gate to prepare.

When the 2-Qbit state is entangled, however, its production requires a 2-Qbit gate that is not a tensor product of two single-Qbit unitaries. The solution, as it turns out, is using a combination of a single cNOT gate and some 1-Qbit unitaries. Consider the general 2-Qbit state:

$$|\Psi\rangle = \alpha_{00}\,|00\rangle + \alpha_{01}\,|01\rangle + \alpha_{10}\,|10\rangle + \alpha_{11}\,|11\rangle\,. \tag{1.75}$$

Observe that this can be re-written in the following way:

$$|\Psi\rangle = |0\rangle \otimes |\psi\rangle + |1\rangle \otimes |\phi\rangle \tag{1.76}$$

where $|\phi\rangle = \alpha_{00}\,|0\rangle + \alpha_{01}\,|1\rangle$ and $|\phi\rangle = \alpha_{10}\,|0\rangle + \alpha_{11}\,|1\rangle$.

Next, consider a unitary **u** whose action on the computational basis is the following:

$$\mathbf{u}\,|0\rangle = a\,|0\rangle + b\,|1\rangle\,, \quad \mathbf{u}\,|1\rangle = -b^*\,|0\rangle + a^*\,|1\rangle \tag{1.77}$$

where $|a|^2 + |b|^2 = 1$, as usual. Applying $\mathbf{u} \otimes \mathbb{I}$ to $|\Psi\rangle$, we get

$$\begin{aligned}(\mathbf{u} \otimes \mathbb{I})\,|\Psi\rangle &= (a\,|0\rangle + b\,|1\rangle) \otimes |\psi\rangle + (-b^*\,|0\rangle + a^*\,|1\rangle) \otimes |\phi\rangle \\ &= |0\rangle \otimes |\psi'\rangle + |1\rangle \otimes |\phi'\rangle\end{aligned} \tag{1.78}$$

where

$$|\psi'\rangle = a\,|\psi\rangle - b^*\,|\phi\rangle\,; \quad |\phi'\rangle = b\,|\psi\rangle + a^*\,|\phi\rangle\,. \tag{1.79}$$

Now, if $|\psi'\rangle$ and $|\phi'\rangle$ are orthogonal, then we're done. We wish to choose complex numbers a, b to achieve this. Consider the inner product of $|\psi'\rangle$ and $|\phi'\rangle$:

$$\langle\phi'|\psi'\rangle = a^2\,\langle\phi|\psi\rangle - b^2\,\langle\phi|\phi\rangle + ab^*\,(\langle\psi|\psi\rangle - \langle\phi|\phi\rangle) \tag{1.80}$$

If $\langle\psi|\phi\rangle \neq 0$ then setting $\langle\phi'|\psi'\rangle$ to 0 gives an quadratic equation for the ratio $a/b^* = 0$, for which there are two complex solutions. Setting a determines a $\mathbf{u}$ for which

$$(\mathbf{u}\otimes\mathbb{I})\,|\Psi\rangle = |0\rangle\otimes|\psi'\rangle + |1\rangle\otimes|\phi'\rangle \tag{1.81}$$

where $\langle\psi'|\phi'\rangle = 0$. If $\langle\phi|\psi\rangle = 0$ then there's nothing else for us to do. $\mathbf{u}\equiv\mathbb{I}$.

Let λ, μ be positive reals such that

$$|\psi''\rangle = \frac{|\psi'\rangle}{\lambda}, \quad |\phi''\rangle = \frac{|\phi'\rangle}{\mu} \tag{1.82}$$

are normalized. This makes $|\psi''\rangle$ and $|\phi''\rangle$ an orthonormal pair. Following an argument earlier, there exists a unitary $\mathbf{v}$ for which

$$|\psi''\rangle = \mathbf{v}\,|0\rangle\,, \quad |\phi''\rangle = \mathbf{v}\,|1\rangle\,. \tag{1.83}$$

From here, we see that

$$|\Psi\rangle = \left(\mathbf{u}^\dagger\otimes\mathbf{v}\right)(\lambda\,|0\rangle\otimes|0\rangle + \mu\,|1\rangle\otimes|1\rangle)\,. \tag{1.84}$$

Now, note that $\mathbf{C}_{10}\,|00\rangle \to |00\rangle$ and $\mathbf{C}_{10}\,|10\rangle \to |11\rangle$. So we can write the equality as follows

$$|\Psi\rangle = \left(\mathbf{u}^\dagger\otimes\mathbf{v}\right)\mathbf{C}_{10}\,(\lambda\,|0\rangle + \mu\,|1\rangle)\otimes|0\rangle\,. \tag{1.85}$$

But we're not finished. Remember that we wish to obtain $|\Psi\rangle$ from $|00\rangle = |0\rangle\otimes|0\rangle$. However, we're very close now, as we only need to obtain $\lambda\,|0\rangle + \mu\,|1\rangle$ from $|0\rangle$. Fortunately, we already know how to do this! Consider another unitary $\mathbf{w}$ for which

$$\mathbf{w}\,|0\rangle = \lambda\,|0\rangle + \mu\,|1\rangle \tag{1.86}$$

whose existence is guaranteed by our earlier arguments. So, with a single cNOT and three unitaries, we can prepare any entangled 2-Qbit state from $|00\rangle$ via

$$|\Psi\rangle = \left(\mathbf{u}^\dagger\otimes\mathbf{v}\right)\mathbf{C}_{10}\,(\mathbf{w}\otimes\mathbb{I})\,|00\rangle\,. \tag{1.87}$$

1.2 Examples

1.2.1 The general computational process

In general, we need at least $n+m$ Qbits for computations, where n is the number of Qbits required to specify the input x, and m is the number of Qbits required to specify the output $f(x)$. The set of n Qbits is called the *input register*. The set of m Qbits is called the *output register*. Havig separate input and output registers is standard practice in the classical theory of reversible computation. Since quantum computers must operate reversibly, they are generally designed to operate with both input and output registers.

We view a quantum computation $f$on $n+m$ Qbits as a unitary transformation $\mathbf{U}_f$ acting on the $n+m$ Qbit states:

$$\mathbf{U}_f(|x\rangle_n |y\rangle_m) = |x\rangle_n |y \oplus f(x)\rangle_m \tag{1.88}$$

where $\oplus$ indicates mod 2 addition, or XOR. If $y = 0$ then

$$\mathbf{U}_f(|x\rangle_n |0\rangle_m) = |x\rangle_n |f(x)\rangle_m \tag{1.89}$$

so we have $f(x)$ in the output register, while the input register $|x\rangle_n$ is intact (and is independent on the state of y).

Clearly, $\mathbf{U}_f$ is its own inverse:

$$\mathbf{U}_f\mathbf{U}_f(|x\rangle |y\rangle) = \mathbf{U}_f(|x\rangle |y \oplus f(x)\rangle) = |x\rangle |y \oplus f(x) \oplus f(x)\rangle = |x\rangle |y\rangle . \tag{1.90}$$

Now, if we apply to each Qbit in the 2-Qbit state $|0\rangle |0\rangle$ the 1-Qbit Hadamard, then we get

$$\begin{aligned}(\mathbf{H} \otimes \mathbf{H})(|0\rangle \otimes |0\rangle) &= (\mathbf{H} |0\rangle) \otimes (\mathbf{H} |0\rangle) \\ &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \otimes \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle) \\ &= \frac{1}{2}(|00\rangle + |01\rangle + |10\rangle + |11\rangle) \\ &= \frac{1}{2}(|0\rangle_2 + |1\rangle_2 + |2\rangle_2 + |3\rangle_2).\end{aligned} \tag{1.91}$$

This generalizes to the n-fold tensor product of n Hadamards, applied to the n-Qbit state $|0\rangle_n$:

$$\mathbf{H}^{\otimes n} |0\rangle_n = \frac{1}{2^{n/2}} \sum_{0 \le x \le 2^n} |x\rangle_n \tag{1.92}$$

If the initial state of the input register is $|0\rangle_n$, then the output state is the perfect superposition of all possible n-Qbit inputs. If we than apply $\mathbf{U}_f$ to that

superposition, with 0 initially in the output register, then we get

$$\begin{aligned}\mathbf{U}_f(\mathbf{H}^{\otimes n}\otimes\mathbb{I}_m)(|0\rangle_n|0\rangle_m) &= \frac{1}{2^{n/2}}\sum_{0\le x\le 2^n}\mathbf{U}_f(|x\rangle_n|0\rangle_m)\\ &= \frac{1}{2^{n/2}}\sum_{0\le x\le 2^n}|x\rangle_n|f(x)\rangle_m\,. \end{aligned}\tag{1.93}$$

Why is this important? This is called *quantum parallelism*. If before letting $\mathbf{U}_f$ act, we only apply the $\mathbf{H}$ transformation to every Qbit of the input register (all initially in 0), then the result of the computation s described by a state whose structure cannot be explicitly specified without knowing the result of all 2^n evaluations of the function f. So, if we start with 100 Qbits in the input register, if a hundred Hadamard gates act on the input register before the application of $\mathbf{U}_f$, then the form of the final state contains the results of $2^{100}\approx 10^{30}$ evaluations of the function f!

Another "miracle" is that we can't say the result of the computation is *actually* 2^n evaluations of f, because when we have a collection of Qbits in a definite but unknown state, there is no way to find out what that state is.

1.2.2 No-cloning theorem

If there were a way to make copies of the output state prior to making measurements (without running the whole computation again) then one could learn the values of f for several different values of x. However, this is forbidden by the so-called *no-cloning theorem*.

The no-cloning theorem is a consequence of linearity. Suppose to get a contradiction that cloning is possible, i.e.

$$\mathbf{U}(|\psi\rangle|0\rangle)=|\psi\rangle|\psi\rangle\,,\quad \mathbf{U}(|\phi\rangle|0\rangle)=|\phi\rangle|\phi\rangle \tag{1.94}$$

then from linearity:

$$\mathbf{U}(a|\psi\rangle+b|\phi\rangle)|0\rangle = a|\psi\rangle|\psi\rangle+b|\phi\rangle|\phi\rangle\,. \tag{1.95}$$

But if $\mathbf{U}$ closed arbitrary inputs, we would also have

$$\begin{aligned}\mathbf{U}(a|\psi\rangle+b|\phi\rangle)|0\rangle &= (a|\psi\rangle+b|\phi\rangle)(a|\psi\rangle+b|\phi\rangle)\\ &= a^2|\psi\rangle|\psi\rangle+b^2|\phi\rangle|\phi\rangle+ab|\phi\rangle|\phi\rangle+ab|\phi\rangle|\psi\rangle\\ &\neq a|\psi\rangle|\psi\rangle+b|\phi\rangle|\phi\rangle \end{aligned}\tag{1.96}$$

unless $ab=0$.

Okay, but what about creating a reasonable, but not quite exact, clone? Suppose

$$\mathbf{U}(|\psi\rangle|0\rangle)\approx|\psi\rangle|\psi\rangle\,,\quad \mathbf{U}(|\phi\rangle|0\rangle)\approx|\phi\rangle|\phi\rangle \tag{1.97}$$

then since $\mathbf{U}$ preserves the inner product we must have that

$$\langle\phi|\psi\rangle \approx \langle\phi|\psi\rangle^2 . \tag{1.98}$$

But this means that $\langle\phi|\psi\rangle$ is close to either 0 or 1. This means that $\mathbf{U}$ does well only if $|\psi\rangle$ and $|\phi\rangle$ are very nearly the same, or are very nearly orthogonal. In any case, $\mathbf{U}$ isn't too useful.

1.2.3 Deutsch's problem

The Deutsch's problem is an example of a quantum tradeoff that sacrifices particular information to acquire relational information.

Consider only functions f that take a single bit into a single bit. There are only 4 such functions, since there are two possibilities for the input and two possibilities for the output. These functions are

$$\mathbf{U}_{f_0} \equiv \mathbb{I} \tag{1.99}$$
$$\mathbf{U}_{f_1} \equiv \mathbf{C}_{io} \tag{1.100}$$
$$\mathbf{U}_{f_2} \equiv \mathbf{C}_{io}\mathbf{X}_o \tag{1.101}$$
$$\mathbf{U}_{f_3} \equiv \mathbf{X}_o \tag{1.102}$$

where of course i, o are for *input* and *output*, respectively. In general,

$$\mathbf{U}_f(|x\rangle\,|y\rangle) = |x\rangle\,|y \oplus f(x)\rangle\,. \tag{1.103}$$

Schematically, this looks like When f is unknown, we can find out what f is by

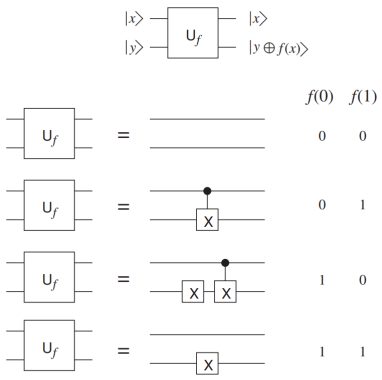


Figure 1.1: By Mermin

letting $\mathbf{U}_f$ act twice, once on $|0\rangle\,|0\rangle$ and then on $|1\rangle\,|0\rangle$. But what if we could only let it act once? And what if we are only interested in knowing whether f is constant (in which case it is f_1 or f_3) or not constant (f_2 or f_4)? With

a classical computer, we will have to evaluate both $f(1)$ and $f(0)$, but with a quantum computer we only need to let $\mathbf{U}_f$ act once, only with the tradeoff of not knowing the values of $f(0)$ and $f(1)$.

To do this we use the Hadamard to put the input into superposition:

$$\mathbf{U}_f(\mathbf{H}\otimes\mathbb{I})(|0\rangle|0\rangle) = \frac{1}{\sqrt{2}}|0\rangle|f(0)\rangle + \frac{1}{\sqrt{2}}|1\rangle|f(1)\rangle. \tag{1.104}$$

This doesn't solve the problem, but we see that we can make further unitary transformations to the state above before making measurements that enable us half the time to state with assurance whether or not $f(0) = f(1)$.

Suppose that instead of starting with letting the Hadamard act on the input of $|0\rangle|0\rangle$, we let two Hadamards act on $|1\rangle|1\rangle$, i.e.

$$\begin{aligned}(\mathbf{H}\otimes\mathbf{H})(\mathbf{X}\otimes\mathbf{X})(|0\rangle|0\rangle) &= (\mathbf{H}\otimes\mathbf{H})(|1\rangle\otimes|1\rangle)\\ &= \left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)\left(\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle\right)\\ &= \frac{1}{2}\left(|0\rangle|0\rangle - |1\rangle|0\rangle - \frac{0}{1} + |1\rangle|1\rangle\right).\end{aligned} \tag{1.105}$$

Then, letting $\mathbf{U}_f$ act on this state to get

$$\begin{aligned}\mathbf{U}_f(\mathbf{H}\otimes\mathbf{H})(\mathbf{X}\otimes\mathbf{X})(|0\rangle|0\rangle) &= \ldots\\ &= \frac{1}{2}\left(|0\rangle|f(0)\rangle - |1\rangle|f(1)\rangle - |0\rangle|1\oplus f(0)\rangle + |1\rangle|1\oplus f(1)\rangle\right)\\ &\equiv \frac{1}{2}\left(|0\rangle|f(0)\rangle - |1\rangle|f(1)\rangle - |0\rangle\left|\tilde{f}(0)\right\rangle + |1\rangle\left|\tilde{f}(1)\right\rangle\right)\end{aligned} \tag{1.106}$$

where we have defined

$$\tilde{f} \equiv 1\oplus f. \tag{1.107}$$

Now, if $f(0) = f(1)$ then the state above becomes

$$\frac{1}{2}\left(|0\rangle - |1\rangle\right)\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), \quad f(0) = f(1) \tag{1.108}$$

else ($f(0) \neq f(1)$) then the state above becomes

$$\frac{1}{2}\left(|0\rangle + |1\rangle\right)\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), \quad f(0) \neq f(1) \tag{1.109}$$

since $f(1) = \tilde{f}(0)$ whenever $f(0) \neq f(1)$.

Now, because

$$\begin{aligned}\mathbf{H}\left(|0\rangle \pm |1\rangle\right) &= \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right) \pm \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \\ &= \begin{cases} \sqrt{2}\,|0\rangle\,, & (+) \\ \sqrt{2}\,|1\rangle\,, & (-) \end{cases}\,, \end{aligned} \tag{1.110}$$

when we let $\mathbf{H}$ act on the *input* register of the state above, we have

$$\begin{cases} |1\rangle \frac{1}{\sqrt{2}}\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), & f(0) = f(1) \\ |0\rangle \frac{1}{\sqrt{2}}\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), & f(0) \neq f(1) \end{cases}\,. \tag{1.111}$$

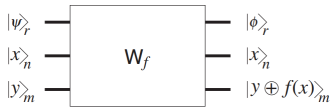
Putting everything together we have

$$\boxed{(\mathbf{H}\otimes\mathbb{I})\mathbf{U}_f(\mathbf{H}\otimes\mathbf{H})(\mathbf{X}\otimes\mathbf{X})(|0\rangle\,|0\rangle) = \begin{cases} |1\rangle \frac{1}{\sqrt{2}}\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), & f(0) = f(1) \\ |0\rangle \frac{1}{\sqrt{2}}\left(|f(0)\rangle - \left|\tilde{f}(0)\right\rangle\right), & f(0) \neq f(1) \end{cases}} \tag{1.112}$$

Now, notice that the state of the input register will be $|1\rangle$ if $f(0) = f(1)$, and $|0\rangle$ if $f(0) \neq f(1)$. So, by measuring the input register, we can tell if f is constant.

Notice that the output register in both cases are the same (differ only in the sign, but that doesn't make the state distinguishable). But in any case, we learn whether or not f is constant by calling $\mathbf{U}_f$ only once.

1.2.4 Why additional Qbits needn't mess things up



We use $n+m$ Qbits to perform quantum computations, where n is the number of Qbits in the input register and m in the output register. In general, however, we also need r additional Qbits. The action of the computer on these Qbits is then described by a unitary $\mathbf{W}_f$, which acts on all $m+n+r$ Qbits. Only in special cases do we have $\mathbf{W}_f$ act separately on the input and output registers and leaving the additional Qbits alone. In general, the r additional Qbits get entangled with the $m+n$ Qbits too.

Let the initial state be

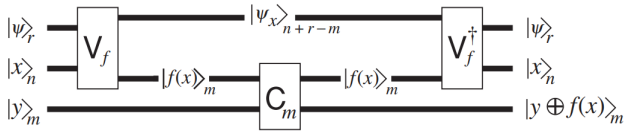
$$|\Psi\rangle_{n+m+r} = |x\rangle_n\,|y\rangle_m\,|\psi\rangle_r \tag{1.113}$$

then applying the global unitary gate $\mathbf{W}_f$ gives:

$$\mathbf{W}_f \left|\Psi\right\rangle_{n+m+r} = \left|x\right\rangle_n \left|y + \oplus f(x)\right\rangle_m \left|\phi\right\rangle_r . \tag{1.114}$$

The state of the addition Qbits is not necessarily intact. However, we require that the additional Qbits are not only unentangled with the input and output registers, but also its state $\left|\phi\right\rangle_r$ is independent of the initial state of the input and output registers. When this is the case, the action of $\mathbb{W}_f$ on $n+m$ Qbits can be represented by the unitary $\mathbf{U}_f$ on the subspace $n+m$ alone, while the r additional Qbits are taken from the initial pure state $\left|\psi\right\rangle_r$ to a final pure state $\left|\phi\right\rangle_r$ that is independent of the initial contents of the input and output registers.

So how do we construct $\mathbf{W}_f$ so that it acts as if $\mathbf{U}_f$ is acting on the $n+m$ Qbits alone and leaving the additional Qbits intact? Schematically, this looks like



There are three steps to the solution:

1. First, we apply a unitary transformation $\mathbf{V}$ that acts only on the n and r input register and additional Qbits. We can design the $\mathbf{V}$ gate such that if the initial input state is $\left|x\right\rangle_n$ and $\left|\psi\right\rangle_r$ then the output is $\left|f(x)\right\rangle_m$ and $\left|\psi\right\rangle_{n+r-m}$. We will then proceed to feed $\left|f(x)\right\rangle_m$ to the next gate.

2. In this step, we let y change into $y \oplus f(x)$ without changing the states of the other $n+r$ Qbits. This can be done with m CNOT gates that combine to make a single unitary $\mathbf{C}_m$ gate. The control bits here will be the bits in $\left|f(x)\right\rangle_m$, so they remain unchanged. Only the $\left|y\right\rangle_m$ Qbits states get changed.

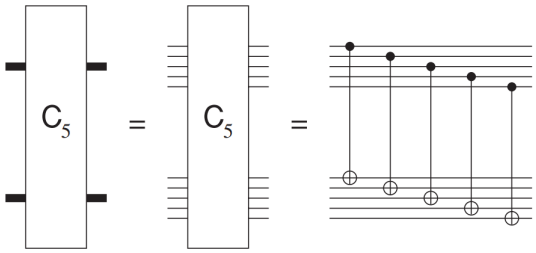
3. Finally, because the state of the $n+r$ Qbits is not altered by $\mathbf{C}_m$, we simply let $\mathbf{V}^\dagger$ reverse these back to their original states, to give us the original $\left|\psi\right\rangle_r$, $\left|x\right\rangle_n$, with $\left|y \oplus f(x)\right\rangle_m$

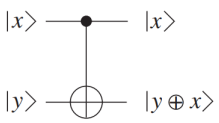
As an aside, the m CNOT gate, for $m = 5$, looks like

1.2.5 The Bernstein-Vazirani problem

Consider the function $f(x) = a \cdot x$, where $a \cdot x$ denotes the the mod-2 sum of the products of the corresponding bits of a and x, i.e.,

$$a \cdot x = \bigoplus_{i=0} a_0 x_0 \tag{1.115}$$





where of course a_i, x_i take values $\{0, 1\}$, and a is a non-negative integer less than 2^n. The problem is to find a. More specifically, suppose we have a subroutine which evaluates $f(x) = a \cdot x$, how many times do we have to query $f(x)$ in order to find what a is? Note that the a rule one has to follow here is that any additional Qbits used by the algorithm (so, except for the input and output registers) must return to their initial state once the evaluation has completed. Notice that the answer we get is either a 0 or 1, so we need only one Qbit in the output register for this.

Classically, here's how we do it. Let a number a be given. Because the expansion of 2^m has a 1 in position m and 0 in all other positions, the m^{th} bit of a is $a \cdot 2^m$ (it's easy to see how this works by writing down the mod-2 inner product). This means that with a classical computer, we can learn the values of n bits of a by applying f to $x = 2^m$, $m \in [0, n]$, n times. This requires querying f n times.

Quantum computers, on the other hand, requires a **single** query to completely determine a, regardless of how big n is. There are two ways to discuss how this works. The first way goes into the algebra, while the second gives pictorial, circuit-theoretical approach to solving this problem. I won't worry about the second approach and will only focus on the first approach.

The (conventional) first way to do this exploits a trick that is useful in dealing with functions like f that act on n Qbits without output to a single Qbit. Suppose the 1-Qbit output register is initially prepared in the state $\mathbf{HX}\,|0\rangle = \mathbf{H}\,|1\rangle = (1/\sqrt{2})(|0\rangle - |1\rangle)$. We know that

$$\mathbf{U}_f\,|x\rangle_n\,|y\rangle_1 = |x\rangle_n\,|y \oplus f(x)\rangle = |x\rangle_n\,|\bar{y}\rangle_1 \iff f(x) = 1. \tag{1.116}$$

This means

$$\mathbf{U}_f \left|x\right\rangle_n \frac{1}{\sqrt{2}} \left(\left|0\right\rangle - \left|1\right\rangle\right) = (-1)^{f(x)} \left|x\right\rangle_n \frac{1}{\sqrt{2}} \left(\left|0\right\rangle - \left|1\right\rangle\right). \tag{1.117}$$

We see that by taking the state of the 1-Qbit output register to be $(1/\sqrt{2})(|0\rangle - |1\rangle)$, we convert a bit flip to an overall change of sign. This is the first trick. The second trick is not exactly a trick, but a fact that we will see in the homework:

$$\mathbf{H}^{\otimes n} \left|x\right\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} (-1)^{x \cdot y} \left|y\right\rangle. \tag{1.118}$$

With this, if we start with the n-Qbit input register in the standard initial state $\mathbf{H}^{\otimes n}|0\rangle$, put the 1-Qbit output register into $\mathbf{H}|1\rangle$, apply $\mathbf{U}_f$, and then again apply $\mathbf{H}^{\otimes n}$ to the input register we will get

$$(\mathbf{H}^{\otimes n} \otimes \mathbb{I}) \mathbf{U}_f (\mathbf{H}^{\otimes n} \otimes \mathbf{H}) \left|0\right\rangle_n \left|1\right\rangle_1 = \frac{1}{2^n} \sum_{0 \le x,y < 2^n} (-1)^{f(x)+x\cdot y} \left|y\right\rangle \frac{1}{\sqrt{2}} \left(\left|0\right\rangle - \left|1\right\rangle\right). \tag{1.119}$$

I will spare the reader the details of this calculation. Parts of this calculation will be verified in the exercises. By inspection, however, the reader should be able to convince himself this result makes sense.

From here, we do sum over x first. If $f(x) = a \cdot x$ then the sum over x produces the factor

$$\sum_{0 \le x < 2^n} (-1)^{a\cdot x + x \cdot y} = \prod_{j=1}^{n} \sum_{x_j \in \{0,1\}} (-1)^{(a_j + y_j) x_j}. \tag{1.120}$$

At least one term in the product is zero unless every bit y_j of y is equal to the corresponding bit a_j of a, i.e., unless $y = a$ (check this by inspection). Therefore, if I apply a final $\mathbf{H}$ onto the 1-Qbit output register to make things a little nicer, then the entire computational process reduces to

$$\mathbf{H}^{\otimes(n+1)} \mathbf{U}_f \mathbf{H}^{\otimes(n+1)} \left|0\right\rangle_n \left|1\right\rangle_1 = \left|a\right\rangle_n \left|1\right\rangle_1. \tag{1.121}$$

So by putting the input and output registers into the appropriate initial states, after a single call for the subroutine followed by $\mathbf{H}^{\otimes n}$ to the input register, the state of the input register becomes $|a\rangle$. Now, all n bits of the number a can ne determined by measuring the input register, even though we have called the subroutine only once.

1.2.6 Simon's problem

In the Bernstein-Vazirani problem, a classical computer must call the subroutine n times, while a quantum computer only need to call the subroutine once. As

the problem grows, the classical complexity scales linearity, while the quantum complexity is still constantly one. In the Simon's problem, the speed-up is even more dramatic. The classical complexity grows exponentially with the size of the problem, while the quantum complexity only grows linearly.

This speed-up involves a probabilistic element characteristic of many quantum computations. The characterization of how the number of calls of the subroutine scales with the number of bits in a applies not to calculating directly, but to learning it with probability very close to 1.

The subroutine $\mathbf{U}_f$ in Simon's problem evaluates a function f on n bits is two to one, i.e., it is a function from n to $n-1$ bits. It is constructed so that the n-bit integers x and y are related by $x = y \oplus a \iff x \oplus y = a$. This is essentially a period-finding problem:

$$f(x \oplus a) = f(x). \tag{1.122}$$

Of course the period here is a. Simon's problem is a precursor of Shor's period-finding algorithm where one finds the period a under the ordinary addition $f(a + x) = f(x)$.

With a classical computer, what you have to do to find a is feeding the subroutine with a bunch of x, and compare $f(x \oplus a)$ against $f(x)$ until you see a match. At any stage of the process prior to success, if you have picked m different values of x, then all you know is that $a \neq x_i \oplus x_j$ for all pairs x_i, x_j. This means you have eliminated at most $(m/2)(m-1)$ values of a. Now, there are $2^n - 1$ possible values for a. The chance of success won't be large unless $(m/2)(m-1)$ is sufficiently close to 2^n. This means $m \sim 2^{n/2}$ is required. This means the number of trials grows exponentially.

In (spectacular) contrast, a quantum computer can determine a with not much more than n queries. Let's see how this is done:

To do this we first return to the standard procedure and let $\mathbf{U}_f$ act on $\mathbf{H}^{\otimes n} |0\rangle_n$. This gives

$$\mathbf{U}_f(\mathbf{H}^{\otimes} |0\rangle_n) = \frac{1}{2^{n/2}} \sum_{0 \leq x < 2^n} |x\rangle |f(x)\rangle \tag{1.123}$$

which the reader can verify from the results in the exercises.

Now, if we subject only the output register to a measurement, then the measurement gate is equally likely to indicate each of the 2^{n-1} different values of f. Looking at the RHS of the equation above again, since f is periodic, we know that each value of f appears in two terms that have the same amplitudes. The Born rule (of partial measurement) tells us that the input register must be

in the state

$$\frac{1}{\sqrt{2}}\left(|x_0\rangle + |x_0 \oplus a\rangle\right) \tag{1.124}$$

for that value x_0 for which $f(x_0)$ agrees with the random value of f given by the measurement.

This looks good, because we just created a superposition of just two computational basis states, associated with two n-bit integers, that differ in $\oplus$ by a. If we knew these two integers their bitwise mod 2 sum would be a. However, when a register is in quantum state there is no way to learn what state it is in. By subjecting the state above to a direct measurement all we can learn is either x_0, a random number, or $x_0 \oplus a$, another random number. a appears only in the relation between these two integers, only one of which we know.

But we can cure this problem. Just like in the Deutsch's problem, we can ignore the possibility of learning either number. It is possible that by applying some operations before measuring can let us extract the relation a. This operation is $\mathbf{H}^{\otimes n}$:

$$\begin{aligned}\mathbf{H}^{\otimes n}\frac{1}{\sqrt{2}}\left(|x_0\rangle + |x_0 \oplus a\rangle\right) &= \frac{1}{\sqrt{2^{n+1}}}\sum_{0\le y<2^n}\left((-1)^{x_0\cdot y} + (-1)^{(x_0\oplus a)\cdot y}\right)|y\rangle \\ &= \frac{1}{\sqrt{2^{n-1}}}\sum_{a\cdot y=0}(-1)^{x_0\cdot y}\,|y\rangle\,. \end{aligned}\tag{1.125}$$

If we now measure the input register, we learn with equal probability any of the values of y for which $a \cdot y = 0$. With this, each call for $\mathbf{U}_f$ gives us a y such that $a \cdot y = 0$.

Next, we want to show that this enables us to determine a with high probability with not many more than n calls for $\mathbf{U}_f$. With a single call for $\mathbf{U}_f$, unless we get $y = 0$ (which is unlikely), we learn a nonzero value of y, and therefore a nontrivial subset of the n bits of a whose mod 2 sum vanishes. This means we have cut the number of possible choice for a by a half, from $2^n - 1$ to $2^{n-1} - 1$. If we repeat this procedure, we will likely learn another nonzero value of y that isn't a repeat of the previous y. This enables us to again halve the number of possible candidates for a. It turns out (after from mathematical analysis) that with $n + x$ calls for $\mathbf{U}_f$ the probability q of acquiring enough information to determine a is

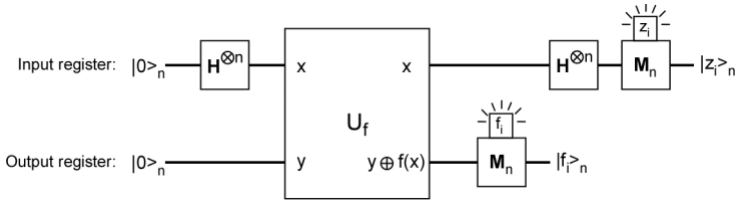
$$q = \left(1 - \frac{1}{2^{x+n}}\right)\left(1 - \frac{1}{2^{x+n-1}}\right)\ldots\left(1 - \frac{1}{2^{x+2}}\right) \ge 1 - \frac{1}{2^{x+1}}. \tag{1.126}$$

1.3 Breaking NSA encryption

1.3.1 Period finding, factoring, and cryptography

Background: In 1993 Dan Simon found a quantum algorithm that can efficiently find a hidden "period" a for a function defined by

$$f(x \oplus a) = f(x). \tag{1.127}$$



Shortly afterwards, in 1994, Peter Shor used a very similar approach to finding the hidden period of a function

$$f(x + k) = f(x) = b^x \mod N \tag{1.128}$$

Our goal is to explain why this is an interesting problem.

1.3.2 Number-theoretic preliminaries

There is a relationship between finding the period of a function

$$f(x) = a^x \mod N \tag{1.129}$$

$$a^{x+k} = a^x \mod N. \tag{1.130}$$

For an integer $N = pq$, where p and q are primes and finding the prime factors p and q of N.

Why is that interesting? $\Longrightarrow$ Because public Key encryption relies on the fact that prime factorization of a composite number is a "hard" (exponential in the number of bits) problem.

I will skip over the rest of the group theory related to this (refer to Abstract Algebra notes for this). But in any case, some of the things we'll need are: Euclid's algorithm, group theory, modular arithmetics. We will also be interested in the group

$$G_p = \{1, 2, \ldots, p-2, p-1\}. \tag{1.131}$$

Fermat's Little Theorem: For every $a \in G_p$, $a^{p-1} \equiv 1 \mod p$. A group $G_{N=pq}$ with N a product of two prime numbers p and q has the order $(p -$

$1)(q-1)$.

Extension to Fermat's Little Theorem: For every $a \in G_{pq}$, $a^{(p-1)(q-1)} \equiv 1 \mod pq$.

Lagrange's Theorem: The order of any subgroup is a divisor of the order of the group.

An interesting observation:

$$a^k \equiv 1 \mod N \implies a^k - 1 \equiv 0 \mod N. \tag{1.132}$$

If k is even (which should be roughly half the time) we can write:

$$a^k \equiv 1 \mod N \implies \left(a^{k/2} - 1\right)\left(a^{k/2} + 1\right) \equiv 0 \mod N. \tag{1.133}$$

Example 1.3.1. When $N = 21, a = 2, k = 6$, we get $2^{6/2} - 1 \equiv 7 \mod N$ and $2^{6/2} + 1 \equiv 9 \mod N$. Now, $\gcd(7, 21) = 7$ and $\gcd(9, 21) = 3$. These are the prime factors of 21.

Note: we can get unlucky for some values of a where we don't learn anything. In these cases, we just try again with a different a.

1.3.3 RSA encryption

Private Key Encryption

In private key encryption two parties (Alice and Bob, of course) know a "secret key" that allows them to both encrypt and decrypt a message. A perfectly secure system is a "one-time pad" containing a list of random characters, one for each character in the message.

Example 1.3.2. Consider the letter M in ASCII encoding: $M = 01001101$. And the secret code is $C = 01000011$. The encoded message is $E = M \oplus C = 00001110$. To decode, we just do $D = E \oplus C = 01001101 = M$.

Public Key Encryption

In public key encryption there are two keys, a public key used for encoding and a private key used for decoding.

The key (sorry) to public key encryption is that even if you know the public key it is hard (mathematically hard) to figure out what the private key is, but if you generate a private key it is easy (mathematically easy) to generate the associated public key.

Every time you do an online transaction sending your credit card number to a company via your web browser you use public key encryption. In particular

you use RSA (named RSA after its inventors Rivest, Shamir, and Adleman who designed it in 1977).

In RSA the private (decoding) key requires knowledge of two prime numbers p and q and the public (encoding) key requires knowledge of only the composite number N = pq. The fact that finding primes is easy, creating a composite is easy, and factoring a composite is hard is the basis for RSA encryption.

RSA Encryption

In RSA the private (decoding) key requires knowledge of two prime numbers p and q and the public (encoding) key requires knowledge of only the composite number N = pq. The fact that finding primes is easy, creating a composite is easy, and factoring a composite is hard is the basis for RSA encryption.

To generate the two keys Bob choose two large prime numbers p and q which he turns into a composite number $N = pq$ and an encoding number (also large) c that is relatively prime with $(p-1)(q-1)$.

So, the public key is going to be $\{N, c\}$. For example, say this is $\{91, 11\}$ where $p = 7, q = 13$. Bob creates decoding key d such that $cd = 1 \mod (p-1)(q-1)$. The private key is then $\{N, d\}$. For example, this can be $\{91, 59\}$.

Here's how RSA encryption works:

- **Encoding:** Alice wants to send a private message to Bob. Bob creates a public key and sends it to Alice over a public channel.
  - Alice creates a numerical representation of the message $\{a_1, \ldots, a_n\}$
  - Alice encodes the message using the pubic key to create $\{e_1, \ldots, e_n\}$ where $e_i = a_i^c \mod N$.
  - Alice sends the encoded message to Bob over a public channel.
- **Decoding:** Bob needs to decode the message. (In fact, anyone who can figure out the prime factors of N = 91 could do this, but factoring is hard!)
  - Bob decodes the message using the private key to create $\{m_1, \ldots, m_n\}$ where $m_i = e_i^d \mod N$.

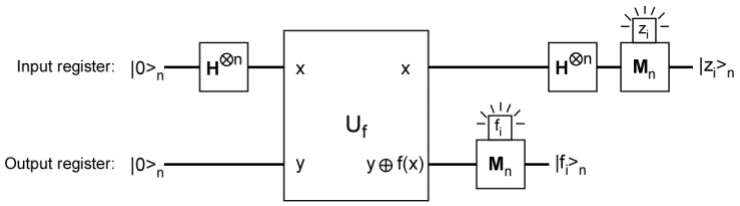
There is a proof in Mermin, page 66, that decoding always works. The reader can also try to prove this on his own.

1.4 Shor's Algorithm & The Quantum Fourier Transform

1.4.1 Background

Background: In 1993 Dan Simon found a quantum algorithm that can efficiently find a hidden "period" a for a function defined by

$$f(x \oplus a) = f(x). \tag{1.134}$$



Shortly afterwards, in 1994, Peter Shor used a very similar approach to finding the hidden period of a function

$$f(x + k) = f(x) = b^x \mod N \tag{1.135}$$

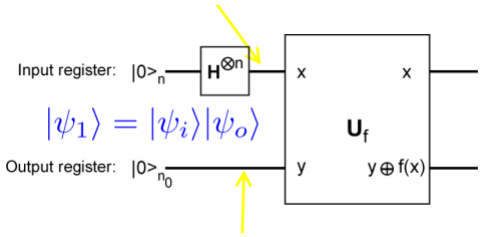
Goal: Explain the approach needed to solve the problem!

1.4.2 Factoring a number quantum mechanically

Shor realized that there's an equivalence between factoring a number N and finding the period (order) r of the function

$$f(x) = b^x \mod N. \tag{1.136}$$

Quantum parallelism: Shor's algorithm starts the same way as Simon's except in this case the "oracle" computes a known function.



The state of the input register after the Hadamard is of course

$$|\psi_i\rangle_n = \mathbf{H}^{\otimes n} |0\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le x < 2^n} |x\rangle_n . \tag{1.137}$$

And the output is

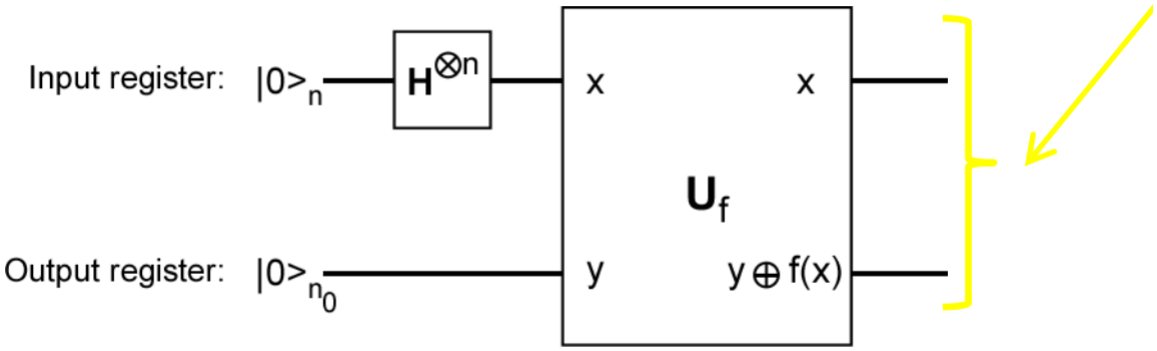
$$|\psi_0\rangle_{n_0} = |0\rangle_{n_0} \tag{1.138}$$

n is the number of bits needed to represent the number of x values you need to evaluate $n = 2n_0$. n_0 is the number of bits needed to represent N.

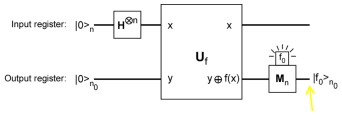
Output of the Oracle: The output of the (linear) oracle is a state that is entangled between the input and output registers:

$$|\psi_2\rangle = \frac{1}{\sqrt{2^n}} \sum_{0 \le x < 2^n} |x\rangle_n |f(x)\rangle_{n_0} . \tag{1.139}$$

All possible x values and their associated f(x) values are equally weighted.



Manipulating the output to get an answer: The output register is measured, and gives specific value of $f(x)$, which is drawn with equal probability from all possible values of $f(x)$.



Of course after the measurement, $|\psi_0\rangle_{n_0} = |f_0\rangle_{n_0}$ where $f_0 = b^{x_0} \mod N$, and

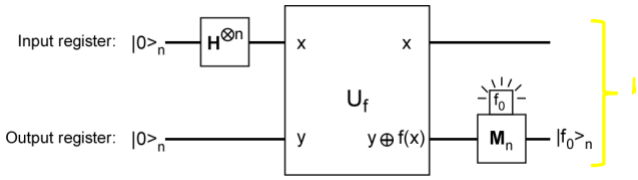
$$|f_0\rangle_{n_0} = |f(x_0)\rangle_{n_0} = |f(x_0 + kr)\rangle_{n_0} \tag{1.140}$$

for any integer k and the period r.

There are many (m) possible values of the input register that are consistent with a specific value of the output register. Here m is the number of states that have k values that satisfy

$$x_0 + kr \leq 2^n \implies m = \left[\frac{2^n}{r}\right]. \tag{1.141}$$

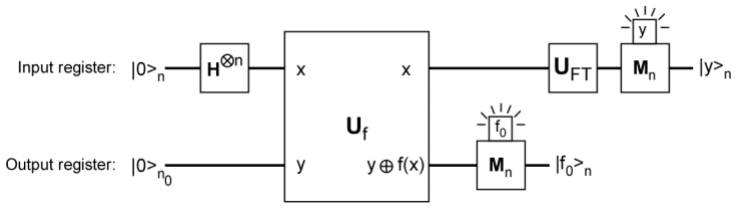
where



$$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle_n |f_0\rangle_{n_0}. \tag{1.142}$$

If we could just clone the output state, we could measure the state multiple times and determine a set of values separated by multiples of r (and therefore r), but the no-cloning theorem says we're out of luck with that approach!

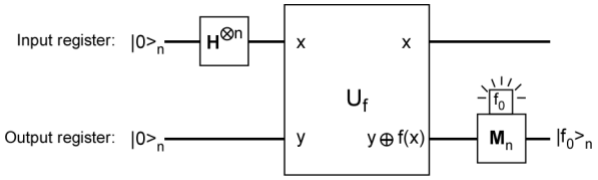
Manipulating the output to get an answer, Part 2: We need to do something more clever than the n-Qbit Hadamard to figure out the periodicity of the state. That thing is the "Quantum Fourier Transform," which preferentially populates states $|y\rangle_n$ that come at integer factors of the period r independent of x_0.



1.4.3 The Quantum Fourier Transform

In 1994, Peter Shor thought about using an "oracle query" approach to find the period of a function

$$f(x + r) = f(x) = b^x \mod N. \tag{1.143}$$



$$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle_n \, |f_0\rangle_{n_0} \, . \tag{1.144}$$

The quantum state, after measurement of the output register, is a state which is periodic in r. Goal: Explain the approach needed to solve the problem!

1.4.4 Fourier Transforms

Fourier transforms are transformation in the *representation* of a function.

Continuous Fourier Transforms

As originally conceived the Fourier transform allows the representation of a function as *either* a function in time or a function of frequency.

$$\begin{aligned} H(f) &= \int_{\mathbb{R}} h(t) e^{2\pi i t f} \, dt \\ h(t) &= \int_{\mathbb{R}} H(f) e^{-2\pi i t f} \, df. \end{aligned} \tag{1.145}$$

In this definition, both $h(t)$ and $H(f)$ are both complex functions. If $h(t)$ is real then

$$H(-f) = H^*(f). \tag{1.146}$$

An important feature of the Fourier transform is "time shifting" or "frequency shifting"

$$\begin{aligned} h(t - t_0) &\iff H(f) e^{2\pi i f t_0} \\ h(t) e^{-2\pi i f_0 t} &\iff H(f - f_0). \end{aligned} \tag{1.147}$$

In physics we often consider the same state vector as a function of position or of momentum. The state vector is $|\psi\rangle$. It is represented in the position basis as $\langle x|\psi\rangle = \psi(x)$, and in momentum basis as $\langle p|\psi\rangle = \phi(p)$.

$$\begin{aligned} \phi(p) = \langle p|\psi\rangle &= \frac{1}{\sqrt{2\hbar\pi}} \int_{\mathbb{R}} \langle x|\psi\rangle \, e^{-ipx/\hbar} \, dx \\ \psi(x) = \langle x|\phi\rangle &= \frac{1}{\sqrt{2\hbar\pi}} \int_{\mathbb{R}} \langle p|\psi\rangle \, e^{+ipx/\hbar} \, dp. \end{aligned} \tag{1.148}$$

This version of the equation is explicit in pointing out that the Fourier transform is simply a (unitary) basis transformation of the same state.

Discrete Fourier Transform

Often data is discretized into a time-series

$$h(t_k) = h(k\Delta), \quad k = 0, 1, 2, \ldots, N-1. \tag{1.149}$$

where Δ is the time interval between samples (called the "sampling rate"). The *sampling theorem* says that if a continuous function contains no frequencies greater than the *Nyquist* frequency

$$f_c \equiv \frac{1}{2\Delta} \tag{1.150}$$

then the continuous function is fully defined by samples taken at interval Δ. The Fourier transform of a discrete finite set of data is defined at a set of frequencies

$$f_j = \frac{j}{N\Delta}. \tag{1.151}$$

For discretized data we can approximate the Fourier integral by a discrete sum.

$$\begin{aligned} H_j &= \sum_{k=0}^{N-1} h_k e^{2\pi i j k/N} \\ h_k &= \frac{1}{N} \sum_{j=0}^{N-1} H_j e^{-2\pi i j k/N}. \end{aligned} \tag{1.152}$$

With this definition the discrete Fourier transform does not depend on the time scale Δ or the frequency scale $1/(2\Delta)$.

An equally valid way of writing the DFT is:

$$\begin{aligned} H_j &= \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} h_k \left(e^{2\pi i/N}\right)^{kj} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} h_k \omega^{kj} \\ h_k &= \frac{1}{\sqrt{N}} \sum_{j=0}^{N-1} H_j \left(e^{2\pi i/N}\right)^{jk} = \frac{1}{\sqrt{N}} \sum_{k=0}^{N-1} H_j \omega*^{kj}. \end{aligned} \tag{1.153}$$

where $\omega = e^{2\pi i/N}$.

To put this in the context of quantum mechanics it's useful to think of h and

H as vectors. The transformation between the two vectors is a *unitary* matrix:

$$\vec{H} = \frac{1}{\sqrt{N}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{N-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(N-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \omega^{N-2} & \omega^{2(N-2)} & \dots & \omega^{(N-2)(N-1)} \\ 1 & \omega^{N-1} & \omega^{2(N-1)} & \dots & \omega^{(N-1)^2} \end{pmatrix} \vec{h} \tag{1.154}$$

Notice that this is an $N \times N$ matrix multiply! In this form the DFT requires N^2 multiplications. For an n-bit number N, this is $(2^n)^2$, which is exponential in n.

The Fast Fourier Transform (FFT)

An algorithm for a "fast" way to evaluate the DFT was made widely known in the mid-1960s by IBM researchers James Cooley and John Tukey. In an FFT the DFT can be evaluated in $N \log N$ steps.

The approach is a recursive application of splitting the problem into two parts:

$$\begin{aligned} H_j &= \sum_{k=0}^{N/2-1} h_{2k} (\omega)^{(2k)j} + \sum_{k=0}^{N/2-1} h_{2k+1} (\omega)^{(2k+1)j} \\ &= \sum_{k=0}^{N/2-1} h_{2k} (\omega^2)^k + \omega^k \sum_{k=0}^{N/2-1} h_{2k+1} (\omega^2)^{kj} \end{aligned} \tag{1.155}$$

A DFT using this equation would take $2(N/2)^2$ operations. Using the approach recursively until each individual transform is of length 1 requires $\log_2 N$ divisions. Doing N transformations gives a total operation count of $\mathcal{O}(N \log_2 N)$.

While a huge advantage in practice for computing DFTs, $N \log N$ is STILL exponential in the number of bits, requiring operations.

The Quantum Fourier Transform (QFT)

The n-Qbit Quantum Fourier Transform is a unitary basis transformation defined in exactly the same way as the DFT. It acts on the basis states $|x\rangle$.

$$\mathbf{U}_{FT} |x\rangle = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} \left(e^{2\pi i / 2^n} \right)^{xy} |y\rangle = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} \omega^{xy} |y\rangle \tag{1.156}$$

where x and y are n-bit integers, and xy represents ordinary integer multiplication. Since the operation is linear, it acts on a general superposition of states

$$|\psi\rangle = \sum_{0\leq x<2^n} \gamma(x)\,|x\rangle \tag{1.157}$$

to give

$$\mathbf{U}_{FT}\,|\psi\rangle = \sum_{0\leq x<2^n} |(\rangle\, x)\mathbf{U}_{FT}\,|x\rangle = \sum_{0\leq x<2^n} \gamma(x)\frac{1}{\sqrt{2^n}} \sum_{0\leq y<2^n} \omega^{xy}\,|y\rangle \tag{1.158}$$

If we reverse the order of the summations:

$$\mathbf{U}_{FT}\,|\psi\rangle = \sum_{0\leq y<2^n} \underbrace{\frac{1}{\sqrt{2^n}} \sum_{0\leq x<2^n} \gamma(x)\omega^{xy}}_{\tilde{\gamma}(y)}\,|y\rangle\,. \tag{1.159}$$

Giving:

$$\mathbf{U}_{FT}\,|\psi\rangle = \sum_{0\leq y<2^n} \tilde{\gamma}(y)\,|y\rangle\,. \tag{1.160}$$

The coefficients of the transformed state are just the Fourier transformed coefficients of the original state:

$$\tilde{\gamma}(y) = \frac{1}{\sqrt{2^n}} \sum_{0\leq x<2^n} \gamma(x)\omega^{xy}. \tag{1.161}$$

The n-Qbit Quantum Fourier Transform is a unitary basis transformation between two states, which have $N = 2n$ components:

$$\vec{\tilde{\gamma}} = \frac{1}{\sqrt{2^n}} \begin{pmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{2^n-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(2^n-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \omega^{2^n-2} & \omega^{2(2^n-2)} & \dots & \omega^{(2^n-2)(2^n-1)} \\ 1 & \omega^{2^n-1} & \omega^{2(2^n-1)} & \dots & \omega^{(2^n-1)^2} \end{pmatrix} \vec{\gamma} \tag{1.162}$$

where $\omega = e^{2\pi i/2^n}$. Again, this is an $N \times N$ matrix multiply! In this form the QFT requires N^2 multiplications. For an n-bit number N, this is $(2^n)^2$, which is exponential in n and therefore is not efficient.

1.4.5 Quantum Fourier Transform and Period Finding

There are two things left to show. First, what does a QFT do to a state like

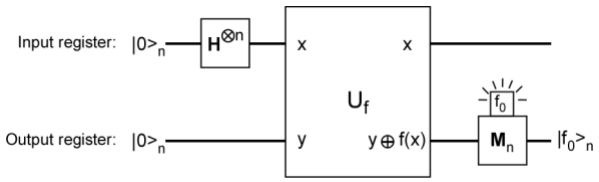
$$|\psi\rangle_n = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle_n \tag{1.163}$$

and how can we use it to determine the period r?

Second, can (really, how can) a QFT be evaluated efficiently (in a number of operations that is polynomial in the number of Qbits not exponential?

Well, using an oracle query approach to find the period of a function:

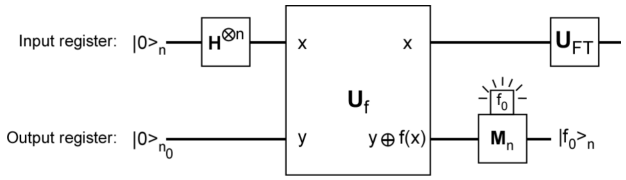
$$f(x+r) = f(x) = b^x \mod N \tag{1.164}$$



$$|\psi_3\rangle = \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} |x_0 + kr\rangle_n \, |f_0\rangle_{n_0} \, . \tag{1.165}$$

After measurement of the output register the input register is a state which is periodic in r. Using the Quantum Fourier Transform on the input register:

$$|\psi_4\rangle = \frac{1}{\sqrt{2^n}} \sum_{y=0}^{2^n-1} \frac{1}{\sqrt{m}} \sum_{k=0}^{m-1} e^{2\pi i (x_0+kr)y/2^n} \, |y\rangle_n \, |f_0\rangle_{n_0} \, . \tag{1.166}$$



Our first goal is to show how to efficiently evaluate the QFT. Given the definition of the QFT:

$$\mathbf{U}_{FT} \, |x\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} \left(e^{2\pi i/2^n} \right)^{xy} |y\rangle_n \, , \tag{1.167}$$

we can think about how to break this up into operators on each Qbit by writing the states explicitly as

$$\begin{aligned} |x\rangle_n &= |x_{n-1}\rangle \, |x_{n-2}\rangle \ldots |x_0\rangle \\ |y\rangle_n &= |y_{n-1}\rangle \, |y_{n-2}\rangle \ldots |y_0\rangle \end{aligned} \tag{1.168}$$

and thinking about the sum as a sequence nested sums over each binary digit.

$$\mathbf{U}_{FT}\left|x\right\rangle_n = \frac{1}{\sqrt{2}}\sum_{0\leq y_{n-1}<2}\frac{1}{\sqrt{2}}\sum_{0\leq y_{n-2}<2}\cdots\frac{1}{\sqrt{2}}\sum_{0\leq y_1<2}\frac{1}{\sqrt{2}}\sum_{0\leq y_0<2}\left(e^{2\pi i/2^n}\right)^{xy}\left|y\right\rangle_n \tag{1.169}$$

and write the product out as a function of those binary digits, too:

$$xy = (2^{n-1}x_{n-1} + 2^{n-2}x_{n-2} + \cdots + 2x_1 + x_0)\times(2^{n-1}y_{n-1} + 2^{n-2}y_{n-2} + \cdots + 2y_1 + y_0). \tag{1.170}$$

Example 1.4.1. Trivially, for a 1-Qbit state, the $\mathbf{U}_{FT}$ is just the Hadamard.

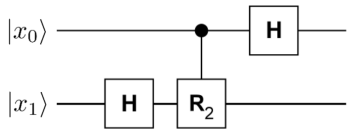
Example 1.4.2. For a 2-Qbit state:

$$xy = (2x_1 + x_0)2y_1 + (2x_1 + x_0)y_0 \tag{1.171}$$

$$\begin{aligned}\mathbf{U}_{FT}\left|x_1\right\rangle\left|x_0\right\rangle &= \frac{1}{\sqrt{2}}\sum_{0\leq y_1<2}\left(e^{2\pi i}\right)^{x_1y_1}\left(e^{i\pi}\right)^{x_0y_1}\left|y_1\right\rangle\times\frac{1}{\sqrt{2}}\sum_{0\leq y_0<2}\left(e^{2\pi i}\right)^{x_0y_0}\left(e^{i\pi/2}\right)^{y_0x_1}\left|y_0\right\rangle \\ &= \frac{1}{\sqrt{2}}\sum_{0\leq y_1<2}(-1)^{x_0y_1}\left|y_1\right\rangle\times\frac{1}{\sqrt{2}}\sum_{0\leq y_0<2}(-1)^{y_0x_1}\left(e^{i\pi/2}\right)^{x_0y_0}\left|y_0\right\rangle.\end{aligned} \tag{1.172}$$

The first sum looks like a Hadamard of $\left|x_0\right\rangle$ but in the $\left|y_1\right\rangle$ Qbit. The second sum looks like a Hadamard of $\left|x_1\right\rangle$ but in the $\left|y_0\right\rangle$ Qbit and with an additional $\pi/2$ phase shift controlled by $\left|x_0\right\rangle$.

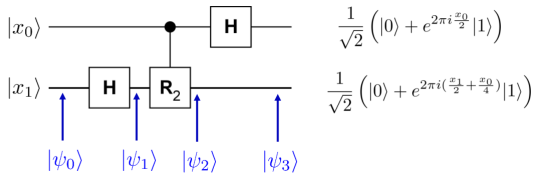
Model implementation: with



$$\boldsymbol{R}_d = \tilde{\boldsymbol{n}} + e^{2\pi i/2^d}\boldsymbol{n} = \begin{pmatrix}1 & 0\\ 0 & e^{2\pi i/2^d}\end{pmatrix} \tag{1.173}$$

$$((\boldsymbol{cR}))_{d)_{01}}\left|x_1\right\rangle\left|x_0\right\rangle = \left(e^{2\pi i/2^d}\right)^{x_0x_1}\left|x_1\right\rangle\left|x_0\right\rangle \tag{1.174}$$

Let's follow the state through the circuit:

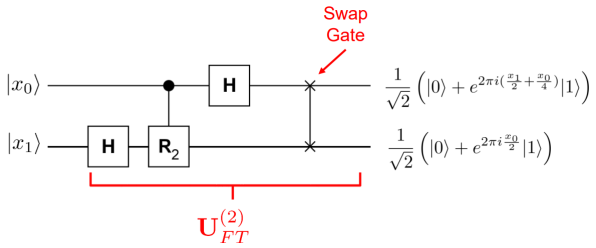


$$|\psi_0\rangle = |x_1\rangle|x_0\rangle$$

$$|\psi_1\rangle = \mathbf{H}_1|\psi_0\rangle = \frac{1}{\sqrt{2}}\sum_{0\le z_1<2}\left(e^{i\pi}\right)^{x_1 z_1}|z_1\rangle|x_0\rangle$$

$$|\psi_2\rangle = (\mathbf{cR}_2)_{01}|\psi_1\rangle = \frac{1}{\sqrt{2}}\sum_{0\le z_1<2}\left(e^{i\pi}\right)^{x_1 z_1}\left(e^{i\pi/2}\right)^{x_0 z_1}|z_1\rangle|x_0\rangle$$

$$|\psi_3\rangle = \mathbf{H}_0|\psi_2\rangle = \frac{1}{\sqrt{2}}\sum_{0\le z_1<2}\left(e^{i\pi}\right)^{x_1 z_1}\left(e^{i\pi/2}\right)^{x_0 z_1}|z_1\rangle\frac{1}{\sqrt{2}}\sum_{0\le z_0<2}\left(e^{i\pi}\right)^{x_0 z_0}|z_0\rangle$$
$$= \frac{1}{\sqrt{2}}\sum_{0\le z_1<2} e^{2\pi i(\frac{x_1}{2}+\frac{x_0}{4})z_1}|z_1\rangle\frac{1}{\sqrt{2}}\sum_{0\le z_0<2} e^{2\pi i(\frac{x_0}{2})z_0}|z_0\rangle$$

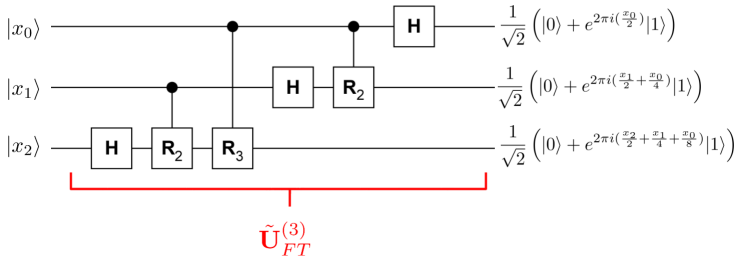


$$|\psi_3\rangle = \frac{1}{\sqrt{2}}\sum_{0\le z_1<2} e^{2\pi i z_1(\frac{x_1}{2}+\frac{x_0}{4})}|z_1\rangle\frac{1}{\sqrt{2}}\sum_{0\le z_0<2} e^{2\pi i z_0(\frac{x_0}{2})}|z_0\rangle$$

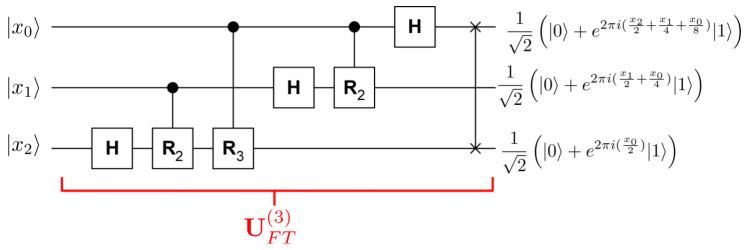
$$|\psi_4\rangle = \mathbf{S}|\psi_3\rangle = \frac{1}{\sqrt{2}}\sum_{0\le z_1<2} e^{2\pi i z_1(\frac{x_0}{2})}|z_1\rangle\frac{1}{\sqrt{2}}\sum_{0\le z_0<2} e^{2\pi i z_0(\frac{x_1}{2}+\frac{x_0}{4})}|z_0\rangle$$

Example 1.4.3. For 3-Qbit state:

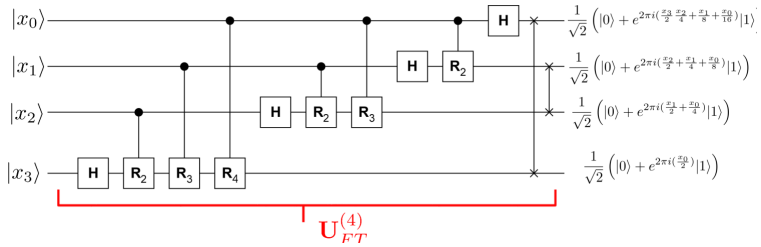
$$\mathbf{U}_{FT}^{(3)}|x_2\rangle|x_1\rangle|x_0\rangle = \frac{1}{\sqrt{2}}\sum_{0\le y_2<2} e^{2\pi i(\frac{x_0}{2})y_2}|y_2\rangle$$
$$\times\frac{1}{\sqrt{2}}\sum_{0\le y_1<2} e^{2\pi i(\frac{x_1}{2}+\frac{x_0}{4})y_1}|y_1\rangle$$
$$\times\frac{1}{\sqrt{2}}\sum_{0\le y_0<2} e^{2\pi i(\frac{x_2}{2}+\frac{x_1}{4}+\frac{x_0}{8})y_0}|y_0\rangle$$



Apply a SWAP gate to correct:



Example 1.4.4. For 4-Qbit gate:



There's a clear pattern, and each additional Qbit k will require $k+1$ gates. The total gate count (not including $[n/2]$ swap gates) is therefore

$$\boxed{G = \sum_{k=0}^{n}(k+1) = \frac{n(n+1)}{2} = \mathcal{O}(n^2)} \tag{1.175}$$

1.5 The Quantum Fourier Transform & Applications

1.5.1 The QFT

1.5.2 Phase estimation

Performance and requirements

1.5.3 Applications: order-finding and factoring

Order-finding

Factoring

1.5.4 General applications of the QFT

Period-finding

Discrete algorithms

The hidden subgroup problem

Other quantum algorithm?

1.6 Quantum search algorithm

1.6.1 The quantum search algorithm

The oracle

The procedure

Geometric visualization

Performance

1.6.2 Quantum search as a quantum simulation

1.6.3 Quantum counting

1.6.4 Speeding up the solution of NP-complete problems

1.6.5 Quantum search of an unstructured database

1.6.6 Optimality of the search algorithm

1.6.7 Black box algorithm limits

1.7 Searching with a quantum computer

1.7.1 The nature of the search

1.7.2 The Grover iteration

1.7.3 How to construct W

1.7.4 Generalization to several special numbers

1.7.5 Searching for one out of four items

1.8 Quantum error correction

1.8.1 The miracle of quantum error correction

1.8.2 A simplified example

1.8.3 The physics of error generation

1.8.4 Diagnosing error syndromes

1.8.5 The 5-Qbit error-correcting code

1.8.6 The 7-Qbit error-correcting code

1.8.7 Operations on 7-Qbit codewords

1.8.8 A 7-Qbit encoding circuit

1.8.9 A 5-Qbit encoding circuit

Part 2

Problems

2.1 Problem Set 1

1. Computational complexity. As mentioned in class, improved algorithms can vastly reduce the number of required gates (or steps) needed to perform a calculation. In class I used the example of the bubble sort versus the heap sort and the straightforward discrete Fourier transform versus the fast Fourier transform. In both of those examples the difference in complexity of the algorithm was N^2 vs. $N \log N$, where for those problems N was the number of elements that be sorted or the number of points in a time-series. With the algorithmic change of scaling with N the difference in number of operations becomes huge as N gets large, but both N^2 and $N \log N$ are polynomial in N and the solution is considered *efficiently computable* with either algorithm.

To think more about what is meant by not being efficiently computable, you will now consider the goal of finding prime factors of a (large) composite number. The basic idea of finding prime factors is that given an integer n-bit integer N find a factorization

$$N = pq \tag{2.1}$$

where at least one of the integers p and q are prime.

(a) The most straightforward algorithm for finding prime factors is called "trial division" or "direct-search factorization" and it is just what it sounds like. Start testing all integers up to $p = \sqrt{N}$ as possible factors. A number N is represented by a binary number with $n = \lg(N)$ bits. Show that this leads to an exponential scaling in the number of bits $\mathcal{O}(2^{n/2})$, which is exponential in the number of bits. This is still the most efficient algorithm for relatively small numbers.

(b) As mentioned in class the most efficiently known classical factoring algorithm for large numbers is the "number field sieve" and its complexity is $\mathcal{O}\left(e^{cn^{1/3}(\ln n)^{2/3}}\right)$, where $c = (64/9)^{1/3}$, which is also exponential in the number of bits. Shors quantum algorithm is $\mathcal{O}(n^3)$ which is polynomial in n, and thus is efficient. Create a table showing the difference in the number of steps (time) needed to factor an n-bit number using these four methods for $n = 4$, 16, 32, 64, 128, 256, 512, and 1024. The RSA challenge for RSA-768 a 768 bit (232 decimal digit) number was factored in 2009 using the number field sieve after several years using "many hundreds of machines." A similar effort factored RSA-240 (795 bits) in 2019

Solution:

(a) Let a number N be given. N is represented by a binary number with $n = \lg(N)$ bits. The trial division algorithm tests integers up to $p = \sqrt{N}$, which can be represented by a binary number with $\tilde{n} = \lg(\sqrt{N}) = n/2$. It follows that the number of test integers (from 1 to p) is on the order

of $2^{n/2}$. This means there is an exponential scaling in the number of bits $\mathcal{O}\left(2^{n/2}\right)$.

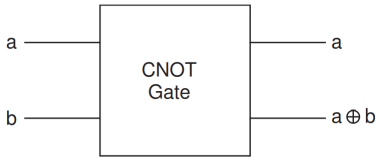
(b)

n	$\exp\left\{\sqrt[3]{64/19}\sqrt[3]{n}(\ln n)^{2/3}\right\}$	n^3
4	19.3	64
16	1730	4096
32	5.41×10^4	3.28×10^4
64	5.43×10^6	2.62×10^5
128	2.53×10^9	2.10×10^6
256	8.92×10^{12}	1.68×10^7
512	4.46×10^{17}	1.34×10^8
1024	7.16×10^{23}	1.07×10^9

□

2. Reversible Computation. Quantum computation is reversible. That is, no matter how sophisticated the computation is, you can always run the program in reverse and figure out what was input to start the computation. In 1973 Charles Bennett (IBM) showed that you could perform any classical computation using reversible gates, and reversible computation has been a part of theoretical computer-science since then. The NOT gate is reversible, but the standard two-CBit gates, AND, OR, and XOR, are clearly not reversible because they have two inputs and only one output so you cant recreate their inputs knowing their outputs.

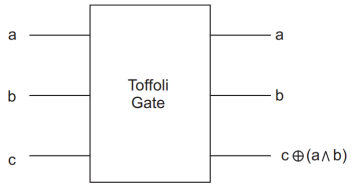
1. *Controlled-not (CNOT).* While the standard exclusive-or (XOR, $\oplus$) gate of Boolean logic is not reversible, there is a reversible equivalent to the XOR gate called the controlled-not (CNOT) gate. As shown below in both the circuit diagram and the truth table it has two inputs and two outputs. Show by creating the truth table that the CNOT gate is reversible - that



a	b	a	$a \oplus b$
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

   you can find the inputs if you know the outputs. In fact, you can show that CNOT is its own inverse. To do this you will "prove" the identity $a \oplus (a \oplus b) = b$.

2. *Toffoli Gate.* While the CNOT gate is reversible, it is not universal. That is, you cannot implement every possible logic operation with the CNOT. The NAND gate is actually *universal*: you can create any logical operation using just NAND gates (but of course it isnt reversible). In 1980 Tommaso Toffoli (MIT) described a universal reversible gate now called the Toffoli gate or sometimes the "controlled-controlled-not." [Theres another universal reversible gate called the Fredkin gate (or "controlled-swap") named after its inventor Edward Fredkin.] The Toffoli gate has three inputs and three outputs and its behavior is shown in the circuit diagram shown below. The outputs are a copy of both inputs and the logical expression $c \oplus (a \wedge b)$. There are 8 possible input combinations to the Toffoli



gate. Create a truth table for the Toffoli gate and show that for $c = 0$ that the output is $a \wedge b$ and for $c = 1$ that the output is $a \wedge b$ where $\wedge$ is the symbol for AND. That is, show the Toffoli gate can generate the same logical output as the AND or NAND gates. Show that the Toffoli gate is its own inverse by creating the truth table using the three outputs you calculated as inputs to a Toffoli gate. Why would the Toffoli gate be called the controlled-controlled-not (CCNOT)?

Solution:

(a) CNOT:

a	b	a	$a \oplus b$	$a \oplus (a \oplus b)$
0	0	0	0	0
0	1	0	1	1
1	0	1	1	0
1	1	1	0	1

We have just shown (by exhaustion) that $a \oplus (a \oplus b) = b$, which means we can invert the CNOT gate to obtain its input (a, b) for any given output $(a, a \oplus b)$.

(b) Toffoli:

a	b	c	$(a \wedge b)$	a	b	$c \oplus (a \wedge b)$
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	1	0
0	1	1	0	0	1	1
1	0	0	0	1	0	0
1	0	1	0	1	0	1
1	1	0	1	1	1	1
1	1	1	1	1	1	0

We see that $\text{Toff}[a, b, 0] = a \wedge b$, and $\text{Toff}[a, b, 1] = \overline{a \wedge b}$. So, by setting c to be 0 or 1, we can make the Toffoli gate an AND or a NAND gate.

To show that the Toffoli gate is its own inverse, we show $c = (a \wedge b) \oplus c \oplus (a \wedge b)$, once again by exhaustion:

a	b	c	$(a \wedge b)$	a	b	$c \oplus (a \wedge b)$	$(a \wedge b) \oplus c \oplus (a \wedge b)$
0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	1
0	1	0	0	0	1	0	0
0	1	1	0	0	1	1	1
1	0	0	0	1	0	0	0
1	0	1	0	1	0	1	1
1	1	0	1	1	1	1	0
1	1	1	1	1	1	0	1

The Toffoli gate would be called the controlled-controlled-not because unlike the CNOT where there is one control bit and one target bit, the Toffoli gate has **two** control bits, namely a and b, and one target bit, namely c. The Toffoli gate flips c if and only if $a = b = 1$.

□

3. In Section 1.4, Mermin defines several 1-Cbit operations that do not correspond to any physical operation but which can be useful in deriving relationships between operations that do have a physical meaning. The first two of these operators are the "projection operators."

$$\mathbf{n} \xrightarrow[\text{computational basis}]{} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \qquad \tilde{\mathbf{n}} \xrightarrow[\text{computational basis}]{} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}.$$

(a) Show, using their matrix form, that as Mermin states on the top of page 12, the two matrices have eigenvalues of 0 and +1. Determine eigenvectors of each matrix assuming that they are in the form of probabilistic CBits

$$|a\rangle = \begin{pmatrix} a_0 \\ a_1 \end{pmatrix} \tag{2.2}$$

where the convention of probability requires $a_0 + a_1 = 1$.

(b) Show, using the definitions, that the properties of Eq. 1.33 and 1.34 are correct.

Solution:

(a) Because $\mathbf{n}$ and $\tilde{\mathbf{n}}$ are diagonal matrices with only entries 0 and 1 along the diagonals, their eigenvalues are 0 and 1.

The (stochastic) 1-eigenvector of $\mathbf{n}$, say $|a\rangle_1 = (a_0\ a_1)^\top$ where $a_0 + a_1 = 1$, must be $|a\rangle_1 = (1\ 0)^\top$ because $\mathbf{n}(a_0\ a_1)^\top = (a_0\ 0)^\top$. By a similar argument, the (stochastic) 0-eigenvector of $\mathbf{n}$ must be $|a\rangle_0 = (0\ 1)^\top$. By symmetry, the 1-eigenvector of $\tilde{\mathbf{n}}$ is $|\tilde{a}\rangle_1 = (0\ 1)^\top$, and the 0-eigenvector of $\tilde{\mathbf{n}}$ is $|\tilde{a}\rangle_0 = (1\ 0)^\top$.

(b) We note that projections are *idempotents*, so $\mathbf{n}^2 = \mathbf{n}$ and $\tilde{\mathbf{n}}^2 = \tilde{\mathbf{n}}$ automatically. Next, because the images of $\mathbf{n}$ and $\tilde{\mathbf{n}}$ are orthogonal spaces (spanned by $(1\ 0)^\top$ and $(0\ 1)^\top$, respectively), $\mathbf{n}\tilde{\mathbf{n}} = \tilde{\mathbf{n}}\mathbf{n} = \mathbf{0}_{2\times 2}$. Finally, $\mathbf{n} + \tilde{\mathbf{n}} = \mathbb{I}$ because the idempotents $\mathbf{n}$ and $\tilde{\mathbf{n}}$ *resolve identity* $\mathbb{I}$.

We can also verify these properties algebraically:

$$\mathbf{n}^2 = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \tag{2.3}$$

$$\tilde{\mathbf{n}}^2 = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \tag{2.4}$$

$$\mathbf{n}\tilde{\mathbf{n}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix}\begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix}\begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \tilde{\mathbf{n}}\mathbf{n} \tag{2.5}$$

$$\mathbf{n} + \tilde{\mathbf{n}} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} = \mathbb{I} \tag{2.6}$$

Next we consider the $\mathbf{X} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$, the bit-flip:

$$\mathbf{nX} = \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 0 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} = \mathbf{X}\tilde{\mathbf{n}} \tag{2.7}$$

$$\tilde{\mathbf{n}}\mathbf{X} = \begin{pmatrix} 0 & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 0 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \end{pmatrix} = \mathbf{Xn}. \tag{2.8}$$

Mermin explained why this makes sense, so I won't repeat that.

□

4. Another pair of operators which do not have physical meaning when acting on CBits are the "phase-flip operator" $\mathbf{Z}$ and Hadamard operator $\mathbf{H}$. where

$$\mathbf{Z} = \mathbf{n} - \tilde{\mathbf{n}} \xrightarrow[\text{basis}]{\text{computational}} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \qquad \mathbf{H} = \frac{1}{\sqrt{2}}(\mathbf{X} - \mathbf{Z}) \xrightarrow[\text{basis}]{\text{computational}} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$$

$\mathbf{X}$ is the "bit-flip" (NOT) operator. Both $\mathbf{Z}$ and $\mathbf{H}$ will be used and do have meaning when applied to QBits.

(a) Show, using their matrix forms, that the three single-bit gates $\mathbf{X}$, $\mathbf{Z}$, and $\mathbf{H}$ are their own inverse. That is, show that $\mathbf{X}^2 = \mathbb{I}$, $\mathbf{Z}^2 = \mathbb{I}$, and $\mathbf{H}^2 = \mathbb{I}$.

(b) Show, using their matrix forms, that the identities of Eq. 1.43 are correct. That is, show that using a Hadamard gate to perform a "similarity transformation" you can convert a bit-flip to a phase-flip and vice-versa.

Solution:

(a)

$$\mathbf{Z}^2 = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{2.9}$$

$$\mathbf{X}^2 = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{2.10}$$

$$\mathbf{H}^2 = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}. \tag{2.11}$$

(b)

$$\mathbf{HXH} = \frac{1}{2} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} = \mathbf{Z} \tag{2.12}$$

$$\mathbf{HXH} = \mathbf{Z} \implies \mathbf{X} = \mathbf{H}^{-1}\mathbf{Z}\mathbf{H}^{-1} \implies \mathbf{X} = \mathbf{HZH} \quad (\mathbf{H}^{-1} = \mathbf{H}) \tag{2.13}$$

□

2.2 Problem set 2

1. Compute, using the matrix representations of the operators and probability-vector states, the result of applying a Hadamard transofmr to both bits of a two-Cbit state $|ba\rangle = |b\rangle\,|a\rangle = |b\rangle \otimes |a\rangle$ first by creating the tensor product of the two vectors $\mathbf{H}\,|b\rangle \otimes \mathbf{H}\,|a\rangle$ and second by creating the tensor product of the Hadamard operators $\mathbf{H}_1\mathbf{H}_0 = \mathbf{H} \otimes \mathbf{H}$ and applying it to the state $|b\rangle\,|a\rangle$.

Solution: The tensor product works the way we want it to work:

$$\begin{aligned}
\mathbf{H}\,|b\rangle \otimes \mathbf{H}\,|a\rangle &= \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}b_0\\ b_1\end{pmatrix} \otimes \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\begin{pmatrix}a_0\\ a_1\end{pmatrix}\\
&= \frac{1}{\sqrt{2}}\begin{pmatrix}b_0+b_1\\ b_0-b_1\end{pmatrix} \otimes \frac{1}{\sqrt{2}}\begin{pmatrix}a_0+a_1\\ a_0-a_1\end{pmatrix}\\
&= \frac{1}{\sqrt{2}}\left[(b_0+b_1)\,|0\rangle + (b_0-b_1)\,|1\rangle\right] \otimes \frac{1}{\sqrt{2}}\left[(a_0+a_1)\,|0\rangle + (a_0-a_1)\,|1\rangle\right]\\
&= \frac{1}{2}\left[(b_0+b_1)(a_0+a_1)\,|00\rangle + (b_0-b_1)(a_0+a_1)\,|10\rangle\right.\\
&\quad \left.+(b_0+b_1)(a_0-a_1)\,|01\rangle + (b_0-b_1)(a_0-a_1)\,|11\rangle\right]. \qquad (2.14)
\end{aligned}$$

$$\begin{aligned}
(\mathbf{H}\otimes\mathbf{H})\,|ba\rangle &= \frac{1}{\sqrt{2}}\begin{pmatrix}\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} & \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\\ \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix} & -\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}\end{pmatrix}\begin{pmatrix}b_0\begin{pmatrix}a_0\\ a_1\end{pmatrix}\\ b_1\begin{pmatrix}a_0\\ a_1\end{pmatrix}\end{pmatrix}\\
&= \frac{1}{2}\begin{pmatrix}1 & 1 & 1 & 1\\ 1 & -1 & 1 & -1\\ 1 & 1 & -1 & -1\\ 1 & -1 & -1 & 1\end{pmatrix}\begin{pmatrix}b_0a_0\\ b_0a_1\\ b_1a_0\\ b_1a_1\end{pmatrix}\\
&= \ldots \text{(multiply in and factor)}\\
&= \frac{1}{2}\left[(b_0+b_1)(a_0+a_1)\,|00\rangle + (b_0-b_1)(a_0+a_1)\,|10\rangle\right.\\
&\quad \left.+(b_0+b_1)(a_0-a_1)\,|01\rangle + (b_0-b_1)(a_0-a_1)\,|11\rangle\right]. \qquad (2.15)
\end{aligned}$$

From Eq. (2.14) and (2.15),

$$\boxed{\mathbf{H}\,|b\rangle \otimes \mathbf{H}\,|a\rangle = (\mathbf{H}\otimes\mathbf{H})\,|ba\rangle} \qquad (2.16)$$

□

2. Writing out the matrix representations, show that Mermin's identity $\mathbf{S}_{ij} = \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij}$ is true for $\mathbf{S}_{ij}$.

Solution: Let $i = 0, j = 1$, then

$$\begin{aligned}
\mathbf{C}_{01}\mathbf{C}_{10}\mathbf{C}_{01} &= \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & & & 1 \\ & 1 & & \\ & & 1 & \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{pmatrix} \\
&= \mathbf{S}_{10} = \mathbf{S}_{01}. \qquad (2.17)
\end{aligned}$$

Let $i = 1, j = 0$, then

$$\begin{aligned}
\mathbf{C}_{10}\mathbf{C}_{01}\mathbf{C}_{01} &= \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix} \begin{pmatrix} 1 & & & \\ & & & 1 \\ & 1 & & \\ & & 1 & \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & & 1 & \\ & 1 & & \\ & & & 1 \end{pmatrix} \\
&= \mathbf{S}_{10} = \mathbf{S}_{01}. \qquad (2.18)
\end{aligned}$$

□

3. Show, using matrix operations, that surrounding a CNOT with two Hadamard gates switches the role of the control-bit and target bit. Specifically, using the matrices show that $\mathbf{C}_{10} = (\mathbf{H}_0\mathbf{H}_1)\mathbf{C}_{01}(\mathbf{H}_0\mathbf{H}_1)$, or formally that

$$\mathbf{C}_{10} = (\mathbf{H} \otimes \mathbf{H})\mathbf{C}_{01}(\mathbf{H} \otimes \mathbf{H}). \tag{2.19}$$

This will demonstrate a special case of

$$\mathbf{C}_{ji} = (\mathbf{H}_j\mathbf{H}_i)\mathbf{C}_{ij}(\mathbf{H}_i\mathbf{H}_j). \tag{2.20}$$

Solution:

$$\begin{aligned}
(\mathbf{H} \otimes \mathbf{H})\mathbf{C}_{01}(\mathbf{H} \otimes \mathbf{H}) &= \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \\
&= \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix} \\
&= \mathbf{C}_{10}
\end{aligned} \tag{2.21}$$

Conversely,

$$\begin{aligned}
(\mathbf{H} \otimes \mathbf{H})\mathbf{C}_{10}(\mathbf{H} \otimes \mathbf{H}) &= \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 & & & \\ & 1 & & \\ & & & 1 \\ & & 1 & \end{pmatrix}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix} \\
&= \frac{1}{4}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}\begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & -1 & -1 & 1 \\ 1 & 1 & -1 & -1 \end{pmatrix} \\
&= \begin{pmatrix} 1 & & & \\ & & & 1 \\ & & 1 & \\ & 1 & & \end{pmatrix} \\
&= \mathbf{C}_{01}
\end{aligned} \tag{2.22}$$

4. Following the examples in class of determining the matrices for the "conditional operations" $\mathbf{S}_{01} = \mathbf{S}_{10}, \mathbf{C}_{01}, \mathbf{C}_{10}, \mathbf{T}_{210}$ determine the 8×8 matrix that represents the Toffoli gate $\mathbf{T}_{012}$, which uses Cbits 0 and 1 as the control bits and Cbits 2 as the target bit.

$$\mathbf{T}_{012} |c\rangle |b\rangle |a\rangle = |c \oplus (a \wedge b)\rangle |b\rangle |a\rangle . \tag{2.23}$$

Solution: Truth table for $\mathbf{T}_{012}$:

c	b	a	$(a \wedge b)$	$c \oplus (a \wedge b)$	b	a
0	0	0	0	0	0	0
0	0	1	0	0	0	1
0	1	0	0	0	1	0
0	1	1	1	1	1	1
1	0	0	0	1	0	0
1	0	1	0	1	0	1
1	1	0	0	1	1	0
1	1	1	1	0	1	1

We see that $\mathbf{T}_{012}$ acts as identity for all states except for some cases:

$$\mathbf{T}_{012} |011\rangle = |111\rangle \tag{2.24}$$
$$\mathbf{T}_{012} |111\rangle = |011\rangle , \tag{2.25}$$

as expected. These correspond to two off-diagonal indices in the matrix for $\mathbf{T}_{012}$. Based on the $\mathbf{T}_{210}$ matrix:

$$\begin{array}{cccccccc} |000\rangle & |001\rangle & |010\rangle & |011\rangle & |100\rangle & |101\rangle & |110\rangle & |111\rangle & \\ \left(\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{array}\right) & \begin{array}{c} |000\rangle \\ |001\rangle \\ |010\rangle \\ |011\rangle \\ |100\rangle \\ |101\rangle \\ |110\rangle \\ |111\rangle \end{array} \end{array} \tag{2.26}$$

By analogy, the matrix for $\mathbf{T}_{012}$ is:

$$\begin{array}{cccccccc} |000\rangle & |001\rangle & |010\rangle & |011\rangle & |100\rangle & |101\rangle & |110\rangle & |111\rangle & \\ \left(\begin{array}{cccccccc} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{array}\right) & \begin{array}{c} |000\rangle \\ |001\rangle \\ |010\rangle \\ |011\rangle \\ |100\rangle \\ |101\rangle \\ |110\rangle \\ |111\rangle \end{array} \end{array} \tag{2.27}$$

2.3 Problem set 3

1. With

$$\mathbf{S}_{ij} = \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j) \tag{2.28}$$

and the identities

$$\begin{aligned} \mathbf{n}^2 = \mathbf{n}, \quad \tilde{\mathbf{n}}^2 = \tilde{\mathbf{n}}, \quad \mathbf{n}\tilde{\mathbf{n}} = \tilde{\mathbf{n}}\mathbf{n} = \mathbf{0}, \quad \tilde{\mathbf{n}} + \mathbf{n} = \mathbb{I}, \\ \mathbf{X}^2 = \mathbb{I}, \quad \mathbf{n}\mathbf{X} = \mathbf{X}\tilde{\mathbf{n}}, \quad \tilde{\mathbf{n}}\mathbf{X} = \mathbf{X}\mathbf{n} \end{aligned} \tag{2.29}$$

show that

$$\mathbf{S}_{ij}^2 = \mathbb{I}. \tag{2.30}$$

Solution: We will extensively use the fact that $[\mathbf{P}_i, \mathbf{Q}_j] = 0$. The strategy here is to make the $\mathbf{X}_i$'s and $\mathbf{X}_j$'s combine and let $\tilde{\mathbf{n}}$ and $\mathbf{n}$ annihilate each other as many times as possible:

$$\begin{aligned} \mathbf{S}_{ij}^2 &= [\mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)]^2 \\ &= \mathbf{n}_i\mathbf{n}_j\mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j\tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)\mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j) \\ &\quad + \cancel{\mathbf{n}_i\mathbf{n}_j\tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j} + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j\mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j) + \cancel{\mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)\mathbf{n}_i\mathbf{n}_j} \\ &\quad + \cancel{\tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j\mathbf{n}_i\mathbf{n}_j} + \cancel{\mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)\tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j} + \cancel{\mathbf{n}_i\mathbf{n}_j\mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)} \\ &= \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + (\tilde{\mathbf{n}}_i\mathbf{n}_j + \mathbf{n}_i\tilde{\mathbf{n}}_j)\underbrace{\mathbf{X}_i^2\mathbf{X}_j^2}_{\mathbb{I}_4}(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j) \\ &\quad + \cancel{\mathbf{X}_i\mathbf{X}_j\mathbf{n}_i\mathbf{n}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j)} \\ &= \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{0} + \tilde{\mathbf{n}}_i\mathbf{n}_j + \mathbf{n}_i\tilde{\mathbf{n}}_j + \mathbf{0} \\ &= \mathbf{n}_i(\mathbf{n}_j + \tilde{\mathbf{n}}_j) + \mathbf{n}_j(\mathbf{n}_j + \tilde{\mathbf{n}}_j) \\ &= \mathbf{n}_i\mathbb{I}_j + \tilde{\mathbf{n}}_i\mathbb{I}_j \\ &= \mathbb{I}_i \otimes \mathbb{I}_j \\ &\equiv \mathbb{I}. \end{aligned} \tag{2.31}$$

□

2. With

$$\mathbf{S}_{ij} = \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij} \tag{2.32}$$

and

$$\mathbf{C}_{ij} = \tilde{\mathbf{n}}_i\mathbb{I}_j + \mathbf{n}_i\mathbf{X}_j \tag{2.33}$$

and other identities show that

$$\mathbf{S}_{ij} = \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j) \tag{2.34}$$

Solution: We can create a lot of cancellations using the identities $\mathbf{nX} = \mathbf{X}\tilde{\mathbf{n}}$ and $\tilde{\mathbf{n}}\mathbf{X} = \mathbf{Xn}$. Also, because $\tilde{\mathbf{n}}$ and $\mathbf{n}$ are idempotents, they are the same as their powers (except power 0). With this, we can "merge" and simplify a lot of the summands.

$$\begin{aligned}
\mathbf{S}_{ij} &= \mathbf{C}_{ij}\mathbf{C}_{ji}\mathbf{C}_{ij} \\
&= [\tilde{\mathbf{n}}_i\mathbb{I}_j + \mathbf{n}_i\mathbf{X}_j][\tilde{\mathbf{n}}_j\mathbb{I}_i + \mathbf{n}_j\mathbf{X}_i][\tilde{\mathbf{n}}_i\mathbb{I}_j + \mathbf{n}_i\mathbf{X}_j] \\
&= [\tilde{\mathbf{n}}_i\mathbb{I}_j + \mathbf{n}_i\mathbf{X}_j][\tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{n}_i\mathbf{X}_j\mathbf{n}_j + \mathbf{X}_i\tilde{\mathbf{n}}_i\mathbf{n}_j + \mathbf{X}_i\mathbf{X}_j\mathbf{n}_i\tilde{\mathbf{n}}_j] \\
&= \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j\mathbf{n}_i\tilde{\mathbf{n}}_j + \mathbf{n}_i\mathbf{n}_j + \mathbf{X}_i\mathbf{X}_j\tilde{\mathbf{n}}_i\mathbf{n}_j + \text{cancellations} \\
&= \mathbf{n}_i\mathbf{n}_j + \tilde{\mathbf{n}}_i\tilde{\mathbf{n}}_j + \mathbf{X}_i\mathbf{X}_j(\mathbf{n}_i\tilde{\mathbf{n}}_j + \tilde{\mathbf{n}}_i\mathbf{n}_j).
\end{aligned} \tag{2.35}$$

The cancellations are exactly those where $\tilde{\mathbf{n}}$ and $\mathbf{n}$ annihilate each other. Some are obvious; some are found by using the identities $\mathbf{nX} = \mathbf{X}\tilde{\mathbf{n}}$ and $\tilde{\mathbf{n}}\mathbf{X} = \mathbf{Xn}$ to "bring the $\tilde{\mathbf{n}}$ and $\mathbf{n}$ together for *mutual annihilation*." □

3. With

$$\mathbf{C}_{ij} = \frac{1}{2}\left[(\mathbb{I}_i + \mathbf{Z}_i)\mathbb{I}_j + (\mathbb{I}_i - \mathbf{Z}_i)\mathbf{X}_j\right] \tag{2.36}$$

and

$$\mathbf{HXH} = \mathbf{Z} \quad \mathbf{HZH} = \mathbf{X} \quad \mathbf{H}^2 = \mathbb{I} \tag{2.37}$$

and other identities show that

$$\mathbf{C}_{ji} = (\mathbf{H}_i\mathbf{H}_j)\mathbf{C}_{ij}(\mathbf{H}_i\mathbf{H}_j) \tag{2.38}$$

Solution: It is easier to go from the RHS to the LHS, using the identities above:

$$\begin{aligned}
(\mathbf{H}_i\mathbf{H}_j)\mathbf{C}_{ij}(\mathbf{H}_i\mathbf{H}_j) &= (\mathbf{H}_i\mathbf{H}_j)\left[\frac{1}{2}\left[(\mathbb{I}_i + \mathbf{Z}_i)\mathbb{I}_j + (\mathbb{I}_i - \mathbf{Z}_i)\mathbf{X}_j\right]\right](\mathbf{H}_i\mathbf{H}_j) \\
&= \frac{1}{2}\left[\mathbf{H}_i^2\mathbf{H}_j^2 + \mathbf{X}_i\mathbf{H}_j^2 + \mathbf{H}_i^2\mathbf{Z}_j - (\mathbf{H}_i\mathbf{Z}_i\mathbf{H}_i)(\mathbf{H}_j\mathbf{X}_j\mathbf{H}_j)\right] \\
&= \frac{1}{2}\left[(\mathbb{I}_i + \mathbf{Z}_i)\mathbb{I}_j + \mathbb{I}_j\mathbf{X}_i - \mathbf{X}_i\mathbf{Z}_j\right] \\
&= \frac{1}{2}\left[(\mathbb{I}_j + \mathbf{Z}_j)\mathbb{I}_i + (\mathbb{I}_j - \mathbf{Z}_j)\mathbb{I}_i\right] \\
&\equiv \mathbf{C}_{ji}.
\end{aligned} \tag{2.39}$$

□

4. Suppose you have been sent a set of photons, and you measure the polarization using a PBS. For each of the states, determine the probability for measuring the photon being in states $|0\rangle = |x\rangle$ and $|1\rangle = |y\rangle$.

(a) $\frac{1}{\sqrt{3}}|0\rangle + \sqrt{\frac{2}{3}}|1\rangle$

(b) $\frac{1}{\sqrt{2}}(|+45\rangle + |-45\rangle)$

(c) $\frac{1}{\sqrt{2}}(|+45\rangle - |-45\rangle)$

(d) $\frac{1}{\sqrt{2}}(|R\rangle - |L\rangle)$

(e) $\frac{\sqrt{3}}{2}|R\rangle + \frac{1}{2}|L\rangle$

Solution:

(a) $P(|0\rangle) = 1/3, P(|1\rangle) = 2/3$.

(b) The given state can be rewritten in the $0-1$ basis:

$$\frac{1}{\sqrt{2}}(|+45\rangle + |-45\rangle) \equiv |0\rangle \tag{2.40}$$

so that $P(|0\rangle) = 1, P(|1\rangle) = 0$.

(c) The given state can be rewritten in the $0-1$ basis:

$$\frac{1}{\sqrt{2}}(|+45\rangle - |-45\rangle) = |1\rangle \tag{2.41}$$

so that $P(|0\rangle) = 0, P(|1\rangle) = 1$.

(d) The given state can be rewritten in the $0-1$ basis:

$$\frac{1}{\sqrt{2}}(|R\rangle - |L\rangle) = \frac{1}{2}\begin{pmatrix}0\\2i\end{pmatrix} = i\begin{pmatrix}0\\1\end{pmatrix} = i|1\rangle \tag{2.42}$$

so $P(|0\rangle) = 0, P(|1\rangle) = 1$.

(e) The given state can be rewritten in the $0-1$ basis as

$$\frac{\sqrt{3}}{2}|R\rangle + \frac{1}{2}|L\rangle = \frac{1}{2\sqrt{2}}\begin{pmatrix}\sqrt{3}+1\\ i(\sqrt{3}-1)\end{pmatrix} \tag{2.43}$$

so that $P(|0\rangle) = \left|\frac{\sqrt{3}+1}{2\sqrt{2}}\right|^2 \approx 0.933, P(|1\rangle) = \left|\frac{i(\sqrt{3}-1)}{2\sqrt{2}}\right|^2 \approx 0.067$.

□

5. A rotation operator $\mathbf{R}_\phi$ on photons is one that takes a photon linearly polarized at angle α and converts it into a linearly polarized photon at angle $\alpha + \phi$. A matrix representation of the rotation operator in the computation basis is

$$\mathbf{R}_\phi = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \tag{2.44}$$

(a) Show that $\mathbf{R}_\phi$ is unitary.

(b) Show that $\mathbf{R}_\phi |0\rangle = |\phi\rangle$ and $\mathbf{R}_\phi |1\rangle = |\pi/2 + \phi\rangle$

(c) Show that $\mathbf{R}_\phi |\alpha\rangle = |\alpha + \phi\rangle$, where $|\alpha\rangle = |\epsilon_{\alpha,0}\rangle$

(d) Show that $\mathbf{R}_\phi$ acting on a right-handed photon $\mathbf{R}_\phi |R\rangle$ gives you back a right handed photon with an overall phase shift ϕ.

(e) Show that the sequence of three operators $\mathbf{R}_\phi \mathbf{Z} \mathbf{R}_{-\phi}$ on a photon in the state $|\epsilon\rangle$ is equivalent to the Hadamard gate if $\phi = \pi/8$. To do this you just need to show that $\mathbf{R}_{-\phi}\mathbf{Z}\mathbf{R}_\phi = \mathbf{H}$ when $\phi = \pi/8$.

Solution:

(a) By inspection, $\mathbf{R}_\phi$ is orthogonal, so it is unitary. Explicitly,

$$\mathbf{R}_\phi^\dagger \mathbf{R}_\phi = \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} = \mathbb{I}_{2\times 2}. \tag{2.45}$$

$\mathbf{R}_\phi^\dagger \mathbf{R}_\phi$ is simply a rotation by ϕ, followed by a rotation by $-\phi$, so the composition is the identity function.

(b)

$$\mathbf{R}_\phi |0\rangle = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos\phi \\ \sin\phi \end{pmatrix} = |\phi\rangle \tag{2.46}$$

$$\begin{aligned} \mathbf{R}_\phi |1\rangle &= \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} \\ &= \begin{pmatrix} -\sin\phi \\ \cos\phi \end{pmatrix} = \begin{pmatrix} \cos(\pi/2 + \phi) \\ \sin(\pi/2 + \phi) \end{pmatrix} = |\pi/2 + \phi\rangle . \end{aligned} \tag{2.47}$$

$\mathbf{R}_\phi$ is a rotation by ϕ, so starting at $|0\rangle$ and $|1\rangle \equiv |\pi/2\rangle$ give $|\phi\rangle$ and $|\pi/2 + \phi\rangle$, respectively.

(c)

$$\begin{aligned} \mathbf{R}_\phi |\alpha\rangle &= \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \cos\alpha \\ \sin\alpha \end{pmatrix} \\ &= \begin{pmatrix} \cos\phi\cos\alpha - \sin\phi\sin\alpha \\ \sin\phi\cos\alpha + \cos\phi\sin\alpha \end{pmatrix} = \begin{pmatrix} \cos(\alpha + \phi) \\ \sin(\alpha + \phi) \end{pmatrix} = |\alpha + \phi\rangle . \end{aligned} \tag{2.48}$$

$|\alpha\rangle$ can also be written in the $|0\rangle, |1\rangle$ basis states. By linearity, each is rotated by ϕ, so $|\alpha\rangle$ becomes $|\alpha + \phi\rangle$.

(d)

$$\mathbf{R}_\phi |R\rangle = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \frac{1}{\sqrt{2}} \begin{pmatrix} 1 \\ i \end{pmatrix} = \frac{1}{\sqrt{2}} \begin{pmatrix} e^{-i\phi} \\ ie^{-i\phi} \end{pmatrix} = e^{-i\phi} |R\rangle . \quad (2.49)$$

The Euler identity allows us to factor out the phase shift factor.

(e) When $\phi = \pi/8$,

$$\begin{aligned} \mathbf{R}_{\pi/8}\mathbf{Z}\mathbf{R}_{-\pi/8} &= \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} \Bigg|_{\phi=\pi/8} \\ &= \begin{pmatrix} \cos\phi & \sin\phi \\ -\sin\phi & \cos\phi \end{pmatrix} \begin{pmatrix} \sin\phi & \cos\phi \\ \cos\phi & -\sin\phi \end{pmatrix} \Bigg|_{\phi=\pi/8} \\ &= \begin{pmatrix} \sin(2\phi) & \cos(2\phi) \\ \cos(2\phi) & -\sin(2\phi) \end{pmatrix} \Bigg|_{\pi/8} \\ &= \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \equiv \mathbf{H}. \end{aligned} \quad (2.50)$$

□

2.4 Problem set 4

1. Measurements. Consider measurements of the two-Qbit states

$$|\psi\rangle_2 = |+\rangle\,|-\rangle$$
$$|\phi\rangle_2 = \frac{1}{\sqrt{2}}\left(|0\rangle\,|+\rangle - |1\rangle\,|-\rangle\right)$$
$$|\chi\rangle_2 = \frac{1}{\sqrt{2}}\left(|+\rangle\,|+\rangle + |-\rangle\,|-\rangle\right)$$

(a) Write out the states in Dirac notation as

$$|\psi\rangle_2 = \sum_{0\le x<2^2} \psi_x\,|x\rangle$$
$$|\phi\rangle_2 = \sum_{0\le x<2^2} \phi_x\,|x\rangle$$
$$|\chi\rangle_2 = \sum_{0\le x<2^2} \chi_x\,|x\rangle$$

for $x \in \{0,1\}^2$.

(b) For each state $|\psi\rangle$, $|\phi\rangle$, and $|\chi\rangle$, what is the probability of finding the system in each of the four different states $|x\rangle$?

(c) Which, if any, of these states are entangled? Explain.

Solution:

(a) We use the identities:

$$|+\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + |1\rangle\right), \quad \text{and} \quad |-\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle - |1\rangle\right) \tag{2.51}$$

to write

$$\begin{aligned}|\psi\rangle_2 &= \frac{1}{2}\left(|0\rangle + |1\rangle\right)\left(|0\rangle - |1\rangle\right)\\ &= \frac{1}{2}\left(|00\rangle + |01\rangle - |10\rangle - |11\rangle\right)\\ &= \boxed{\frac{1}{2}\,|0\rangle_2 + \frac{1}{2}\,|1\rangle_2 - \frac{1}{2}\,|2\rangle_2 - \frac{1}{2}\,|3\rangle_2}\end{aligned} \tag{2.52}$$

$$\begin{aligned}|\phi\rangle_2 &= \frac{1}{2}\left(|0\rangle\left[|0\rangle + |1\rangle\right] - |1\rangle\left[|0\rangle - |1\rangle\right]\right)\\ &= \frac{1}{2}\left(|00\rangle + |01\rangle - |10\rangle + |11\rangle\right)\\ &= \boxed{\frac{1}{2}\,|0\rangle_2 + \frac{1}{2}\,|1\rangle_2 - \frac{1}{2}\,|2\rangle_2 + \frac{1}{2}\,|3\rangle_2}\end{aligned} \tag{2.53}$$

$$\begin{aligned}
|\chi\rangle_2 &= \frac{1}{2\sqrt{2}}\left([|0\rangle + |1\rangle]\,[|0\rangle + |1\rangle] + [|0\rangle - |1\rangle]\,[|0\rangle - |1\rangle]\right) \\
&= \frac{1}{2\sqrt{2}}\left(|00\rangle + |11\rangle + |00\rangle + |11\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\
&= \boxed{\frac{1}{\sqrt{2}}|0\rangle_2 + \frac{1}{\sqrt{2}}|3\rangle_2}
\end{aligned} \tag{2.54}$$

(b) We simply read off the the modulus squares of the coefficients. $|\psi\rangle_2$ and $|\phi\rangle_2$ are equal superpositions of the basis states, while $|\chi\rangle_2$ is an equal superposition of only the "symmetric" states.

$$P(|\psi\rangle_2 \to |x\rangle) = \frac{1}{4}, \quad \forall x \in \{0,1\}^2 \tag{2.55}$$

$$P(|\phi\rangle_2 \to |x\rangle) = \frac{1}{4}, \quad \forall x \in \{0,1\}^2 \tag{2.56}$$

$$\begin{cases} P(|\chi\rangle_2 \to |0\rangle_2) = P(|\chi\rangle_2 \to |3\rangle_2) = \frac{1}{2} \\ P(|\chi\rangle_2 \to |x\rangle) = 0, \quad x = 1, 2. \end{cases} \tag{2.57}$$

(c) As column vectors:

$$|\psi\rangle_2 = \frac{1}{2}\begin{pmatrix} 1 & 1 & -1 & -1 \end{pmatrix}^\top \tag{2.58}$$

$$|\phi\rangle_2 = \frac{1}{2}\begin{pmatrix} 1 & 1 & -1 & 1 \end{pmatrix}^\top \tag{2.59}$$

$$|\chi\rangle_2 = \frac{1}{\sqrt{2}}\begin{pmatrix} 1 & 0 & 0 & 1 \end{pmatrix}^\top. \tag{2.60}$$

A state $\begin{pmatrix} a & b & c & d \end{pmatrix}^\top$ can be written as a tensor product if and only if $ad = bc$. Only $|\psi\rangle_2$ satisfies this constraint (the tensor factorization is $|+-\rangle$). On the other hand, $|\phi\rangle_2$ and $|\chi\rangle_2$ are entangled states, because they don't satisfy the constraint $ad = bc$.

□

2. Creating the Bell States with Operators. The "Bell States" are the maximally entangled states

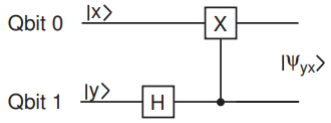
$$|\psi_{yx}\rangle = \frac{1}{\sqrt{2}}\left(|0x\rangle + (-1)^y\,|1\bar{x}\rangle\right), \tag{2.61}$$

where $x, y \in \{0, 1\}$. We showed in class that you can generate $|\psi_{00}\rangle$ by acting with a cNOT gate and a Hadamard gate on the state $|00\rangle$, which turns an separable state into an entangled state, which is why cNOT is called an entangling gate.

Use your skill with the algebra of multi-Qbit operators and states show that

$$|\psi_{yx}\rangle = \mathbf{C}_{10}\mathbf{H}_1\,|yx\rangle\,. \tag{2.62}$$

Note that in this case the Hadamard acts on Qbit 1, and the control Qbit for the cNOT is Qbit 1 and the target Qbit is Qbit 0. This operation can be drawn as a quantum circuit diagram as shown below.



Solution: $\mathbf{C}_{10}$ does nothing to Qbit 0 when Qbit 1 is in state 0, and flips Qbit 0 when Qbit 1 is in state 1.

$$\begin{aligned}
\mathbf{C}_{10}\mathbf{H}_1\,|yx\rangle &= \mathbf{C}_{10}(\mathbf{H}\otimes\mathbb{I})\,|y\rangle\,|x\rangle \\
&= \mathbf{C}_{10}\left(\frac{1}{\sqrt{2^1}}\left(|0\rangle + (-1)^y\,|1\rangle\right)\right)\otimes|x\rangle \\
&= \frac{1}{\sqrt{2}}\mathbf{C}_{10}\left(|0x\rangle + (-1)^y\,|1x\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left(|0x\rangle + (-1)^y\,|1\bar{x}\rangle\right) \\
&\equiv |\psi_{yx}\rangle
\end{aligned} \tag{2.63}$$

□

3. Measurements of the Bell States. The "Bell States" are the maximally entangled states

$$|\psi_{yx}\rangle = \frac{1}{\sqrt{2}}\left(|0x\rangle + (-1)^y\,|1\bar{x}\rangle\right), \tag{2.64}$$

where $x, y \in \{0, 1\}$. These are all orthogonal states.

If measurements are made in the computational basis, can you distinguish any one of these states from the others?

Solution: We cannot uniquely determine which Bell state is being measured (in the computation basis) without "untanglement", i.e., reversing the process in the previous problem to get $|yx\rangle$ from $|\psi_{yx}\rangle$.

Suppose (to get a contradiction) that we can. Then consider this counter-example. When we measure Qbit 1 in either one of the four Bell states, we cannot distinguish the Bell states

$$\begin{aligned} |\psi_{00}\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) \\ |\psi_{10}\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle\right) \end{aligned} \tag{2.65}$$

from each other. In both cases, we find Qbit 1 to be in state 0 with probability 1/2. Also, in both cases, whenever Qbit 1 is in state 0, Qbit 0 is also in state 0. So, there is no way to distinguish these Bell states by measuring in the computational basis without transforming them first. □

4. Disentangling the Bell States. Above you showed that the individual Bell states can all be created using a Hadamard gate on bit 0 and a cNOT gate using bit 0 as the control and bit 1 as the target. A question arises in quantum teleportation is "which Bell state is the two-Qbit system in?" Clearly you need to do something more than just make measurements to find out for sure which one it is. Design a system that will allow you to unambigouously identify which of the Bell states a two-Qbit system is in. To do this, you should consider figuring out a set of unitary operations that gives

$$|\psi_{yx}\rangle \implies |yx\rangle \equiv |y\rangle\,|x\rangle \tag{2.66}$$

with $x, y \in \{0,1\}^2$, and demonstrate that it works for all four states.

Solution: We use the fact that quantum computational transformations are reversible. From Problem 2, we know that $|\psi_{yx}\rangle = \mathbf{C}_{10}\mathbf{H}_1\,|yx\rangle$. By applying the inverse of $\mathbf{C}_{10}\mathbf{H}_1$ (whose existence is guaranteed) to $|\psi_{yx}\rangle$, we will get back $|yx\rangle$. We know that $\mathbf{H}^{-1} = \mathbf{H}$. Further, $(\mathbf{C}_{10})^{-1} = \mathbf{C}_{10}$ because

$$\begin{aligned}\mathbf{C}_{10}\mathbf{C}_{10}\,|yx\rangle &= \mathbf{C}_{10}\,|y\rangle\,|x\oplus y\rangle \\ &= |y\rangle\,|x\oplus y\oplus y\rangle \\ &= |yx\rangle \\ \iff \mathbf{C}_{10}\mathbf{C}_{10} = \mathbb{I} &\iff \mathbf{C}_{10} = (\mathbf{C}_{10})^{-1}.\end{aligned} \tag{2.67}$$

Therefore,

$$(\mathbf{C}_{10}\mathbf{H}_1)^{-1} = \mathbf{H}_1^{-1}(\mathbf{C}_{10})^{-1} = \mathbf{H}_1\mathbf{C}_{10}. \tag{2.68}$$

We will check that $\mathbf{H}_1\mathbf{C}_{10}\,|\psi_{yx}\rangle = |yx\rangle$:

$$\begin{aligned}\mathbf{H}_1\mathbf{C}_{10}\,|\psi_{yx}\rangle &= \mathbf{H}_1\mathbf{C}_{10}\left(\frac{1}{\sqrt{2}}\left[|0x\rangle + (-1)^y\,|1\bar{x}\rangle\right]\right) \\ &= \frac{1}{\sqrt{2}}\mathbf{H}_1\left(|0x\rangle + (-1)^y\,|1x\rangle\right) \\ &= \frac{1}{\sqrt{2}}\left(\frac{1}{\sqrt{2}}\left[|0\rangle + |1\rangle\right] + (-1)^y\frac{1}{\sqrt{2}}\left[|0\rangle - |1\rangle\right]\right)\otimes|x\rangle \\ &= \frac{1}{2}\underbrace{\left(|0\rangle + |1\rangle + (-1)^y\,|0\rangle - (-1)^y\,|1\rangle\right)}_{\boxed{?}}\otimes|x\rangle\end{aligned} \tag{2.69}$$

To find $\boxed{?}$ we observe that

$$\begin{cases}|0\rangle + |1\rangle + (-1)^y\,|0\rangle - (-1)^y\,|1\rangle = 2\,|0\rangle\,, & \text{whenever } y = 0 \\ |0\rangle + |1\rangle + (-1)^y\,|0\rangle - (-1)^y\,|1\rangle = 2\,|1\rangle\,, & \text{whenever } y = 1\end{cases} \tag{2.70}$$

and so $\boxed{?}$ is identically $2\,|y\rangle$. With this,

$$\mathbf{H}_1\mathbf{C}_{10}\,|\psi_{yx}\rangle = \frac{1}{2}\cdot 2\,|y\rangle\,|x\rangle \equiv |yx\rangle\,. \tag{2.71}$$

□

2.5 Problem set 5

1. Operator Communication. This will get you to think about a problem related to "superdense coding." Alice and Bob share a pair of Qbits in the entangled state $|\psi_{01}\rangle$. For concreteness, assume that Alice possesses Qbit 0 and Bob possesses Qbit 1. Alice chooses to apply one of four operators to her Qbit: $\mathbb{I}, \mathbf{X}_0, \mathbf{Y}_0$, or $\mathbf{Z}_0$. After doing that she sends her Qbit to Bob. Explain how Bob can unambiguously determine what one of the four operators Alice applied to her one Qbit by operating on and making measurements of the Qbit pair that he now possesses.

Solution: Let Alice and Bob share a pair of Qbits in the entangled state $|\psi_{01}\rangle$:

$$|\psi_{01}\rangle_{BA} = \frac{1}{\sqrt{2}} \left(|01\rangle_{BA} + |10\rangle_{BA} \right). \tag{2.72}$$

Alice can apply $\mathbb{I}_0, \mathbf{X}_0, \mathbf{Y}_0$, or $\mathbf{Z}_0$ on her one Qbit. In each case, the outcome is

$$\begin{aligned}
(\mathbb{I} \otimes \mathbb{I}) |\psi_{01}\rangle &= \frac{1}{\sqrt{2}} \left(|01\rangle + |10\rangle \right) = |\psi_{01}\rangle \\
(\mathbb{I} \otimes \mathbf{X}) |\psi_{01}\rangle &= \frac{1}{\sqrt{2}} \left(|00\rangle + |11\rangle \right) = |\psi_{00}\rangle \\
(\mathbb{I} \otimes \mathbf{Y}) |\psi_{01}\rangle &= \frac{1}{\sqrt{2}} \left(i(-1)^1 |00\rangle + i(-1)^0 |11\rangle \right) = \frac{-i}{\sqrt{2}} \left(|00\rangle - |11\rangle \right) = -i |\psi_{10}\rangle \\
(\mathbb{I} \otimes \mathbf{Z}) |\psi_{01}\rangle &= \frac{1}{\sqrt{2}} \left((-1)^1 |01\rangle + (-1)^0 |10\rangle \right) = \frac{-1}{\sqrt{2}} \left(|01\rangle - |10\rangle \right) = - |\psi_{11}\rangle .
\end{aligned} \tag{2.73}$$

What Bob can do given these (distinguishable) outputs is inverting them using $\mathbf{H}_1\mathbf{C}_{10}$. By making measurements in the computational basis, Bob can determine exactly what operator Alice has used on her Qbit:

$$\begin{aligned}
|\psi_{01}\rangle &\xrightarrow{\mathbf{H}_1\mathbf{C}_{10}} |01\rangle \implies \text{Alice used } \mathbb{I} \\
|\psi_{00}\rangle &\xrightarrow{\mathbf{H}_1\mathbf{C}_{10}} |00\rangle \implies \text{Alice used } \mathbf{X}_0 \\
-i |\psi_{10}\rangle &\xrightarrow{\mathbf{H}_1\mathbf{C}_{10}} -i |10\rangle \implies \text{Alice used } \mathbf{Y}_0 \\
- |\psi_{11}\rangle &\xrightarrow{\mathbf{H}_1\mathbf{C}_{10}} - |11\rangle \implies \text{Alice used } \mathbf{Z}_0.
\end{aligned} \tag{2.74}$$

□

2. Partial Measurements. Consider the entangled state

$$|\psi\rangle_2 = \frac{1}{\sqrt{2}}|00\rangle - \frac{1}{2}|10\rangle + \frac{i}{2}|11\rangle \tag{2.75}$$

Consider a partial measurement of the system where Qbit 1 is measured.

(a) What is the probability that when Qbit 1 is measured the outcome will be a 0? What is the state of the system after that bit has been measured?

(b) What is the probability that when Qbit 1 is measured the outcome will be a 1? What is the state of the system after that bit has been measured?

Solution:

(a) The probability that when Qbit 1 is measured the outcome will be a 0 is $(1/\sqrt{2})^2 = \boxed{1/2}$. The state of the system after Qbit 1 has been measured to be 0 is

$$|\psi'\rangle_2 = \frac{\frac{1}{\sqrt{2}}|00\rangle}{\left\|\frac{1}{\sqrt{2}}|00\rangle\right\|} = \boxed{|00\rangle} \tag{2.76}$$

(b) The probability that when Qbit 1 is measured the outcome will be a 1 is one minus the probability that when Qbit 1 is measured the outcome will be a 0. So, the answer is $1 - 1/2 = \boxed{1/2}$. The state of the system after Qbit 1 has been measured to be 1 is

$$|\psi'\rangle_2 = \frac{\frac{-1}{2}|00\rangle + \frac{i}{2}|11\rangle}{\left\|\frac{-1}{2}|00\rangle + \frac{i}{2}|11\rangle\right\|} = \boxed{\frac{-1}{\sqrt{2}}(|10\rangle - i|11\rangle)} \tag{2.77}$$

□

3. Games with Algebra. The original paper on quantum quantum teleportation by Bennett, Brassard, Crépeau, Jozsa, Peres, and Wooters in 1993 was described to me by Bill Wooters as "games with algebra." The original article can be found via this link.

Here's a way to understand the algebraic game. In the initial state of the system, Alice has two Qbits (2 and 1) and Bob has one Qbit (0). Qbits 1 and 0 are in the entangled state $|\psi_{00}\rangle$, and Qbit 2 is in an unknown state $|\alpha\rangle = \alpha_0 |0\rangle + \alpha_1 |1\rangle$. The state is written most naturally as a 3-bit partially entangled state

$$|\psi\rangle_3 = |\alpha\rangle |\psi_{00}\rangle . \tag{2.78}$$

Show that this state can be written (most unnaturally!) as

$$|\psi\rangle_3 = \frac{1}{2} \left(|\psi_{00}\rangle |\alpha\rangle + |\psi_{01}\rangle \mathbf{X} |\alpha\rangle + |\psi_{10}\rangle \mathbf{Z} |\alpha\rangle + |\psi_{11}\rangle \mathbf{XZ} |\alpha\rangle \right) \tag{2.79}$$

where the single-Qbit operators $\mathbf{X}$ and $\mathbf{Z}$ act only on Qbit 0 and the Bell-states are for Qbits 2 and 1. Explain how the observation that you just demonstrated provided the motivation for the protocol that we discussed in class for quantum teleportation.

Solution: We want to show that

$$\frac{1}{2} \left(|\psi_{00}\rangle |\alpha\rangle + |\psi_{01}\rangle \mathbf{X} |\alpha\rangle + |\psi_{10}\rangle \mathbf{Z} |\alpha\rangle + |\psi_{11}\rangle \mathbf{XZ} |\alpha\rangle \right) = |\alpha\rangle |\phi_{00}\rangle , \tag{2.80}$$

to which end we use the identities:

$$\begin{aligned} \mathbf{X} |\alpha\rangle &= \mathbf{X}(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_0 |1\rangle + \alpha_1 |0\rangle \\ \mathbf{Z} |\alpha\rangle &= \mathbf{Z}(\alpha_0 |0\rangle + \alpha_1 |1\rangle) = \alpha_0 |0\rangle - \alpha_1 |1\rangle \\ \mathbf{XZ} |\alpha\rangle &= \mathbf{X}(\alpha_0 |0\rangle - \alpha_1 |1\rangle) = \alpha_0 |1\rangle - \alpha_1 |0\rangle . \end{aligned} \tag{2.81}$$

We proceed by expanding the LHS until we get to the RHS:

$$\begin{aligned} & \frac{1}{2} \left(|\psi_{00}\rangle |\alpha\rangle + |\psi_{01}\rangle \mathbf{X} |\alpha\rangle + |\psi_{10}\rangle \mathbf{Z} |\alpha\rangle + |\psi_{11}\rangle \mathbf{XZ} |\alpha\rangle \right) = |\alpha\rangle |\phi_{00}\rangle \\ &= \frac{1}{2\sqrt{2}} \left[(|00\rangle + |11\rangle) (\alpha_0 |0\rangle + \alpha_1 |1\rangle) + (|10\rangle + |01\rangle) (\alpha_1 |0\rangle + \alpha_0 |1\rangle) \right. \\ &\qquad \left. + (|00\rangle - |11\rangle) (\alpha_0 |0\rangle - \alpha_1 |1\rangle) + (|01\rangle - |10\rangle) (\alpha_0 |1\rangle - \alpha_1 |0\rangle) \right] \\ &= \frac{1}{2\sqrt{2}} \left[2\alpha_0 |000\rangle + (\alpha_1 - \alpha_1) |001\rangle + (\alpha_1 - \alpha_1) |010\rangle + 2\alpha_0 |011\rangle \right. \\ &\qquad \left. + 2\alpha_1 |100\rangle + (\alpha_0 - \alpha_0) |101\rangle + (\alpha_0 - \alpha_0) |110\rangle + 2\alpha_1 |111\rangle \right] \\ &= \frac{1}{\sqrt{2}} \left(\alpha_0 |000\rangle + \alpha_0 |011\rangle + \alpha_1 |100\rangle + \alpha_1 |111\rangle \right) \\ &= (\alpha_0 |0\rangle + \alpha_1 |1\rangle) \frac{1}{\sqrt{2}} (|00\rangle + |11\rangle) \\ &= |\alpha\rangle |\psi_{00}\rangle . \end{aligned} \tag{2.82}$$

This result motivates the protocol for quantum teleportation in the following way. Suppose Alice (who is the sender in this case) has a Qbit in some arbitrary state $|\alpha\rangle$ and wants to send this state (not the Qbit, just the state) to Bob, without knowing what state his Qbit is in. What Alice and Bob can do is to have first share an pair of Qbits in the Bell state $|00\rangle$, each possessing one Qbit of the pair. Then, they can consider the system of three Qbits whose state is $|\psi\rangle_3 = |\alpha\rangle \otimes |\psi_{00}\rangle$. From the exercise, we know that $|\psi\rangle_3$ can be written as

$$|\psi\rangle_3 = \frac{1}{2}\left(|\psi_{00}\rangle\,|\alpha\rangle + |\psi_{01}\rangle\,\mathbf{X}\,|\alpha\rangle + |\psi_{10}\rangle\,\mathbf{Z}\,|\alpha\rangle + |\psi_{11}\rangle\,\mathbf{XZ}\,|\alpha\rangle\right). \tag{2.83}$$

Now, we see that Alice's Qbits are in one of the four Bell states $|\psi_{yx}\rangle$. By applying the inversion $\mathbf{H}_2\mathbf{C}_{21}$ to $|\psi_{yx}\rangle$, followed by measurements on the computational basis, Alice can send two classical bits $|y\rangle$ and $|x\rangle$, $x, y \in \{0, 1\}$ to Bob. With these classical bits, there is a way for Bob to make sure his Qbit is in $|\alpha\rangle$:

- If Alice sends $|00\rangle$, Bob does nothing. His Qbit is already in $|\alpha\rangle$.
- If Alice sends $|01\rangle$, Bob applies $\mathbf{X}$ to his Qbit: $\mathbf{X}(\mathbf{X}\,|\alpha\rangle) = |\alpha\rangle$.
- If Alice sends $|10\rangle$, Bob applies $\mathbf{Z}$ to his Qbit: $\mathbf{Z}(\mathbf{Z}\,|\alpha\rangle) = |\alpha\rangle$.
- if Alice sends $|11\rangle$, Bob applies $\mathbf{ZX}$ to his Qbit: $\mathbf{ZX}(\mathbf{XZ})\,|\alpha\rangle = |\alpha\rangle$.

Here we rely on the fact that the Pauli matrices are involutory, i.e. $\sigma_i^2 = \mathbb{I}, \forall i = 1, 2, 3$.

With this protocol, Alice can send the state $|\alpha\rangle$ of her Qbit to Bob's Qbit. □

4. The goal of this problem is to consider a slightly simplified version of quantum teleportation. In this problem Alice and Bob start (as with most problems like this) by sharing two Qbits in an entangled state. For the purposes of this problem the two QBits are an entangled pair of photons in the entangled (Bell) state

$$|\psi_{00}\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle . \tag{2.84}$$

Alice wants Bob to have a photon Qbit in a specific state $|\phi\rangle$ but wants to send Bob the minimum amount of information possible for him to make it. The goal of this problem is to show that (if they have the resource of an entangle Qbit pair) she can make that happen by sending Bob only **one** classical bit of information.

Start by remembering the definition of a rotation operator $\mathbf{R}_\phi$ on photons, which takes a photon linearly polarized at angle α and converts it into a linearly polarized photon at angle $\alpha + \phi$. The matrix representation of the rotation operator in the computational basis is

$$\mathbf{R}_\phi = \begin{pmatrix} \cos\phi & -\sin\phi \\ \sin\phi & \cos\phi \end{pmatrix} . \tag{2.85}$$

As you showed on an earlier assignment $\mathbf{R}_\phi$ is a unitary operator with $\mathbf{R}_\phi^\dagger = \mathbf{R}_\phi^{-1} = \mathbf{R}_{-\phi}$ and $\mathbf{R}_\phi|0\rangle = |\phi\rangle$ and $\mathbf{R}_\phi|1\rangle = |\pi/2 + \phi\rangle$. To slightly simplify notation, call $|\pi/2 + \phi\rangle = |\phi_\perp\rangle$.

(a) Show that $\mathbf{ZX}|\phi_\perp\rangle = |\phi\rangle$.

(b) Show that the entangled (Bell) state $|\psi_{00}\rangle$ can also be written as

$$|\psi_{00}\rangle = \frac{1}{\sqrt{2}}|\phi\phi\rangle + \frac{1}{\sqrt{2}}|\phi_\perp\phi_\perp\rangle . \tag{2.86}$$

This says something special about the state $|\psi_{00}\rangle$. What?

(c) Imagine that Alice applies the operator $\mathbf{R}_\phi^{-1}$ to her Qbit. What is the state of the system after she does that?

(d) Alice measures her QBit. What is the probability that she measures 0? What is the probability that she measures 1?

(e) What state does Bobs QBit have if Alice measures 0? What state does Bobs QBit have if she measures 1?

(f) If Alice tells Bob the value that she measured, what can he do to assure himself that his Qbit is in the state $|\phi\rangle$?

Solution:

(a) We want to show that $\mathbf{ZXR}_{\pi/2} = \mathbb{I}$, since

$$\begin{aligned} \mathbf{ZX}\,|\phi_\perp\rangle = |\phi\rangle \,\forall\, |\phi\rangle &\iff \mathbf{ZXR}_{\pi/2}\,|\phi\rangle = |\phi\rangle \,\forall\, |\phi\rangle \\ &\iff \mathbf{ZXR}_{\pi/2} \equiv \mathbb{I}. \end{aligned} \tag{2.87}$$

Since $\mathbf{ZXR}_{\pi/2}$ is a linear operator, its action is completely determined by what it does to the basis vectors:

$$\begin{aligned} \mathbf{ZXR}_{\pi/2}\,|0\rangle &= \mathbf{ZX}\,|\pi/2\rangle = \mathbf{Z}\,|0\rangle = |0\rangle \\ \mathbf{ZXR}_{\pi/2}\,|1\rangle &= \mathbf{ZX}\,|\pi\rangle = -\mathbf{ZX}\,|0\rangle = -\mathbf{Z}\,|1\rangle = |1\rangle\,, \end{aligned} \tag{2.88}$$

where we have used the fact that $|1\rangle \equiv |\pi/2\rangle$ and $-|0\rangle \equiv |\pi\rangle$. This can readily be checked by plugging in values of ϕ into $\begin{pmatrix}\cos\phi & \sin\phi\end{pmatrix}^\top$.

Since $\mathbf{ZXR}_{\pi/2}$ acts as the identity function on the entire computational basis, it *is* the identity function, by linearity. Evidently: for any $|\phi\rangle = \cos\phi\,|0\rangle + \sin\phi\,|1\rangle$,

$$\mathbf{ZXR}_{\pi/2}(\cos\phi\,|0\rangle + \sin\phi\,|1\rangle) = \cos\phi\,|0\rangle + \sin\phi\,|1\rangle \equiv |\phi\rangle\,. \tag{2.89}$$

Therefore,

$$|\phi\rangle = \mathbf{ZXR}_{\pi/2}\,|\phi\rangle = \mathbf{ZX}\,|\phi_\perp\rangle\,, \forall\, |\phi\rangle\,. \tag{2.90}$$

(b) In the computational basis, $|\phi\rangle = \begin{pmatrix}\cos\phi & \sin\phi\end{pmatrix}^\top$, and $|\phi_\perp\rangle = \begin{pmatrix}-\sin\phi & \cos\phi\end{pmatrix}^\top$. With these, we expand the RHS to (hopefully) get the LHS:

$$\begin{aligned} &\frac{1}{\sqrt{2}}\left(|\phi\phi\rangle + |\phi_\perp\phi_\perp\rangle\right) \\ =&\frac{1}{\sqrt{2}}\left[\begin{pmatrix}\cos\phi\\ \sin\phi\end{pmatrix} \otimes \begin{pmatrix}\cos\phi\\ \sin\phi\end{pmatrix} + \begin{pmatrix}-\sin\phi\\ \cos\phi\end{pmatrix} \otimes \begin{pmatrix}-\sin\phi\\ \cos\phi\end{pmatrix}\right] \\ =&\frac{1}{\sqrt{2}}\left[\begin{pmatrix}\cos^2\phi\\ \frac{\sin 2\phi}{2}\\ \frac{\sin 2\phi}{2}\\ \sin^2\phi\end{pmatrix} + \begin{pmatrix}\sin^2\phi\\ -\frac{\sin 2\phi}{2}\\ -\frac{\sin 2\phi}{2}\\ \cos^2\phi\end{pmatrix}\right] \\ =&\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 0 & 0 & 1\end{pmatrix}^\top \\ \equiv&\,|\phi_{00}\rangle\,. \end{aligned} \tag{2.91}$$

So we're done. This result says that $|\phi_{00}\rangle$ is invariant under rotations by ϕ on both Qbits, i.e. $\mathbf{R}_\phi \otimes \mathbf{R}_\phi\,|\psi_{00}\rangle = |\psi_{00}\rangle$. Further, by the previous part, $|\psi_{00}\rangle$ is also invariant under $(\mathbf{ZX}) \otimes (\mathbf{ZX})$, by symmetry:

$$(\mathbf{ZX}) \otimes (\mathbf{ZX})\,|\psi_{00}\rangle \xrightarrow{\phi \leftrightarrow \phi_\perp} |\psi_{00}\rangle\,. \tag{2.92}$$

(c) Let

$$|\phi_{00}\rangle = \frac{1}{\sqrt{2}}\left(|\phi\phi\rangle + |\phi_\perp\phi_\perp\rangle\right) \tag{2.93}$$

be given. By the previous part, we know that the expression above is equivalent to $(1/\sqrt{2})\left(|00\rangle + |11\rangle\right)$. So, when Alice applies the operator $\mathbf{R}_\phi^{-1} \equiv \mathbf{R}_{-\phi}$ to her (0) Qbit, $|\phi_{00}\rangle$ is transformed into

$$\begin{aligned}(\mathbb{I}\otimes\mathbf{R}_{-\phi})\,|\psi_{00}\rangle &= \frac{1}{\sqrt{2}}\left(|\phi\rangle\,\mathbf{R}_{-\phi}\,|\phi\rangle + |\phi_\perp\rangle\,\mathbf{R}_{-\phi}\,|\phi_\perp\rangle\right)\\ &= \boxed{\frac{1}{\sqrt{2}}\left(|\phi 0\rangle + |\phi_\perp 1\rangle\right)}\end{aligned} \tag{2.94}$$

where we have used the identifications: $|\phi_\perp\rangle \equiv |\phi + \pi/2\rangle$ and $|\pi/2\rangle \equiv |1\rangle$.

(d) Evidently, the probability that Alice measures 0 is 1/2, because $(\mathbb{I}\otimes\mathbf{R}_{-\phi})\,|\psi_{00}\rangle$ is in equal superposition of $|\phi 0\rangle$ and $|\phi_\perp 1\rangle$. By symmetry, the probability that Alice measures 1 is also 1/2

(e) If Alice measures 0, then Bob's Qbit is in the state $|\phi\rangle$. If Alice measures 1, then Bob's Qbit is in the state $|\phi_\perp\rangle$.

(f) Let Alice's measurement be given as $x \in \{0,1\}$. To ensure that his Qbit is in the state $|\phi\rangle$, Bob must leave his Qbit (in state $|\phi\rangle$) alone if $x = 0$ and apply $\mathbf{ZX}$ to his Qbit (now in state $|\phi_\perp\rangle$) to change it to $|\phi\rangle$ if $x = 1$.

The protocol can be summarized as

$$\begin{cases} x = 0 \implies \text{Bob does nothing to his Qbit} \implies \mathbb{I}\,|\phi\rangle = |\phi\rangle \\ x = 1 \implies \text{Bob applies } \mathbf{ZX} \text{ to his Qbit} \implies \mathbf{ZX}\,|\phi_\perp\rangle = |\phi\rangle\,. \end{cases} \tag{2.95}$$

We can write this protocol more concisely as $\boxed{(\mathbf{ZX})^x, x \in \{0,1\}}$.

□

2.6 Problem set 6

1. Monroe Colloquium

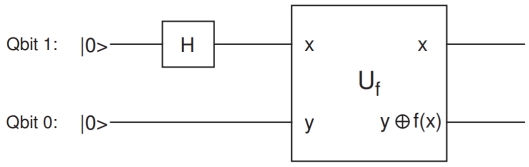
2. Deutsch's Original Algorithm. In class on March 12th we discussed the Deutsch problem. Determining if a function $f(x)$ is either constant or balanced, where x is a single bit.

There are four functions on one bit, f_0 gives the constant output 0, f_1 gives the constant output 1, f_b (buffer) gives an output equal to the input, and f_i (invert) has an output equal to x. The functions f_0 and f_1 are constant, and the functions f_b and f_i are balanced. Tabulated they are:

x	0	1
$f_0(x)$	0	0
$f_1(x)$	1	1
$f_b(x)$	0	1
$f_i(x)$	1	0

In Appendix F, Mermin looks at the approach that Deutsch originally considered (in 1985) for determining if a function $f : \{0,1\} \to \{0,1\}$ is balanced or constant. As mentioned in class the algorithm outlined in Mermin Chapter 2 and in class was developed much later (1997) by Cleve, Ekert, Macchiavello, and Mosca using the idea of "phase kickback."

Consider the "quantum parallelism" approach to the Deutsch problem shown below as a quantum circuit diagram.



In the diagram Qbit 1 is the "input register" and Qbit 0 is the "output register."

(a) Show that the output of the operator $\mathbf{U}_f$ is, as written in Eq. F.2-F.5 in Mermin

$$\begin{aligned} |\psi_0\rangle &= |+\rangle\,|0\rangle \\ |\psi_1\rangle &= |+\rangle\,|1\rangle \\ |\psi_b\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle\,|0\rangle + |1\rangle\,|1\rangle\right) = |\psi_{00}\rangle \\ |\psi_i\rangle &= \frac{1}{\sqrt{2}}\left(|0\rangle\,|1\rangle + |1\rangle\,|0\rangle\right) = |\psi_{01}\rangle \end{aligned} \tag{2.96}$$

The states due to the constant functions are product states, and the output states due to the balanced functions are two of the entangled Bell states. Deutschs idea was to show that you could manipulate the output state in such a way that a measurement of the output could distinguish between constant and balanced functions half the time, but that half the time you would make a measurement that gave no information at all.

(b) Think about measuring the states of Qbits 0 and 1 for the four different states above. What does it allow you to distinguish about the states?

(c) Consider the approach described by Mermin when you apply a Hadamard gate to both Qbits 0 and 1 after the application of $\mathbf{U}_f$. Show that Eqs. F.6-F.9 in Mermin are correct, namely that

$$\begin{aligned}
\mathbf{H}^{\otimes 2}\left|\psi_0\right\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |01\rangle\right) \\
\mathbf{H}^{\otimes 2}\left|\psi_1\right\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |01\rangle\right) \\
\mathbf{H}^{\otimes 2}\left|\psi_b\right\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle + |11\rangle\right) = |\psi_{00}\rangle \\
\mathbf{H}^{\otimes 2}\left|\psi_i\right\rangle &= \frac{1}{\sqrt{2}}\left(|00\rangle - |11\rangle = |\psi_{10}\rangle\right)
\end{aligned} \tag{2.97}$$

where $\mathbf{H}^{\otimes 2}$ is the Hadamard gate applied to both Qbits.

(d) In your own words, explain how these states allow you to determine whether or not the function is constant or balance half the time.

(e) (Optional. For those who took PH431.) Follow, and outline in detail, Mermins argument that there is NO possible operation you can do to the two output Qbits that allow you to determine whether the function is balanced or constant with certainty (100% of the time).

(f) (Optional^2. For those who are looking for a serious challenge.) Follow, and outline in detail, Mermins argument that the best you can do with operations on the two output Qbits is to determine whether the function is balanced or constant 50% of the time.

Solution:

(a) The unitary $\mathbf{U}_f$ is such that

$$\mathbf{U}_f\left|xy\right\rangle = \left|x\right\rangle\left|y \oplus f(x)\right\rangle, \tag{2.98}$$

where $y \in \{0,1\}$ and $|x\rangle = \mathbf{H}\,|x'\rangle = |\pm\rangle$ with $x' \in \{0,1\}$. With this, the

general output of $\mathbf{U}_f$ has the form:

$$\begin{aligned}\mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) &= \frac{1}{\sqrt{2}}\mathbf{U}_f(\left|00\right\rangle + \left|10\right\rangle) \\ &= \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|0\oplus f(0)\right\rangle + \left|1\right\rangle\left|0\oplus f(1)\right\rangle\right) \\ &= \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|f(0)\right\rangle + \left|1\right\rangle\left|f(1)\right\rangle\right) \end{aligned} \tag{2.99}$$

like that in Eq. F-1. It follows that the four possible f's and associated outputs of $\mathbf{U}_f$ are

$$\begin{aligned} f\equiv f_0: &\quad \left|\psi_0\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|0\right\rangle + \left|1\right\rangle\left|0\right\rangle\right) = \left|+\right\rangle\left|0\right\rangle \\ f\equiv f_1: &\quad \left|\psi_1\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|1\right\rangle + \left|1\right\rangle\left|1\right\rangle\right) = \left|+\right\rangle\left|1\right\rangle \\ f\equiv f_b: &\quad \left|\psi_b\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|0\right\rangle + \left|1\right\rangle\left|1\right\rangle\right) \\ f\equiv f_i: &\quad \left|\psi_i\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|1\right\rangle + \left|1\right\rangle\left|0\right\rangle\right). \end{aligned} \tag{2.100}$$

where the values of f for each f_j is tabulated above.

(b) Looking at the general form of $\mathbf{U}_f$ again,

$$\left|\psi\right\rangle = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|f(0)\right\rangle + \left|1\right\rangle\left|f(1)\right\rangle\right), \tag{2.101}$$

we see that a direct measurement of both Qbits reveals the value f takes at either 0 or 1. In particular, suppose we see Qbit 0 in state 0. Then, if we see Qbit 1 in state 0 then we know f is either f_0 or f_b but can't tell which is f is; similarly, if we see Qbit 1 in state 1 then we know f is either f_0 or f_i but can't tell which is f. Repeating this argument for state 1, we see that we cannot distinguish f_1 from f_i and f_1 from f_b. To resolve this problem, we will want to distinguish between the two cases:

$$\textbf{Case 1:}\quad \left|\psi\right\rangle = \left|\psi_0\right\rangle \text{or} \left|\psi_1\right\rangle; \quad \textbf{Case 2:}\quad \left|\psi\right\rangle = \left|\psi_i\right\rangle \text{or} \left|\psi_b\right\rangle \tag{2.102}$$

To this end, we will want to use a $\mathbf{H}^{\otimes 2}$ that appears in the next item.

(c) In this item, we are just verifying that $\mathbf{H}^{\otimes 2}$ transforms the outputs in part (a) into those given in part (c):

$$\begin{aligned}\mathbf{H}^{\otimes 2}\left|\psi_0\right\rangle &= \mathbf{H}^{\otimes 2}\left(\left|+\right\rangle\left|0\right\rangle\right) \\ &= \mathbf{H}\left|+\right\rangle \otimes \mathbf{H}\left|0\right\rangle \\ &= \left|0\right\rangle \otimes \left|+\right\rangle \\ &= \frac{1}{\sqrt{2}}\left(\left|00\right\rangle + \left|01\right\rangle\right) \end{aligned} \tag{2.103}$$

$$\begin{aligned}\mathbf{H}^{\otimes 2}|\psi_1\rangle &= \mathbf{H}^{\otimes 2}(|+\rangle|1\rangle)\\ &= \mathbf{H}|+\rangle \otimes \mathbf{H}|1\rangle\\ &= |0\rangle \otimes |-\rangle\\ &= \frac{1}{\sqrt{2}}(|00\rangle - |01\rangle)\end{aligned} \tag{2.104}$$

$$\begin{aligned}\mathbf{H}^{\otimes 2}|\psi_b\rangle &= \mathbf{H}^{\otimes 2}\left(\frac{1}{\sqrt{2}}(|0\rangle|0\rangle + |1\rangle|1\rangle)\right)\\ &= \frac{1}{\sqrt{2}}(|+\rangle|+\rangle + |-\rangle|-\rangle)\\ &= \frac{2}{2\sqrt{2}}(|00\rangle + |11\rangle)\\ &= \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)\end{aligned} \tag{2.105}$$

$$\begin{aligned}\mathbf{H}^{\otimes 2}|\psi_i\rangle &= \mathbf{H}^{\otimes 2}\left(\frac{1}{\sqrt{2}}(|0\rangle|1\rangle + |1\rangle|0\rangle)\right)\\ &= \frac{1}{\sqrt{2}}(|+\rangle|-\rangle + |-\rangle|+\rangle)\\ &= \frac{2}{2\sqrt{2}}(|00\rangle - |11\rangle)\\ &= \frac{1}{\sqrt{2}}(|00\rangle - |11\rangle).\end{aligned} \tag{2.106}$$

(d) Suppose we measure both Qbits. If f is a constant function, then we will see 00 half the time and 01 half the time. If f is balanced, then we will see 00 half the time and 11 half the time. So, for any f, we will always see 00 half the time. In the other half of the time we will see either 11 or 01, from which we can deduce (with absolute certainty) either f is balanced (if 11) or constant (if 01). So, the probability of success here is $50\% \times 100\% = 50\%$.

(e) Claim: There is no transformation on $|\psi\rangle = \mathbf{U}_f\mathbf{H}_1|00\rangle$ that could enable one to tell whether f is constant or balanced with certainty.

Proof. Suppose (to get a contradiction) that there exists such a (unitary) 2-Qbit transformation. We will call it $\mathbf{U}$. For $\mathbf{U}$ to work, it must be such that each measurement on a Qbit is "meaningful." In particular, $\mathbf{U}$ should not act like $\mathbf{H}$ which allows $|00\rangle$ to appear in all of the outputs, rendering the observation 00 meaningless. This imposes the condition that the computational basis states that appear in $\mathbf{U}|\psi_0\rangle$ and $\mathbf{U}|\psi_1\rangle$ cannot appear in $\mathbf{U}|\psi_b\rangle$ and $\mathbf{U}|\psi_i\rangle$ (and vice versa), because otherwise this "common basis state, $|e\rangle$" will appear as $\mathbf{U}|e\rangle$ output states of both cases of f, making

the probability of successfully discriminating the cases less than 1. We claim, however, that this condition cannot be satisfied for the following reason: The condition above implies that the vector space spanned by the computation basis can be decomposed into a direct sum of orthogonal subspaces $\mathbf{V} \oplus \mathbf{V}^{\perp}$, where $\mathbf{U}\left|\psi_0\right\rangle, \mathbf{U}\left|\psi_1\right\rangle \in \mathbf{V}$, and $\mathbf{U}\left|\psi_i\right\rangle, \mathbf{U}\left|\psi_b\right\rangle \in \mathbf{V}^{\perp}$. It follows that $v \perp v' \, \forall v \in \mathbf{V}, v' \in \mathbf{V}^{\perp}$. In particular, we must have that

$$0 = \langle \mathbf{U}\psi_{1,0} | \mathbf{U}\psi_{i,b} \rangle = \left\langle \mathbf{U}^{\dagger}\mathbf{U}\psi_{1,0} \middle| \psi_{i,b} \right\rangle = \langle \psi_{1,0} | \psi_{i,b} \rangle \tag{2.107}$$

when in fact

$$\langle \psi_{1,0} | \psi_{i,b} \rangle = 1/2. \tag{2.108}$$

So that's a contradiction. Thus, no such $\mathbf{U}$ exists. □

(f) Claim: The probability of correctly determining whether f is constant or balanced (by applying a unitary to $\mathbf{U}_f\mathbf{H}_1\left|00\right\rangle$) is *at most* 50%.

Proof. Suppose we bring in n additional ancillary Qbits. These might be used to transform the input and output registers after the application of $\mathbf{U}_f$ on the Qbits 0 and 1. Let a unitary $\mathbf{W}$ denote this transformation. $\mathbf{W}$ acts on the state space spanned by $n+2$ Qbits. Suppose further (without loss of generality) that these n Qbits are initially in $\left|0\right\rangle_n$ (since any general state $\left|\chi\right\rangle_n$ can be written as $\mathcal{U}\left|0\right\rangle_n$ where $\mathcal{U}$ is some unitary, which we let $\mathbf{W}$ absorb).

Now let the state $\left|\psi\right\rangle = \mathbf{U}_f\mathbf{H}_1\left|00\right\rangle$ be given. Let $\mathbf{W}$ act on $\left|\psi\right\rangle$. The probability of measuring $x_2 \in [0, 2^2)$ for the Qbit pair and $y_n \in [0, 2^n)$ for the n additional Qbits is given by

$$\Pr_{\left|\psi\right\rangle_2}(x_2, y_n) = \left|\left\langle x_2 y_n \right| \mathbf{W} \left|\psi_2, 0_n\right\rangle\right|^2 = 0 \tag{2.109}$$

where the short-hand $\left|\psi_2, 0_n\right\rangle$ denotes $\left|\psi\right\rangle_2 \otimes \left|0\right\rangle_n$. Let any arbitrary pair state $\left|\phi\right\rangle$ given, then

$$\Pr_{\left|\phi\right\rangle_2}(x, y) = 0 \iff \left\langle x_2 y_n \right| \mathbf{W} \left|\phi_2, 0_n\right\rangle = 0. \tag{2.110}$$

Suppose these $\left|\phi\right\rangle$'s span some subspace $\mathbf{V}$ and that $\Pr_{\left|\phi\right\rangle}(x, y) = 0$ for all $\left|\phi\right\rangle$, then linearity says that for any $v \in \mathbf{V}$, $\Pr_v(x, y) = 0$ as well. Now, consider (again) the four possible states of $\mathbf{U}_f\mathbf{H}_1\left|00\right\rangle$:

$$\begin{aligned}
f \equiv f_0: &\quad \left|\psi_0\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|0\right\rangle + \left|1\right\rangle\left|0\right\rangle\right) = \left|+\right\rangle\left|0\right\rangle \\
f \equiv f_1: &\quad \left|\psi_1\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|1\right\rangle + \left|1\right\rangle\left|1\right\rangle\right) = \left|+\right\rangle\left|1\right\rangle \\
f \equiv f_b: &\quad \left|\psi_b\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|0\right\rangle + \left|1\right\rangle\left|1\right\rangle\right) \\
f \equiv f_i: &\quad \left|\psi_i\right\rangle = \mathbf{U}_f\left(\mathbf{H}_1\left|00\right\rangle\right) = \frac{1}{\sqrt{2}}\left(\left|0\right\rangle\left|1\right\rangle + \left|1\right\rangle\left|0\right\rangle\right).
\end{aligned} \tag{2.111}$$

Any measurement that allows us to distinguish with certainty **Case 1** from **Case 2** must have *zero* probability for detecting either (i) any linear combination of $|\psi_0\rangle$ and $|\psi_1\rangle$ OR (ii) any linear combination of $|\psi_i\rangle$ and $|\psi_b\rangle$ (this OR is exclusive). Otherwise, if there's some nonzero probability of detecting some states in both **Case 1** and **Case 2** then we cannot distinguish the cases.

With this, we consider the state

$$|\alpha\rangle = \frac{1}{2}\left(|00\rangle + |01\rangle + |10\rangle + |11\rangle\right). \tag{2.112}$$

We notice that $|\alpha\rangle$ requires the full 2-Qbit computational basis to describe. Any measurement outcomes x, y with $\Pr_{|\alpha\rangle}(x, y) \neq 0$ is meaningless. It follows that the probability of a measurement outcome that *fails* to discriminate **Case 1** from **Case 2** is *at least*:

$$\Pr_{\min} = \sum_{\substack{x,y \\ \Pr_{|\alpha\rangle}(x,y)\neq 0}} \Pr_{|\psi\rangle}(x, y). \tag{2.113}$$

Next, we want to relate $|\psi\rangle$ to $|\alpha\rangle$. It turns out that every one of $|\psi\rangle$ can be written as

$$|\psi\rangle = \frac{1}{\sqrt{2}}\left(|\alpha\rangle + |\beta\rangle\right) \tag{2.114}$$

where $|\beta\rangle \perp |\alpha\rangle$. Explicitly:

$$\begin{aligned}
|\psi_0\rangle &= \frac{1}{\sqrt{2}}\left[|\alpha\rangle + \frac{1}{2}\left(|00\rangle - |01\rangle + |10\rangle - |11\rangle\right)\right] \\
|\psi_1\rangle &= \frac{1}{\sqrt{2}}\left[|\alpha\rangle - \frac{1}{2}\left(|00\rangle - |01\rangle + |10\rangle - |11\rangle\right)\right] \\
|\psi_b\rangle &= \frac{1}{\sqrt{2}}\left[|\alpha\rangle + \frac{1}{2}\left(|00\rangle - |01\rangle - |10\rangle + |11\rangle\right)\right] \\
|\psi_i\rangle &= \frac{1}{\sqrt{2}}\left[|\alpha\rangle - \frac{1}{2}\left(|00\rangle - |01\rangle - |10\rangle + |11\rangle\right)\right]
\end{aligned} \tag{2.115}$$

From here, we have that

$$\begin{aligned}
\Pr_{|\psi\rangle}(x, y) &= \left|\langle x, y| \mathbf{W} |\psi, 0\rangle\right|^2 \\
&= \frac{1}{2}\left|\langle x, y| \mathbf{W} |\alpha, 0\rangle + \langle x, y| \mathbf{W} |\beta, 0\rangle\right|^2 \\
&= \frac{1}{2}\left(\Pr_{|\alpha\rangle}(x, y) + \Pr_{|\beta\rangle}(x, y) + 2\,\mathrm{Re}\left\{\langle \beta, 0| \mathbf{W}^\dagger |x, y\rangle \langle x, y| \mathbf{W} |\alpha, 0\rangle\right\}\right)
\end{aligned} \tag{2.116}$$

where we have used the identity

$$\begin{aligned}|a+b|^2 &= (a+b)\cdot(\overline{a+b}) \\ &= |a|^2 + |b|^2 + \underbrace{a\bar{b} + \bar{a}b} \\ &= |a|^2 + |b|^2 + 2\,\mathrm{Re}\{a\cdot\bar{b}\}. \end{aligned} \tag{2.117}$$

And so we have

$$\begin{aligned}\mathrm{Pr}_{\min} = &\sum_{\substack{x,y \\ \mathrm{Pr}_{|\alpha\rangle}(x,y)\neq 0}} \mathrm{Pr}_{|\psi\rangle}(x,y) \\ = &\frac{1}{2}\sum_{\substack{x,y \\ \mathrm{Pr}_{|\alpha\rangle}(x,y)\neq 0}} \big[\mathrm{Pr}_{|\alpha\rangle}(x,y) + \mathrm{Pr}_{|\beta\rangle}(x,y) \\ &+2\,\mathrm{Re}\left\{\langle\beta,0|\,\mathbf{W}^\dagger\,|x,y\rangle\,\langle x,y|\,\mathbf{W}\,|\alpha,0\rangle\right\}\big] \end{aligned} \tag{2.118}$$

Now, we will extend this sum to all x, y. We notice that

$$\begin{aligned}\sum_{x,y}\mathrm{Pr}_{|\alpha\rangle}(x,y) = &\underbrace{\sum_{\substack{x,y \\ \mathrm{Pr}_{|\alpha\rangle}(x,y)\neq 0}} \mathrm{Pr}_{|\alpha\rangle}(x,y)}_{1} + \underbrace{\sum_{\substack{x,y \\ \mathrm{Pr}_{|\alpha\rangle}(x,y)= 0}} \mathrm{Pr}_{|\alpha\rangle}(x,y)}_{0} \\ = &\,1, \end{aligned} \tag{2.119}$$

and

$$\begin{aligned}2\,\mathrm{Re}\sum_{x,y}\langle\beta,0|\,\mathbf{W}^\dagger\,|x,y\rangle\,\langle x,y|\,\mathbf{W}\,|\alpha,0\rangle &= 2\,\mathrm{Re}\,\langle\beta,0|\,\mathbf{W}^\dagger\mathbf{W}\,|\alpha,0\rangle \\ &= 2\,\mathrm{Re}\,\langle\beta,0|\,\mathbb{I}\,|\alpha,0\rangle \\ &= 2\,\mathrm{Re}\,\langle\beta,0|\alpha,0\rangle \\ &= 0, \end{aligned} \tag{2.120}$$

where we have used the completion property and that $|\alpha\rangle$ and $|\beta\rangle$ are orthogonal. So,

$$\mathrm{Pr}_{\min} = \frac{1}{2}\left(1 + \sum_{\substack{x,y \\ \mathrm{Pr}_{|\alpha\rangle}(x,y)\neq 0}} \mathrm{Pr}_{|\beta\rangle}(x,y)\right) \geq \frac{1}{2}. \tag{2.121}$$

So, the probability of failing is at least 1/2, i.e. the probability of success is at most 1/2.

□

3. The binary dot product. In several problems well see during the rest of the semester we will see the "dot product" $a \cdot b$ appear. This represents the "bitwise" inner product of two n-bit binary numbers a and b

$$a \cdot b \equiv_2 a_{n-1}b_{n-1} + \cdots + a_1 b_1 + a_0 b_0. \tag{2.122}$$

In quantum computer science the dot product is defined modulo-2, which means that it is given by a one-bit number - zero if the dot product is even, and 1 if the dot product is odd.

(a) For two five bit numbers $a = 10101$ and $b = 11011$, what is the dot product $a \cdot b$? How about for $a = 11001$ and $b = 11011$?

(b) Show that the "normal" way (not modulo 2) of calculating the dot product gives the same result for

$$\phi = (-1)^{a \cdot b}. \tag{2.123}$$

Most of the time we use it, the dot product will appear in this form.

(c) Show that the dot product could also be written

$$a \cdot b = a_{n-1}b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \tag{2.124}$$

which is how Mermin writes it.

Solution:

(a) We just apply the formula given in the problem to compute these values:

$$10101 \cdot 11011 \equiv_2 1 \cdot 1 + 0 \cdot 1 + 1 \cdot 0 + 0 \cdot 1 + 1 \cdot 1 \equiv_2 1 + 1 \equiv_2 \boxed{0}$$
$$11001 \cdot 11011 \equiv_2 1 \cdot 1 + 1 \cdot 1 + 0 \cdot 0 + 0 \cdot 1 + 1 \cdot 1 \equiv_2 1 + 1 + 1 \equiv_2 \boxed{1}$$

(b) Let $c = a \cdot b$ (without modulo 2). We know that c odd $\iff$ $c \bmod 2 = 1$. So, $(-1)^c = (-1)^{c \bmod 2}$ for all values of $c = a \cdot b$.

(c) The addition $\oplus$ is addition mod 2. To prove:

$$a \cdot b = a_{n-1}b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \equiv_2 a_{n-1}b_{n-1} + \cdots + a_1 b_1 + a_0 b_0 \tag{2.125}$$

We can do this by induction. Suppose $n = 1$, then $a, b \in \{0, 1\}$. Because $a, b \in \{0, 1\}$, $a \cdot b \equiv_2 ab$. So this base case holds. Next, suppose the statement holds up to n, i.e.,

$$a_{n-1}b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \equiv_2 a_{n-1}b_{n-1} + \cdots + a_1 b_1 + a_0 b_0. \tag{2.126}$$

We want to show it also holds for $n + 1$:

$$a_n b_n \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \equiv_2 a_n b_n + \cdots + a_1 b_1 + a_0 b_0. \tag{2.127}$$

We write the RHS as

$$\begin{aligned} & a_n b_n + (a_{n-1} b_{n-1} + \cdots + a_1 b_1 + a_0 b_0) \\ \equiv_2 \; & a_n b_n + (a_{n-1} b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0) \,. \end{aligned} \tag{2.128}$$

We notice that $a_n, b_n \in \{0, 1\}$ and $a_{n-1} b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \in \{0, 1\}$. By the definition of $\oplus$, we have

$$\begin{aligned} & a_n b_n + (a_{n-1} b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0) \\ \equiv_2 \; & a_n b_n \oplus (a_{n-1} b_{n-1} \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0) \\ = \; & \text{LHS}. \end{aligned} \tag{2.129}$$

So,

$$a_n b_n \oplus \cdots \oplus a_1 b_1 \oplus a_0 b_0 \equiv_2 a_n b_n + \cdots + a_1 b_1 + a_0 b_0 \tag{2.130}$$

as desired. □

4. Walsh-Hadamard. A question that will come up in algorithms is what the outcome is when an n-Qbit Hadamard gate (sometimes called the Walsh-Hadamard gate $\mathbf{H}^{\otimes n}$ is applied to a general quantum state.

In class we saw that

$$\mathbf{H}^{\otimes n} |0\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} |y\rangle_n \tag{2.131}$$

where y is one of the 2^n n-Qbit eigenstates in the computational basis. The goal of this problem is to show that

$$\mathbf{H}^{\otimes n} |x\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} (-1)^{x \cdot y} |y\rangle_n \tag{2.132}$$

where x is a single eigenstate in the computational basis.

(a) Show that for a 1-Qbit state $|x\rangle$ you can write

$$\mathbf{H} |x\rangle = \frac{1}{\sqrt{2}} \sum_{0 \le y < 2} (-1)^{x \cdot y} |y\rangle \tag{2.133}$$

for $x, y \in \{0, 1\}$.

(b) Show further that for a 2-Qbit state $|x\rangle_2 = |x_1\rangle |x_0\rangle$ that

$$\mathbf{H}^{\otimes 2} |x_1 x_0\rangle = \frac{1}{2} \left[|00\rangle + (-1)^{x_0} |01\rangle + (-1)^{x_1} |10\rangle + (-1)^{x_1 + x_0} |11\rangle \right] \tag{2.134}$$

(c) Show that for you can write the result of the last pair as

$$\mathbf{H}^{\otimes 2} |x_1 x_0\rangle = \frac{1}{\sqrt{2^2}} \sum_{0 \le y < 2^2} (-1)^{x \cdot y} |y\rangle \tag{2.135}$$

for $x, y \in \{0, 1\}^2$.

(d) Show that, as stated above, for n-Qbit states $|x\rangle_n$ that

$$\mathbf{H}^{\otimes n} |x\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0 \le y < 2^n} (-1)^{x \cdot y} |y\rangle_n . \tag{2.136}$$

Solution:

(a) Let a 1-Qbit eigenstate $|x\rangle$ be given. If $|x\rangle = |0 \text{ or } 1\rangle$ then

$$\begin{aligned}
\mathbf{H}|x\rangle &= |\pm\rangle \\
&= \frac{1}{\sqrt{2}}\left(|0\rangle \pm |1\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left((-1)^{(0 \text{ or } 1)\cdot 0}|0\rangle + (-1)^{(0 \text{ or } 1)\cdot 1}|1\rangle\right) \\
&= \frac{1}{\sqrt{2}}\left((-1)^{x\cdot 1}|0\rangle + (-1)^{x\cdot 1}|1\rangle\right) \\
&= \frac{1}{\sqrt{2}}\sum_{0\le y<2}(-1)^{x\cdot y}|y\rangle\,. \qquad (2.137)
\end{aligned}$$

(b) We use the result from (a) to show the claim in (b):

$$\begin{aligned}
\mathbf{H}^{\otimes 2}|x_1x_0\rangle &= \mathbf{H}|x_1\rangle \otimes \mathbf{H}|x_0\rangle \\
&= \frac{1}{2}\left(\sum_{0\le y<2}(-1)^{x_1\cdot y}|y\rangle\right)\otimes\left(\sum_{0\le y<2}(-1)^{x_0\cdot y}|y\rangle\right) \\
&= \frac{1}{2}\Big((-1)^{(x_1+x_0)\cdot 0}|00\rangle + (-1)^{x_1\cdot 0+x_0\cdot 1}|01\rangle \\
&\quad +(-1)^{x_1\cdot 1+x_0\cdot 0}|10\rangle + (-1)^{(x_1+x_0)\cdot 1}|11\rangle\Big) \qquad (2.138) \\
&= \frac{1}{2}\left[|00\rangle + (-1)^{x_0}|01\rangle + (-1)^{x_1}|10\rangle + (-1)^{x_1+x_0}|11\rangle\right] \qquad (2.139)
\end{aligned}$$

(c) Clearly,

$$\begin{aligned}
x_1\cdot 0 + x_0\cdot 0 = x\cdot y\Big|_{|y\rangle=|00\rangle}, \quad x_1\cdot 0 + x_0\cdot 1 = x\cdot y\Big|_{|y\rangle=|01\rangle} \\
x_1\cdot 1 + x_0\cdot 0 = x\cdot y\Big|_{|y\rangle=|10\rangle}, \quad x_1\cdot 1 + x_0\cdot 1 = x\cdot y\Big|_{|y\rangle=|11\rangle} \qquad (2.140)
\end{aligned}$$

So, we just rewrite the third equality in part (b) as

$$\begin{aligned}
&\frac{1}{\sqrt{2^2}}\left((-1)^{x\cdot 00}|00\rangle + (-1)^{x\cdot 01}|01\rangle + (-1)^{x\cdot 10}|10\rangle + (-1)^{x\cdot 11}|11\rangle\right) \\
=&\frac{1}{\sqrt{2^2}}\sum_{0\le y<2^2}(-1)^{x\cdot y}|y\rangle\,. \qquad (2.141)
\end{aligned}$$

(d) Now we want to generalize to n. We show that the given statement holds, by induction. In parts (a) to (c), we have shown that the base cases hold. Now, suppose the statement holds up to n, i.e.,

$$\mathbf{H}^{\otimes n}|x\rangle_n = \frac{1}{\sqrt{2^n}}\sum_{0\le y<2^n}(-1)^{x\cdot y}|y\rangle_n\,. \qquad (2.142)$$

We want to show it also holds for $n+1$. So, to prove:

$$\mathbf{H}^{\otimes(n+1)} \left|x\right\rangle_{n+1} = \frac{1}{\sqrt{2^{n+1}}} \sum_{0\leq y<2^{n+1}} (-1)^{x\cdot y} \left|y\right\rangle_{n+1}. \tag{2.143}$$

We first look at the LHS. Let $\left|x\right\rangle_{n+1} = \left|x_n\right\rangle \otimes \left|x_{n-1}\ldots x_0\right\rangle_n$ with $x_i \in \{0,1\}$, then we have that

$$\begin{aligned}
\mathbf{H}^{\otimes(n+1)} \left|x\right\rangle_{n+1} &= \mathbf{H}\left|x_n\right\rangle \otimes \mathbf{H}^{\otimes n} \left|x_{n-1}\ldots x_0\right\rangle_n \equiv \mathbf{H}\left|x_n\right\rangle \otimes \mathbf{H}^{\otimes n} \left|b\right\rangle_n \\
&= \left(\frac{1}{\sqrt{2}} \sum_{0\leq y<2} (-1)^{x_n\cdot y} \left|y\right\rangle\right) \otimes \left(\frac{1}{\sqrt{2^n}} \sum_{0\leq y'<2^n} (-1)^{b\cdot y'} \left|y'\right\rangle_n\right) \\
&= \frac{1}{\sqrt{2^{n+1}}} \sum_{\substack{0\leq y<2 \\ 0\leq y'<2^n}} (-1)^{x_n\cdot y + b\cdot y'} \left|yy'\right\rangle_{n+1}.
\end{aligned} \tag{2.144}$$

It is not difficult to see that for any $0 \leq y < 2^{n+1}$

$$x_0\cdot y + b\cdot y' \equiv_2 = x_n y_n + x_{n-1}y_{n-1} + \cdots + x_0 y_0 = x\cdot y. \tag{2.145}$$

And so we have

$$\frac{1}{\sqrt{2^{n+1}}} \sum_{\substack{0\leq y<2 \\ 0\leq y'<2^n}} (-1)^{x_n\cdot y + b\cdot y'} \left|yy'\right\rangle_{n+1} = \frac{1}{\sqrt{2^{n+1}}} \sum_{0\leq y<2^{n+1}} (-1)^{x\cdot y} \left|y\right\rangle_{n+1}. \tag{2.146}$$

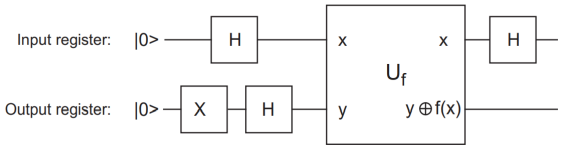
Thus, we have proven the equality:

$$\mathbf{H}^{\otimes n} \left|x\right\rangle_n = \frac{1}{\sqrt{2^n}} \sum_{0\leq y<2^n} (-1)^{x\cdot y} \left|y\right\rangle_n. \tag{2.147}$$

□

2.7 Problem set 7

1. Phase Kickback and the Deutsch Problem. The 100% solution to the Deutsch problem, the Deutsch-Jozsa problem, and the Bernstein-Vazirani problem (the first we discussed on March 12th, and the second and third will be discussed for the next class) all use the clever approach of "phase kickback" to get an answer to the question. This problem is designed to have you walk through the application of phase kickback in the Deutsch problem. This problem has a 1-Qbit input register and a 1-Qbit output register.



(a) What is the state of the system just before the operator $\mathbf{U}_f$?

(b) Determine, using what you know about phase kickback, the state of the system just after the application of $\mathbf{U}_f$ (and before the final Hadamard) given that you don't know which of the four 1-bit to 1-bit functions has been applied.

(c) The four 1-bit to 1-bit functions are defined by the following table.

x	$f_0(x)$	$f_1(x)$	$f_b(x)$	$f_i(x)$
0	0	1	0	1
1	0	1	1	0

What is the state of the system after the application of $\mathbf{U}_f$ (and before the final Hadamard) for each of the four functions? What's the difference between constant and balanced functions?

(d) What is the state of the system after the final Hadamard gate for each of the four possible 1-bit to 1-bit functions $\mathbf{U}_f$? How can you distinguish constant and balanced functions?

Solution:

(a) Just before $\mathbf{U}_f$, the state of the system is given by

$$\mathbf{H}|0\rangle \otimes \mathbf{HX}|0\rangle = |+-\rangle \tag{2.148}$$

since $\mathbf{X}$ flips $|0\rangle$ to $|1\rangle$ and $\mathbf{H}|1\rangle = |-\rangle$.

(b) We did this in (physical) class. The formula for the output immediately after the application of $\mathbf{U}_f$, provided in the input in the problem is

$$\boxed{|\psi\rangle = \frac{1}{\sqrt{2}} \sum_{0 \le x < 2} (-1)^{f(x)} |x\rangle \otimes |-\rangle} \tag{2.149}$$

(c) We do this one by one:

$$\begin{aligned}
f = f_0 : |\psi\rangle &= \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)|-\rangle = |+-\rangle \\
f = f_1 : |\psi\rangle &= \frac{1}{\sqrt{2}}(-|0\rangle - |1\rangle)|-\rangle = -|+-\rangle \\
f = f_b : |\psi\rangle &= \frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)|-\rangle = |--\rangle \\
f = f_i : |\psi\rangle &= \frac{1}{\sqrt{2}}(-|0\rangle + |1\rangle)|-\rangle = -|--\rangle
\end{aligned} \tag{2.150}$$

Constant functions return Qbits with different handedness, while balanced functions return Qbits with the same handedness. More importantly, constant and balanced functions produce different handedness for Qbit 1. With the Hadamard, we will be able to distinguish these handedness and hence the balanced/constant nature of f.

(d) If the output immediately after $\mathbf{U}_f$ is $|+-\rangle$ then the final output is $\mathbf{H}_1|+-\rangle = |01\rangle$. If the output immediately after $\mathbf{U}_f$ is $|--\rangle$ then the final output is $\mathbf{H}_1|--\rangle = |11\rangle$. By measuring Qbit 1, we can distinguish constant and balanced functions. If Qbit 1 is in state 0 then f is constant, else f is balanced.

□

2. Consider the output of the Deutsch-Josza problem where the oracle encodes a function $f : \{0,1\}^n \to \{0,1\}$ which is known to be either balanced or constant. As shown in the Screencast, after the final set of Hadamards, the state of the input register is

$$\begin{aligned} |\psi_i\rangle &= \frac{1}{2^n} \sum_{0 \le x < 2^n} (-1)^{f(x)} \sum_{0 \le y < 2^n} (-1)^{x \cdot y} |y\rangle_n \\ &= \frac{1}{2^n} \sum_{0 \le y < 2^n} \sum_{0 \le x < 2^n} (-1)^{f(x)} (-1)^{x \cdot y} |y\rangle_n . \end{aligned} \tag{2.151}$$

(a) Show, following the argument made in the Screencast, that the state of the input register $|\psi_i\rangle$ for constant functions leads to a 100% probability of finding the state in the state $|0\rangle_n$.

(b) Show, following the argument made in the Screencast, that the state of the input register $|\psi_i\rangle$ for balanced functions leads to a 0% probability of finding the state in the state $|0\rangle_n$.

(c) Argue, extending the argument made in the Screencast, that the state of the input register $|\psi_i\rangle$ for constant functions leads to a 0% probability of finding the state in some state $|y\rangle_n$ for $y \neq 0$.

(d) Argue, extending the argument made in the Screencast, that the state of the input register $|\psi_i\rangle$ for balanced functions leads to a nonzero probability of finding the state in the state $|y\rangle_n$ for $y \neq 0$.

Solution:

(a) Suppose f is constant. Then the terms $(-1)^{f(x)}$ in the sum over x are either all zeros or all ones. This mean we can rewrite the state of the input register as

$$|\psi_i\rangle = \pm \frac{1}{2^n} \sum_{0 \le y < 2^n} \sum_{0 \le x < 2^n} (-1)^{x \cdot y} |y\rangle_n . \tag{2.152}$$

Next, we look at the $|0\rangle_n$ term in the sum. Its amplitude is

$$\frac{\pm 1}{2^n} \sum_{0 \le x < 2^n} (-1)^{x \cdot 0} = \frac{\pm 1}{2^n} \sum_{0 \le x < 2^n} 1 = \frac{\pm 1}{2^n} 2^n = \pm 1. \tag{2.153}$$

And so there is a 100% probability of finding the state in $|0\rangle_n$. Of course, this leaves zero probability to finding $|\psi_i\rangle$ in any other state.

(b) Suppose f is balanced. Then the terms $(-1)^{f(x)}$ in the sum over x are either zeros exactly half the time. Consider the summands with $|00\rangle$. This term looks like

$$\frac{1}{2} \sum_{0 \le x < 2^n} (-1)^{f(x)} (-1)^{x \cdot 0} |0\rangle_n = \frac{1}{2} \sum_{0 \le x < 2^n} (-1)^{f(x)} |0\rangle_n \tag{2.154}$$

Since $(-1)^{f(x)}$ is 1 exactly half the time and -1 exactly half the time, the amplitude of the state $|0\rangle_n$ is zero. So, the probability that $|\psi_i\rangle$ is in $|0\rangle_n$ is zero.

(c) Clearly, when the probability that $|\psi_i\rangle$ is in state $|0\rangle_n$ is unity, the probability that $|\psi_i\rangle$ is in any other state is necessarily zero.

(d) Similarly, when the probability that $|\psi_i\rangle$ is in state $|0\rangle_n$ is zero, the probability that $|\psi_i\rangle$ is in any other state is not necessarily zero.

□

3. A quantum decoder. Note: This problem is based on an idea from a lecture by John Watrous, who was at the University of Calgary when he wrote it, but is now at the University of Waterloo and is a member of the Institute for Quantum Computing.

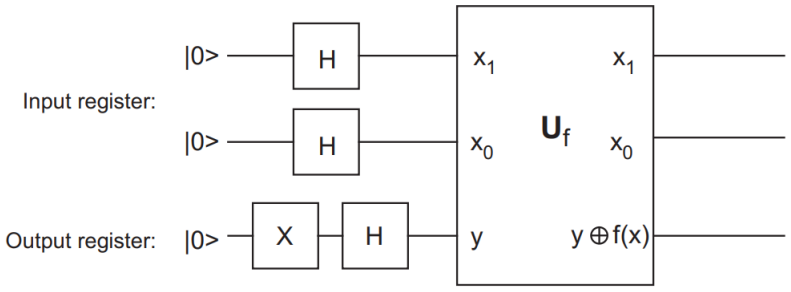
In electronics a decoder is a device that sets an output to one for a specific pair of inputs. In the language of quantum computation, an output of a decoder is an oracle that encodes a function of the form $f : \{0,1\}^2 \rightarrow \{0,1\}$.

There are four possible decoders, tabulated below.

x	$f_{00}(x)$	$f_{01}(x)$	$f_{10}(x)$	$f_{11}(x)$
00	1	0	0	0
01	0	1	0	0
10	0	0	1	0
11	0	0	0	1

If you didn't know which decoder you had, you would need to query the decoder three times to be sure of what one you had. With phase kickback you can do it in one query.

Consider a device $\mathbf{U}_f$ that encodes one of the four decoders, using the phase kickback approach as shown in the quantum circuit diagram below.

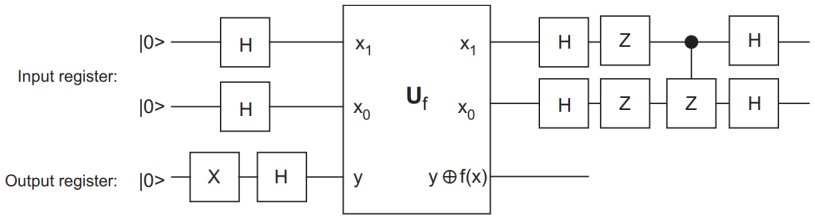


(a) What is the state of the system just before the operator $\mathbf{U}_f$?

(b) Determine, using what you know about phase kickback, the state of the system just after the application of $\mathbf{U}_f$ given that you don't know which of the decoder circuits has been implemented.

(c) Explain why the output for each of the four decoders is an equal superposition of the four states $(|00\rangle, |01\rangle, |10\rangle, |11\rangle)$ with the phase of the "marked" state being -1, while all the rest have a phase +1.

(d) These states are all orthogonal entangled states (but not the Bell states). Since they are orthogonal, if you (or John Watrous) are clever you can create a quantum circuit that disentangle them and put them in basis states of the computational basis. Show that for the function f_{00} that circuit below leads to the state $|\psi(f_{00})\rangle = |00\rangle\,|-\rangle$.



(e) (Optional, continuation). Show that the same circuit serves to put the state into the appropriate state $|\psi(f_{ij})\rangle = |ij\rangle\,|-\rangle$.

Solution:

(a) The state of the system just before entering $\mathbf{U}_f$ is

$$\mathbf{H}\otimes\mathbf{H}\otimes\mathbf{HX}\,|000\rangle = \mathbf{H}\,|0\rangle\otimes\mathbf{H}\,|0\rangle\otimes\mathbf{H}\,|1\rangle = |++-\rangle\,. \tag{2.155}$$

(b) The function $\mathbf{U}_f$ is defined by

$$\mathbf{U}_f\,|x_1x_0y\rangle = |x_1x_0\rangle\,|y\oplus f(x_1x_0)\rangle\,. \tag{2.156}$$

Given the input $\mathbf{H}^{\otimes n}\,|0\rangle_n\otimes(\mathbf{HX})\,|0\rangle$ the function $\mathbf{U}_f$ defined above, the output is

$$\mathbf{U}_f\left\{\mathbf{H}^{\otimes n}\,|0\rangle_n\otimes(\mathbf{HX})\,|0\rangle\right\} = \frac{1}{\sqrt{2^n}}\sum_{0\le x<2^n}(-1)^{f(x)}\,|x\rangle_n\otimes|-\rangle\,. \tag{2.157}$$

In class we did an example at $n=1$, but it is not hard to see how the example generalizes. The key is the identity: $|0\oplus a\rangle = -\,|1\oplus a\rangle = (-1)^{f(a)}(|0\rangle - |1\rangle)$. The factor $(|0\rangle - |1\rangle)$ ultimately factors out as a $|-\rangle$ in the formula.

Okay, with this, setting $n=2$ (2 input Qbits) we see what the output immediately after $\mathbf{U}_f$ is:

$$\boxed{\frac{1}{\sqrt{2^2}}\sum_{0\le x<2^2}(-1)^{f(x)}\,|x\rangle_2\otimes|-\rangle} \tag{2.158}$$

(c) We can do this in general for any n, not just for the $n=2$ case. Say the marked state is $x=\alpha$, then it follows that $f_\alpha(x)=1$ if and only if $x=\alpha$, and (of course) for 2^n-1 other times $f_\alpha(x)=0$. Look at the output immediately after $\mathbf{U}_f$ again:

$$\frac{1}{\sqrt{2^n}}\sum_{0\le x<2^n}(-1)^{f(x)}\,|x\rangle_n\otimes|-\rangle\,. \tag{2.159}$$

We can write this sum as

$$\begin{aligned}&\frac{1}{\sqrt{2^n}}(-1)^{f_\alpha(\alpha)}\left|\alpha\right\rangle\otimes\left|-\right\rangle+\frac{1}{\sqrt{2^n}}\sum_{\substack{0\leq x<2^n\\x\neq\alpha}}(-1)^{f_\alpha(x)}\left|x\right\rangle_n\otimes\left|-\right\rangle\\=&\frac{1}{\sqrt{2^n}}(-1)\left|\alpha\right\rangle\otimes\left|-\right\rangle+\frac{1}{\sqrt{2^n}}\sum_{\substack{0\leq x<2^n\\x\neq\alpha}}(1)\cdot\left|x\right\rangle_n\otimes\left|-\right\rangle.\end{aligned}\tag{2.160}$$

So, we see that not only this is an equal superposition of all computational basis states, it is also one where the marked state $|\alpha\rangle$ carries a phase of (-1), while the other states $+1$. And to get an exact answer for item (c), we can just let $n = 2$.

(d) Assume $f = f_{00}$. Then the output immediately after $\mathbf{U}_f$ is given by

$$\frac{1}{\sqrt{2^2}}\sum_{0\leq x<2^2}(-1)^{f_{00}(x_1x_0)}\left|x_1x_0\right\rangle\otimes\left|-\right\rangle.\tag{2.161}$$

Since all operators act on the two input Qbits, we can just ignore that output Qbit for now. The evaluation is straightforward (provided no malicious typos appear)

$$\begin{aligned}&\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\mathbf{Z}^{\otimes2}\mathbf{H}^{\otimes2}\frac{1}{\sqrt{2^2}}\sum_{0\leq x<2^2}(-1)^{f_{00}(x_1x_0)}\left|x_1x_0\right\rangle\\=&\frac{1}{2}\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\mathbf{Z}^{\otimes2}\mathbf{H}^{\otimes2}\left[-\left|00\right\rangle+\left|01\right\rangle+\left|10\right\rangle+\left|11\right\rangle\right]\\=&\frac{1}{2}\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\mathbf{Z}^{\otimes2}\left[-\left|{+}{+}\right\rangle+\left|{+}{-}\right\rangle+\left|{-}{+}\right\rangle+\left|{-}{-}\right\rangle\right]\\=&\frac{1}{2}\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\left[-\left|{-}{-}\right\rangle+\left|{-}{+}\right\rangle+\left|{+}{-}\right\rangle+\left|{+}{+}\right\rangle\right]\\=&\frac{1}{2}\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\left[\sqrt{2}\left|1\right\rangle\left|-\right\rangle+\sqrt{2}\left|0\right\rangle\left|+\right\rangle\right]\\=&\frac{1}{\sqrt{2}}\mathbf{H}^{\otimes2}\mathbf{C}_{21}^{\mathbf{Z}}\left[\left|1\right\rangle\left|-\right\rangle+\left|0\right\rangle\left|+\right\rangle\right]\\=&\frac{1}{\sqrt{2}}\mathbf{H}^{\otimes2}\left[\left|1\right\rangle\left|+\right\rangle+\left|0\right\rangle\left|+\right\rangle\right]\\=&\mathbf{H}^{\otimes2}\left|{+}{+}\right\rangle\\=&\left|00\right\rangle.\end{aligned}\tag{2.162}$$

And so the final state of the system is $|00\rangle\,|-\rangle$.

(e) How do we do this in general? Since there are only 3 more cases, it's worth doing this by exhaustion. We already know what happens when $i = j = 0$. Now consider $i = 0, j = 1$. In this case, $|01\rangle$ will carry a -1 phase, which turns the seventh line to $\sim \mathbf{H}^{\otimes2}\left[-\left|1\right\rangle\left|-\right\rangle+\left|0\right\rangle\left|-\right\rangle\right] \sim \mathbf{H}^{\otimes2}\left|{+}{-}\right\rangle = \left|01\right\rangle$.

When $i = j = 1$, the state $|11\rangle$ carries the -1 phase. This turns the sixth line $\mathcal{O}\left[|0\rangle|-\rangle - |1\rangle|+\rangle\right]$ where $\mathcal{O}$ is a combination of operators and scalars. Applying $\mathbf{C}_{21}^{\mathbf{Z}}$, followed by $\mathbf{H}^{\otimes}$, gives $\mathbf{H}^{\otimes 2}|--\rangle = |11\rangle$. By elimination, we know when $i = 1, j = 0$, the output is going to be $|10\rangle$. But we can also check (why not?). When $i = 1, j = 0$, the term $|10\rangle$ carries a -1 phase. This result in the sixth line being $\sim \mathcal{O}[|0\rangle|+\rangle + |1\rangle|-\rangle]$. Applying $\mathbf{C}_{21}^{\mathbf{Z}}$ followed by $\mathbf{H}^{\otimes 2}$ gives $\mathbf{H}^{\otimes 2}\left(|0\rangle|+\rangle - |1\rangle|+\rangle\right) \sim \mathbf{H}^{\otimes 2}|-+\rangle = |01\rangle$.

To summarize:

	$j=0$	$j=1$
$i=0$	$\to \|00\rangle$	$\to \|01\rangle$
$i=1$	$\to \|10\rangle$	$\to \|11\rangle$

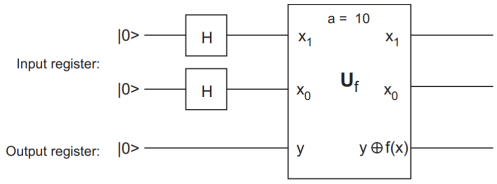
We conclude that $|\psi(f_{ij})\rangle = |ij\rangle|-\rangle$.

□

4. Bernstein-Vazirani without Phase Kickback. The idea of this problem is to look again at the Bernstein-Vazirani problem without using phase kickback as an introduction to the techniques used for Simons problem.

Recall that in the Bernstein-Vazirani problem you are given a function $f : \{0,1\}^n \to \{0,1\}$ that encodes a hidden bit-string a such that $f(x) = a \cdot x \pmod 2$ for any input string x.

Consider the circuit below that implements a 2-Qbit version of the Bernstein-Vazirani problem with the string $a = 10$.



(a) What is the state of the system after the application of $\mathbf{U}_f$?

(b) Are the input and output registers entangled? How do you know?

(c) Imagine that you measured the output register and found that the output register was in the state $|1\rangle$. What is the state of the system after that (partial) measurement?

(d) Look at the solution you got, and note that the states $|x_i\rangle$ of the input register correspond to states with $a \cdot x_i = 1$.

(e) Imagine that you measured the output register and found that the output register was in the state $|0\rangle$. What is the state of the system after that (partial) measurement?

(f) Look at the solution you got, and note that the states $|x_j\rangle$ of the input register correspond to states with $a \cdot x_j = 0$. Of course if you measured the output you would only find one of the states.

(g) Now, after measuring y and x this way a couple of times you would have a set of equations $a \cdot x = 0$ or $a \cdot x = 1$ for different x values. Convince yourself that you could use these values to figure out a. (Unless you are unlucky and get $x = 0$, which tells you nothing, because $a \cdot 0 = 0$ for all a.)

Note: This approach is no more efficient than a classical algorithm would be if you randomly chose input values. The importance of this approach using multiple queries and then using the resulting classical answers to determine the behavior of the oracle was recognized by Dan Simon for a function $f : \{0,1\}^n \to \{0,1\}^n$.

Solution:

(a) The state of the system immediately after the application of $\mathbf{U}_f$ is

$$\begin{aligned}\mathbf{U}_f(\mathbf{H}_2\mathbf{H}_1\left|000\right\rangle) &= \mathbf{U}_f\left|++0\right\rangle \\ &= \frac{1}{2}\mathbf{U}_f\left[\left|00\right\rangle+\left|01\right\rangle+\left|10\right\rangle+\left|11\right\rangle\right]\otimes\left|0\right\rangle \\ &= \frac{1}{2}\left[\left|00f(00)\right\rangle+\left|01f(01)\right\rangle+\left|10f(10)\right\rangle+\left|11f(11)\right\rangle\right] \\ &= \frac{1}{2}\left[\left|000\right\rangle+\left|010\right\rangle+\left|101\right\rangle+\left|111\right\rangle\right] \\ &= \frac{1}{\sqrt{2}}\left|0+0\right\rangle+\frac{1}{\sqrt{2}}\left|1+1\right\rangle. \end{aligned} \tag{2.163}$$

where $f(x)\equiv_2 a\cdot x$.

(b) The state of the system immediately after the application of $\mathbf{U}_f$ is entangled. In particular, the Qbit in the output register is entangled with the top input Qbit. Whenever the Qbit in the output register is in state i, the top Qbit in the input register is also in state i.

(c) Suppose the state of the output register is $\left|1\right\rangle$. Then the state of the input register is $\left|1+\right\rangle$. The state of the system as a result of the measurement is $\left|1+1\right\rangle$.

(d) Suppose the output Qbit is in state $\left|1\right\rangle$. The input register is in the states $\left|10\right\rangle$ and $\left|11\right\rangle$. Clearly, $a\cdot x_i = 10\cdot 11 = 10\cdot 10 = 1$.

(e) Suppose the output Qbit is in state $\left|0\right\rangle$. The state of the system after that measurement collapses to $\left|0+0\right\rangle$.

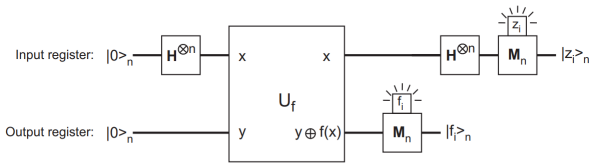
(f) Whenever the output Qbit is in state $\left|0\right\rangle$, the input register is in states $\left|01\right\rangle$ and $\left|00\right\rangle$. So, $a\cdot x_j = 10\cdot 01 = 10\cdot 00 = 0$.

(g) In the context of this problem, $a\cdot x = 0$ whenever $x=0$ or $x=1$, and $a\cdot x = 1$ whenever $x=2$ or $x=3$. When $x=0$, we don't learn anything (but this is fine because when size of the vector space grows exponentially as the problem gets bigger, this possibility is small). When $x=1$, because $a\cdot 1 = 0$ we know that $a = a_1a_0 = a_10$. Next, because $a\cdot 2 = a\cot 3 = 1$, $a_1 = 1$. This gives $a = 10$.

□

2.8 Problem set 8

1. A Simon's Algorithm Primer. You are given a unitary operator $\mathbf{U}_f$ that encodes a function $f : \{0,1\}^n \to \{0,1\}^n$ for a 3-bit input ($n = 3$) which has the characteristics $f(x) = f(x \oplus a)$ for an unknown a – but it is known that $a \neq 0$.



The outputs of the specific function $f(x)$ for the eight possible inputs are tabulated below.

	x	$f(x)$
0	000	110
1	100	000
2	010	001
3	011	101
4	100	000
5	101	110
6	110	101
7	111	001

(a) Use a classical algorithm (look for collisions) and determine the hidden string a.

For the rest of this problem, consider the quantum algorithm for solving Simon's problem shown in the circuit diagram above.

(b) What is the state of the system $|\psi_1\rangle$ just before the oracle?

(c) What is the state of the system $|\psi_2\rangle$ just after the oracle?

(d) What is the state of the system ψ_3 just after a measurement of the output register gave the value $f_0 = 001$?

(e) What is the state of the system $|\psi_4\rangle$ after the final set of Hadamard operations on the input register (and, of course, after the measurement of $f_0 = 001$)?

(f) Show that each component of the input register $|z_i\rangle$ has the property $z_i \cdot a = 0$ (using the value of a you found in part (a)) and that the states $|z_j\rangle$ **not** included in the superposition have the property $z_j \cdot a = 1$.

(g) Go back a step and consider a second query of the oracle. Since the input $|\psi_1\rangle$ is the same as the output of the oracle $|\psi_2\rangle$ is also the same. What is the state of the system $|\psi_3\rangle$ just after the measurement of the output register gave the value $f_1 = 101$?

(h) What is the state of the system $|\psi_4\rangle$ after the final set of Hadamard operations on the input register (and, of course, after the measurement of $f_1 = 101$)? Compare the input register state with the one in part (e).

(i) Suppose that after your two queries of the oracle, measurements of the input register gave $z_0 = 010$ and $z_1 = 111$. Show, that (if is known that $a \neq 000$) these two measurements determine the value of a. If it's not known that $a \neq 000$ then $a = 0$ is a solution as well no matter what the values of z are.

Solution:

(a) We know that f is a 2 to 1 function. So, it suffices to look at two inputs $x_0 \neq y_0$ such that $f(x_0) = f(y_0)$. In this case, we necessarily have $x_0 = y_0 \oplus a$. Since we're working over the finite field $\mathbb{F}^{2n}$, we know that

$$\begin{aligned} & x_0 = y_0 \oplus a \\ \iff & x_0 \oplus y_0 = y_0 \oplus y_0 \oplus a, \quad \text{addition is commutative} \\ \iff & x_0 \oplus y_0 = a. \end{aligned} \tag{2.164}$$

From the table, we know that 001 and 100 give the same output 000. And so $a = 001 \oplus 100 = 101$. So, $\boxed{a = 101}$. As a sanity check, $000 \oplus 101 = 010 \oplus 111 \oplus 011 \oplus 110 = 101$ as well.

(b) Just before the oracle, the state of the system is

$$\begin{aligned} |\psi_1\rangle &= (\mathbf{H}^{\otimes n} \otimes \mathbb{I}^{\otimes n}) \, |0\rangle_n \, |0\rangle_n \\ &= \frac{1}{\sqrt{2^n}} \sum_{0 \leq x < 2^n} |x\rangle_n \, |0\rangle_n \\ &= \frac{1}{\sqrt{2^3}} \sum_{0 \leq x < 2^3} |x\rangle_3 \, |0\rangle_3 \,, \quad n = 3. \end{aligned} \tag{2.165}$$

(c) The state of the system just after the oracle is

$$\begin{aligned} |\psi_2\rangle &= \mathbf{U}_f \, |\psi_1\rangle \\ &= \mathbf{U}_f \frac{1}{\sqrt{2^3}} \sum_{0 \leq x < 2^3} |x\rangle_3 \, |0\rangle_n \\ &= \frac{1}{\sqrt{2^3}} \sum_{0 \leq x < 2^3} |x\rangle_3 \, |f(x)\rangle_3 \,. \end{aligned} \tag{2.166}$$

(d) If $x_0 = 001$, then after the measurement the state of the system must carry $|001\rangle$ in the output register. Since f is 2-to-1, there are two states with the output register in $|001\rangle$. So,

$$|\psi_3\rangle = \frac{1}{\sqrt{2}} \left[|010\rangle + |111\rangle\right] |001\rangle \,. \tag{2.167}$$

(e) To find the state after Hadamard-ing the input register, we just go through the straightforward computation:

$$\begin{aligned} |\psi_4\rangle &= \mathbf{H}^{\otimes 3} \otimes \mathbb{I}^{\otimes 3} \frac{1}{\sqrt{2}} \left[|010\rangle + |111\rangle\right] |001\rangle \\ &= \frac{1}{\sqrt{2}} \left[|+-+\rangle + |---\rangle\right] |001\rangle \\ &= \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2^3}} \left[|000\rangle - |010\rangle + |101\rangle - |111\rangle\right] |001\rangle \,. \end{aligned} \tag{2.168}$$

(f) We can see easily that each component-state z_i of the input register is such that $z_i \cdot a = 0$:

$$\begin{aligned} 000 \cdot 101 &= 0 \oplus 0 \oplus 0 = 0 \\ 010 \cdot 101 &= 0 \oplus 0 \oplus 0 = 0 \\ 101 \cdot 101 &= 0 \oplus 0 \oplus 0 = 0 \\ 111 \cdot 101 &= 0 \oplus 0 \oplus 0 = 0. \end{aligned} \tag{2.169}$$

We can also see states that are not those above have the property that $z_j \cdot a = 1$:

$$\begin{aligned} 001 \cdot 101 &= 0 \oplus 0 \oplus 1 = 1 \\ 011 \cdot 101 &= 0 \oplus 0 \oplus 1 = 1 \\ 100 \cdot 101 &= 1 \oplus 0 \oplus 0 = 1 \\ 110 \cdot 101 &= 1 \oplus 0 \oplus 0 = 1. \end{aligned} \tag{2.170}$$

(g) If $f_1 = 101$, then $|\psi_3\rangle \sim \left[|011\rangle + |110\rangle\right] |101\rangle$. Normalizing this gives

$$|\psi_3\rangle = \frac{1}{\sqrt{2}} \left[|011\rangle + |110\rangle\right] |101\rangle \,. \tag{2.171}$$

(h) Hadamard-ing to the input register we get $|\psi_4\rangle \sim \left[|+--\rangle + |--+\rangle\right] |101\rangle \sim \left[|000\rangle - |010\rangle - |101\rangle + |111\rangle\right] |101\rangle$. We see that the components of the input register in this case is the same as before. Of course, after normalization,

$$|\psi_4\rangle = \frac{1}{\sqrt{2}} \frac{1}{\sqrt{2^3}} \left[|000\rangle - |010\rangle - |101\rangle + |111\rangle\right] |101\rangle \,. \tag{2.172}$$

(i) Let $a = a_2a_1a_0 \neq 000$. If measurements of the input register gives $z_0 = 010$ and $z_1 = 111$, then we know that $0 = 010 \cdot a_2a_1a_0 = 111 \cdot a_2a_1a_0 = 0$. The first equality tells us that $a_1 = 0$. The last equality tells us that $a_2 \oplus 0 \oplus a_0 = 0$, which means $a_2 = a_1 = 1$ (since they can't both be 0). So $a = 101$, as expected.

□

2. A Simon's Algorithm Primer'. You are given a unitary operator $\mathbf{U}_f$ that encodes the function $f : \{0,1\}^n \rightarrow \{0,1\}^n$ which has the characteristics $f(x) = f(x \oplus a)$ but this time for $a = 0$. The fact that $a = 0$ means that every output value of $f(x)$ is distinct – the function is one-to-one.

Show that employing the circuit diagram from the prior problem will lead to set of all 2^n possible states of the input register, each with an equal probability $P_i = 1/2^n$ of being measured in the final step of the algorithm.

Notice that in this case (if you didnt know that $a = 0$ and weren't assured that $a \neq 0$) if you just ran the algorithm you could also generate a set of equations satisfying $a \cdot z_i = 0$ for $0 \leq i \leq n_1$. If you did the same classical post-processing (solving the equations for a you would get some nonzero value. Of course all states satisfy $a \cdot z_i = 0$ if $a = 0$. You'd then have to check two inputs x and $x \oplus a$ to the oracle - maybe $|000\rangle$ and $|a\rangle$ - and see if they have the same output value. If not then you know $a = 0$.

Solution: When $a = 0 \implies f$ is 1-to-1 (and onto), we have, by looking at $|\psi_2\rangle$ in the previous problem, that each $|x\rangle$ goes with a distinct $|f(x)\rangle$. This means $|\psi_3\rangle$ is a tensor product of two n-Qbit computational basis states: the first is that of the input register, the second is that of the output register. Because the input register is now in a basis state, Hadamard-ing it gives an equal superposition of all 2^n computational basis states without destructive interference (unlike in the previous problem where this happens). This means each state of the input register has an amplitude $1/\sqrt{2^n}$, which corresponds to a probability $P_i = 1/2^n$ of being measured. □

2.9 Problem set 9

1. Compute 118 (mod 5). Do this using the definition, not Excel.

Solution: We write $118 = 115 + 3$ to get $118 \mod 5 = 3 \mod 5 = 3$. □

2. What is the greatest common denominator of 102 and 12? Do this with Euclid's algorithm on paper with a pen or pencil, but feel free to check your answer using the Excel spreadsheet from Moodle.

Solution: Euclid's algorithm:

$$\begin{array}{lll} n = 0 & a_0 = 102 & b_0 = 12 \\ n = 1 & a_1 = 12 & b_1 = 102 \mod 12 = 6 \\ n = 2 & a_2 = 6 & b_2 = 12 \mod 6 = 0 \end{array} \tag{2.173}$$

And so $\gcd(102, 12) = 6$. □

3. What is $27 \cdot 26 \pmod{21}$?

Solution:

$$\begin{aligned} 27 \cdot 26 \mod 21 &= (27 \mod 21) \cdot (26 \mod 21) \mod 21 \\ &= (6 \cdot 5) \mod 21 = 9. \end{aligned} \tag{2.174}$$

4. List the elements in G_5. What is the order of the group?

Solution: $G_5 = \{x \in \mathbb{N} : \gcd(x, 5) = 1, x < 5\} = \{1, 2, 3, 4\}$. The order of G_5 is $|G_5| = 4$. □

5. What is the order of the element 2 in the group G_5? Figure this out on paper with a pen or pencil, but feel free to check using the Excel spreadsheet from Moodle.

Solution: We can do this by exhaustion: $2^1 = 2 \equiv_5 2$, $2^2 = 4 \equiv_5 4$, $2^3 = 8 \equiv_5 3$, $2^4 = 16 \equiv 51$. We can also do the following: the order of any element must divide the order of the group. The order of G_5 is 4, so the order of 2 is $1, 2$, or 4. Obviously the order of 2 is not 1. Also, $2^2 = 4 \equiv_5 4 \neq 1$, so $|2| = 4$. □

6. What is $2^8 \pmod 5$? (Again, paper and pencil or pen!)

Solution: We can write 2^8 as a product of numbers "close" to 5 like this: $2^8 = (2^4)^2 = 16^2 \equiv_5 (16 \mod 5)(16 \mod 5) \mod 5 = 1$. □

7. What is $2^{513} \pmod 5$? Think before you plunge into a long calculation.

Solution: The exponent is big, so we don't want to do what we did in the previous problem. Rather, we use the fact that $|G_5| = 4$, which means 2^{4k}, for any natural number k, is $1 \mod 5 = 1$. Okay, because $513 \mod 4 = (512 + 1) \mod 4 = 1$, we have that $2^{513} \equiv_5 2^1 = \boxed{2}$. Here we have used the fact that $ab \mod N = (a \mod N)(b \mod N) \mod N$.

SageMathCell is very good at algebra:

```
sage: Mod(2**513,5)
>>>   2
```

□

8. List the elements in G_{15}. What is the order of the group?

Solution: By definition

$$G_{15} = \{x \in \mathbb{N} : \gcd(x, 15) = 1, x < 15\} = \{1, 2, 4, 7, 8, 11, 13, 14\}. \tag{2.175}$$

The order of the group is 8. We can also "calculate" $|G_{15}|$ is two ways. We know that $G_{15} = G_{3\cdot 5} = (3-1)(5-1) = 2 \cdot 4 = 8$, using Fermat's Little Theorem. We can also use the Euler's totient function ϕ in Sage:

```
sage: euler_phi(15)
>>>   8
```

□

9. Within the group G_{15}, what is the order of the element 7?

Solution: The order of any element divides the order of the group. Here $|G_{15}| = 8$, so the order of the element 7, $|7|$, is 1,2,4, or 8. We notice that $7^4 = 49 \cdot 49 \equiv_5 (49 \mod 5) \cdot (49 \mod 5) \mod 5 = 16 \mod 5 = 1$. So, $|7| = 4$.

□

10. Using the result of the prior question, find the prime factors of 15.

Solution: The setup is $N = 15, k = 4, a = 7$. With these, we have

$$\left(7^{4/2} - 1\right) \mod 15 = 48 \mod 15 = 3 \tag{2.176}$$

$$\left(7^{4/2} + 1\right) \mod 15 = 50 \mod 15 = 5. \tag{2.177}$$

We can already see where this is going...

$$\gcd(3, 15) = 3, \quad \gcd(5, 15) = 5. \tag{2.178}$$

So the prime factors of 15 are 3 and 5. We know that these are all the prime factors because they're prime and $3 \cdot 5 = 15$. □

11. (Optional) What is the order of G_{55}? What is the order of the element 7 within G_{55}? Using that order, determine the prime factors of 55.

Solution: Suppose we can cheat and know 5 and 11 are the prime factors of 55. Fermat's little theorem says $|G_{55}| = |G_{5\cdot 11}| = (5-1)(11-1) = 40$. We can also check this in Sage:

```
sage: euler_phi(55)
>>>   40
```

Suppose we don't cheat then it is still possible to go through and find the elements of G_{55} one-by-one. This can be done in Excel, Python, etc.

The order of the element 7 divides the order of the group, which is 40. The factors of 40 are 1, 2, 4, 5, 8, 10, 20, 40. We easily eliminate 1 and 2. $7^{40} \equiv_{55} 1$, so let's check 20: $7^{20} \equiv_{55} 36^5 \equiv_{55} = 31 \cdot 31 \cdot 36 \mod 5 = 26 \cdot 36 \mod 55 = 1$. Note that the order of 7 cannot be lower than 20 because in the process about we never see $(1 \mod 55)^x \mod 55$ where x is a factor of 20. So, $|7| = 20$.

So we have $N = 55, k = 20, a = 7$. So,

$$\begin{aligned} \left(2^{20/2} - 1\right) \mod 55 &= 1023 \mod 55 = 33 \\ \left(2^{20/2} + 1\right) \mod 55 &= 1025 \mod 55 = 35 \end{aligned} \tag{2.179}$$

The rest is easy:

$$\begin{aligned} \gcd(55, 33) &= 11 \\ \gcd(55, 35) &= 5. \end{aligned} \tag{2.180}$$

So the algorithm says 5 and 11 are the prime factors of 55, as expected. □

12. The goal of this problem is to work through an example of RSA encryption and decryption to see how they work. In this example, Bob wants to ask Alice what her favorite mascot is, and she want an encoded message from Alice (which is "mule"). Bob chooses to use the integer $N = 55$ and the coding number $c = 17$ as the public key.

(a) Determine a numerical representation of the message $\{a_1, a_2, a_3, a_4\}$ using a scheme that Alice and Bob can agree on publicly.

(b) Using the public key determine the encoded version of the message $\{e_1, e_2, e_3, e_4\}$. Feel free to use the version of the repeated-squaring calculation of powers, but try it at least once using just paper and a pencil to get a feel for the algorithm.

(c) Since you know an algorithm for factoring 55 into its two prime factors (or you did it above) you can figure out what Bob uses as a decoding number d. What is it? Show explicitly that

$$cd = 1 \mod (p-1)(q-1) \tag{2.181}$$

(d) Use the decoding algorithm to determine the decoded message $\{m_1, m_2, m_3, m_4\}$ and show that it is equal to Alice's original message.

Solution:

(a) In the ordered English alphabet, "mule" is represented by the string {`13, 21, 12, 5`}.

(b) Now we use the public key to determine the encoded version of the message $\{e_1, e_2, e_3, e_4\}$, where $e_i = a_i^c \mod N$.

$$\begin{aligned}
13^{17} &\mod 55 = 4^8 \cdot 13 \mod 55 = 36^2 \cdot 13 \mod 55 = 18 \\
21^{17} &\mod 55 = (21^2)^8 \cdot 21 \mod 55 = 1 \cdot 21 \mod 55 = 21 \\
12^{17} &\mod 55 = (34^2)^4 \cdot 12 \mod 55 = 1 \cdot 12 \mod 55 = 12 \\
5^{17} &\mod 55 = 20^4 \cdot 5 \mod 55 = 15^2 \cdot 5 \mod 55 = 5^2 \mod 55 = 25
\end{aligned}$$

So the encoded version of the message is {`18, 21, 12, 25`}.

(c) The public key is {`55, 17`}. The number d is such that

$$17d \mod (5-1)(11-1) = 17d \mod 40 = 1. \tag{2.182}$$

The smallest d is $\boxed{33}$. After some trials and errors I realized finding d by hand wasn't very easy – especially because 17 is prime. So I resorted to Excel. In any case,

$$17 \cdot 33 \mod 40 = (560 + 1) \mod 40 = 1. \tag{2.183}$$

(d) The decoded message is $\{m_1, m_2, m_3, m_4\}$ where $m_i = e_i^{33} \mod 55$

$$\begin{aligned}
m_1 &\equiv_{55} 18^{33} \equiv_{55} 49^{16} \cdot 18 \equiv_{55} 36^8 \cdot 18 \equiv_{55} 31^4 \cdot 18 \equiv_{55} 26^2 \cdot 18 \equiv_{55} 13 \\
m_2 &\equiv_{55} 21^{33} \equiv_{55} 1^{16} \cdot 21 \equiv_{55} 21 \\
m_3 &\equiv_{55} 12^{33} \equiv_{55} 34^16 \cdot 12 \equiv_{55} 1^8 \cdot 12 \equiv_{55} 12 \\
m_4 &\equiv_{55} 25^{33} \equiv_{55} 20^{16} \cdot 25 \equiv_{55} 15^8 \cdot 25 \equiv_{55} 5^4 \cdot 25 \equiv_{55} 20 \cdot 25 \equiv_{55} 5.
\end{aligned} \tag{2.184}$$

So, the the decoded message is {`13, 21, 12, 25`}, which is Alice's original message. □

2.10 Problem set 10

1. What is the matrix representation of the operator $\mathbf{U}_{FT}$ for a 1-Qbit state (n=1)? Does this look like an operator you have seen before?

Solution: The 1-Qbit FT matrix is given by

$$\mathbf{U}_{FT} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & e^{2i\pi/2} \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \tag{2.185}$$

which is the just the regular 1-Qbit Hadamard. □

2. What is the matrix representation of the operator $\mathbf{U}_{FT}$ for a 2-Qbit state (n=2)?

Solution: The 2-Qbit FT matrix is given by

$$\mathbf{U}_{FT} = \frac{1}{\sqrt{4}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & e^{2i\pi/4} & e^{4i\pi/4} & e^{6i\pi/4} \\ 1 & e^{4i\pi/4} & e^{8i\pi/4} & e^{12i\pi/4} \\ 1 & e^{6i\pi/4} & e^{12i\pi/4} & e^{18i\pi/4} \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \tag{2.186}$$

□

3. We wish to apply the 2-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{2})\,(|0\rangle + |2\rangle)$:

$$\mathbf{U}_{FT}\,|\psi\rangle = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}} \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} = |\psi\rangle\,. \tag{2.187}$$

□

4. We wish to apply the 2-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{2})\,(|0\rangle + |3\rangle)$:

$$\mathbf{U}_{FT}\,|\psi\rangle = \frac{1}{2\sqrt{2}} \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix} = \frac{1}{2\sqrt{2}} \begin{bmatrix} 2 \\ 1-i \\ 0 \\ 1+i \end{bmatrix} \tag{2.188}$$

□

5.

(a) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{2})\,(|2\rangle + |6\rangle)$. What do you observe about the transformed state?

$$\begin{aligned}
\mathbf{U}_{FT}\,|\psi\rangle &= \frac{1}{\sqrt{2^4}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix} \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 1 & 0 & -1 & 0 & 1 & 0 & -1 & 0 \end{bmatrix}^{\top} \\
&= \frac{1}{2} \left[|0\rangle - |2\rangle + |4\rangle - |6\rangle \right]. \qquad (2.189)
\end{aligned}$$

Observations:

- The "distribution" of states in the beginning is small (only two), the transformed state spans over more states (all with the same weights). In the language of FT, this makes sense that we have a large number of frequency components in the output because we start with a function that is "uncertain" in frequencies.
- The transformed state is periodic with frequency 2 in the sense that when we plot the amplitudes as a function of the basis states, we get an "oscillation" between 1 and -1, which repeats every 2 states.
- We can think of the FT of the state $(1/\sqrt{2})(|2\rangle+|6\rangle)$ as a combination of equal weights of the FT of $|2\rangle$ and $|6\rangle$. The action of the FT on $|2\rangle$ as wrapping around the origin in the + direction at frequency 2 (in the sense that the amplitudes complete 2 full revolutions around the origin), while the action of the FT on $|6\rangle$ as wrapping around the origin in the + direction at frequency 6. These actions combined give an overall frequency of 2.
- Graphically, if we write out the basis states on a line and for each basis state we draw a complex plane then the FT of $|2\rangle$ looks like a + helix of frequency 2 and the FT of $|6\rangle$ looks like a + helix of frequency 6. These helices cancel in amplitudes for odd states $|1\rangle, \ldots$ and add for even states $|0\rangle, \ldots$. The FT of $|2\rangle + |6\rangle$ thus looks sinusoidal (with frequency 2) in the Real plane.

(b) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{2})\,(|1\rangle + |5\rangle)$. What do you observe about the transformed state?

$$\begin{aligned}
\mathbf{U}_{FT}\,|\psi\rangle &= \frac{1}{\sqrt{2^4}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} \\
&= \frac{1}{2} \begin{bmatrix} 1 & 0 & i & 0 & -1 & 0 & -i & 0 \end{bmatrix}^\top \\
&= \frac{1}{2} \left[|0\rangle + i\,|2\rangle - |4\rangle - i\,|6\rangle \right]. \qquad (2.190)
\end{aligned}$$

Observations:

- The transformed state here is only different from that in (a) by shifts in the relative phases.
- In the same language we used to graphically describe the states in part (a), we say that the transformed state here is a + helix with frequency 1. This is because the FT of $|1\rangle$ is a + helix with frequency 1 and the FT of $|5\rangle$ is a + helix with frequency 5. Together, the amplitudes add for even states $|0\rangle, \ldots$ and cancel for odd states $|1\rangle, \ldots$. The resulting state is a + helix of frequency 1.

(c) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/2)(|1\rangle + |3\rangle + |5\rangle + |7\rangle)$. What do you observe about the transformed state?

$$\begin{aligned}
\mathbf{U}_{FT}|\psi\rangle &= \frac{1}{\sqrt{2^4}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{bmatrix} \\
&= \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \end{bmatrix}^\top \\
&= \frac{1}{\sqrt{2}} \left[|0\rangle - |4\rangle \right]. \qquad (2.191)
\end{aligned}$$

Observation(s):

- The original state spans multiple (4) basis states, and so it makes sense the FT spans only over 2 states. We also notice that the input is quite periodic, so it also makes sense that the output has few "frequency" components.
- This FT is a combination of the FT of $|1\rangle + |5\rangle$ and $|3\rangle + |7\rangle$, which only differ in the relative phases. Their combination is such that the amplitudes of $|2\rangle$ and $|6\rangle$ cancel out, leaving only nonzero amplitudes for $|0\rangle$ and $|4\rangle$.

(d) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/2)(|0\rangle + |2\rangle + |4\rangle + |6\rangle)$. What do you observe about the transformed state?

$$\begin{aligned}
\mathbf{U}_{FT}|\psi\rangle &= \frac{1}{\sqrt{2^4}}\begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 0 \end{bmatrix} \\
&= \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \end{bmatrix}^\top \\
&= \frac{1}{\sqrt{2}}\left[|0\rangle + |4\rangle\right]. \qquad (2.192)
\end{aligned}$$

Observation(s):

- Again, this differs from the FT in the previous item only in the relative phases.

(e) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{3})\,(|1\rangle + |4\rangle + |7\rangle)$. What do you observe about the transformed state? What makes this transformation different than the prior ones?

$$\mathbf{U}_{FT}\,|\psi\rangle = \frac{1}{\sqrt{2^4}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix} \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}$$

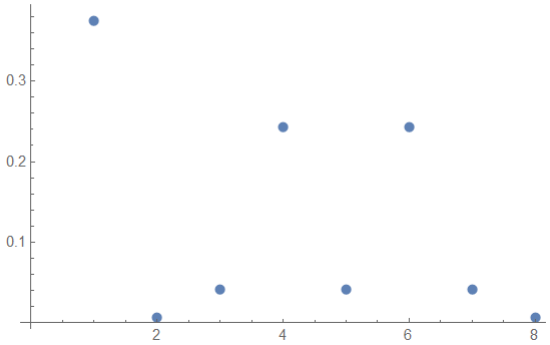
$$= \frac{1}{\sqrt{24}} \begin{bmatrix} 3 \\ -1+\sqrt{2} \\ 1 \\ -1-\sqrt{2} \\ -1 \\ -1-\sqrt{2} \\ 1 \\ -1+\sqrt{2} \end{bmatrix}. \tag{2.193}$$

Observation(s): We will make more observations after looking at part (f), but the main difference between the states here and those in the previous parts is that $|1\rangle + |4\rangle + |7\rangle$ doesn't "complete" a cycle. While we suspect that the function has period 3, $|7+3\rangle$ does not quite loop back to $|1\rangle$, unlike before where $|6+4\rangle \equiv |2\rangle$, for instance. This means the FT will have some "uncertainty" regarding the frequency components of this "signal." We will see this part (f).

(f) For the state in part (e) plot the probability of finding the system in each of the computational basis states $|x_i\rangle$ after it has been Fourier transformed.

```
In[90]:= ListPlot[U.{0, 1, 0, 0, 1, 0, 0, 1} *
           U.{0, 1, 0, 0, 1, 0, 0, 1} / 24]
```

Out[90]=



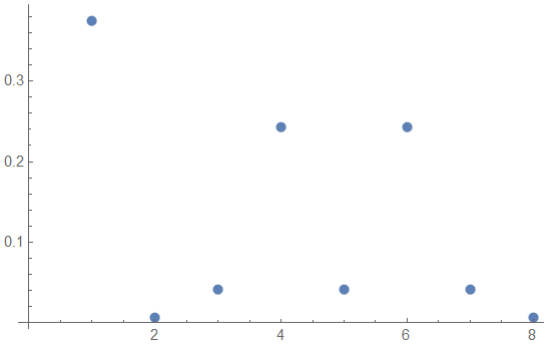
(g) Apply the 3-Qbit QFT to the state $|\psi\rangle = (1/\sqrt{3})\,(|0\rangle + |3\rangle + |6\rangle)$. What do you observe about the transformed state? What makes this transformation different than the prior ones?

$$\mathbf{U}_{FT}\,|\psi\rangle = \frac{1}{\sqrt{2^4}} \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & \sqrt{i} & i & i\sqrt{i} & -1 & -\sqrt{i} & -i & -i\sqrt{i} \\ 1 & i & -1 & -i & 1 & i & -1 & -i \\ 1 & i\sqrt{i} & -i & \sqrt{i} & -1 & -i\sqrt{i} & i & -\sqrt{i} \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & -\sqrt{i} & i & -i\sqrt{i} & -1 & \sqrt{i} & -i & i\sqrt{i} \\ 1 & -i & -1 & i & 1 & -i & -1 & i \\ 1 & -i\sqrt{i} & -i & -\sqrt{i} & -1 & i\sqrt{i} & i & \sqrt{i} \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}$$

$$= \frac{1}{\sqrt{24}} \begin{bmatrix} 3 \\ 1 - i + \sqrt{i} \\ -i \\ 1 + i + \sqrt{i} \\ 1 \\ 1 - i - \sqrt{i} \\ i \\ 1 + i - \sqrt{i} \end{bmatrix}. \tag{2.194}$$

Observation(s): The input here is linearly shifted from the input from part (f), so it makes sense that the output is phase-shifted from the output in (f) – and it is. The number's aren't so clear so we look at the spectrum of the FT: As expected, it looks exactly the same as that in (f).

```
ListPlot[U.{1, 0, 0, 1, 0, 0, 1, 0} *
  Conjugate[U.{1, 0, 0, 1, 0, 0, 1, 0}] / 24]
```



□