

Chapter 10

Quantum algorithms

This book started with the world’s oldest and most widely used algorithms (the ones for adding and multiplying numbers) and an ancient hard problem (FACTORING). In this last chapter the tables are turned: we present one of the latest algorithms—and it is an efficient algorithm for FACTORING!

There is a catch, of course: this algorithm needs a *quantum computer* to execute.

Quantum physics is a beautiful and mysterious theory that describes Nature in the small, at the level of elementary particles. One of the major discoveries of the nineties was that quantum computers—computers based on quantum physics principles—are radically different from those that operate according to the more familiar principles of classical physics. Surprisingly, they can be exponentially more powerful: as we shall see, quantum computers can solve FACTORING in polynomial time! As a result, in a world with quantum computers, the systems that currently safeguard business transactions on the Internet (and are based on the RSA cryptosystem) will no longer be secure.

10.1 Qubits, superposition, and measurement

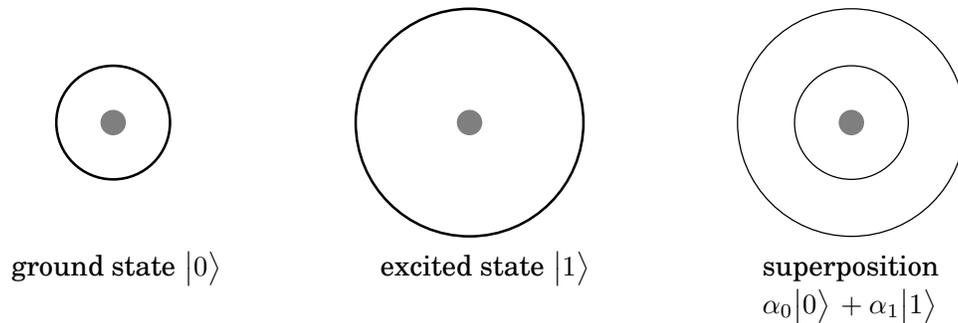
In this section we introduce the basic features of quantum physics that are necessary for understanding how quantum computers work.¹

In ordinary computer chips, bits are physically represented by low and high voltages on wires. But there are many other ways a bit could be stored—for instance, in the state of a hydrogen atom. The single electron in this atom can either be in the ground state (the lowest energy configuration) or it can be in an excited state (a high energy configuration). We can use these two states to encode for bit values 0 and 1, respectively.

Let us now introduce some quantum physics notation. We denote the ground state of our electron by $|0\rangle$, since it encodes for bit value 0, and likewise the excited state by $|1\rangle$. These are

¹This field is so strange that the famous physicist Richard Feynman is quoted as having said, “I think I can safely say that no one understands quantum physics.” So there is little chance you will understand the theory in depth after reading this section! But if you are interested in learning more, see the recommended reading at the book’s end.

Figure 10.1 An electron can be in a ground state or in an excited state. In the Dirac notation used in quantum physics, these are denoted $|0\rangle$ and $|1\rangle$. But the superposition principle says that, in fact, the electron is in a state that is a *linear combination* of these two: $\alpha_0|0\rangle + \alpha_1|1\rangle$. This would make immediate sense if the α 's were probabilities, nonnegative real numbers adding to 1. But the superposition principle insists that they can be *arbitrary complex numbers*, as long as the squares of their norms add up to 1!



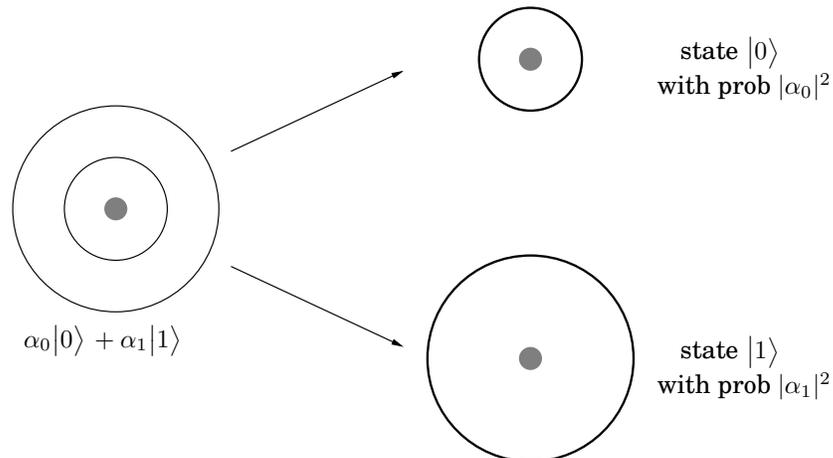
the two possible states of the electron in classical physics. Many of the most counterintuitive aspects of quantum physics arise from the *superposition principle* which states that if a quantum system can be in one of two states, then it can also be in *any linear superposition* of those two states. For instance, the state of the electron could well be $\frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$ or $\frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$; or an infinite number of other combinations of the form $\alpha_0|0\rangle + \alpha_1|1\rangle$. The coefficient α_0 is called the *amplitude* of state $|0\rangle$, and similarly with α_1 . And—if things aren't already strange enough—the α 's can be complex numbers, as long as they are normalized so that $|\alpha_0|^2 + |\alpha_1|^2 = 1$. For example, $\frac{1}{\sqrt{5}}|0\rangle + \frac{2i}{\sqrt{5}}|1\rangle$ (where i is the imaginary unit, $\sqrt{-1}$) is a perfectly valid quantum state! Such a superposition, $\alpha_0|0\rangle + \alpha_1|1\rangle$, is the basic unit of encoded information in quantum computers (Figure 10.1). It is called a *qubit* (pronounced “cubit”).

The whole concept of a superposition suggests that the electron does not make up its mind about whether it is in the ground or excited state, and the amplitude α_0 is a measure of its inclination toward the ground state. Continuing along this line of thought, it is tempting to think of α_0 as the *probability* that the electron is in the ground state. But then how are we to make sense of the fact that α_0 can be negative, or even worse, imaginary? This is one of the most mysterious aspects of quantum physics, one that seems to extend beyond our intuitions about the physical world.

This linear superposition, however, is the private world of the electron. For us to get a glimpse of the electron's state we must make a *measurement*, and when we do so, we get a single bit of information—0 or 1. If the state of the electron is $\alpha_0|0\rangle + \alpha_1|1\rangle$, then the outcome of the measurement is 0 with probability $|\alpha_0|^2$ and 1 with probability $|\alpha_1|^2$ (luckily we normalized so $|\alpha_0|^2 + |\alpha_1|^2 = 1$). Moreover, the act of measurement causes the system to change its state: if the outcome of the measurement is 0, then the new state of the system is $|0\rangle$ (the ground state), and if the outcome is 1, the new state is $|1\rangle$ (the excited state). This

feature of quantum physics, that a measurement disturbs the system and forces it to choose (in this case ground or excited state), is another strange phenomenon with no classical analog.

Figure 10.2 Measurement of a superposition has the effect of forcing the system to decide on a particular state, with probabilities determined by the amplitudes.



The superposition principle holds not just for 2-level systems like the one we just described, but in general for k -level systems. For example, in reality the electron in the hydrogen atom can be in one of many energy levels, starting with the ground state, the first excited state, the second excited state, and so on. So we could consider a k -level system consisting of the ground state and the first $k - 1$ excited states, and we could denote these by $|0\rangle, |1\rangle, |2\rangle, \dots, |k - 1\rangle$. The superposition principle would then say that the general quantum state of the system is $\alpha_0|0\rangle + \alpha_1|1\rangle + \dots + \alpha_{k-1}|k - 1\rangle$, where $\sum_{j=0}^{k-1} |\alpha_j|^2 = 1$. Measuring the state of the system would now reveal a number between 0 and $k - 1$, and outcome j would occur with probability $|\alpha_j|^2$. As before, the measurement would disturb the system, and the new state would *actually become* $|j\rangle$ or the j th excited state.

How do we encode n bits of information? We could choose $k = 2^n$ levels of the hydrogen atom. But a more promising option is to use n qubits.

Let us start by considering the case of two qubits, that is, the state of the electrons of *two* hydrogen atoms. Since each electron can be in either the ground or excited state, in classical physics the two electrons have a total of four possible states—00, 01, 10, or 11—and are therefore suitable for storing 2 bits of information. But in quantum physics, the superposition principle tells us that the quantum state of the two electrons is a linear combination of the four classical states,

$$|\alpha\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle,$$

normalized so that $\sum_{x \in \{0,1\}^2} |\alpha_x|^2 = 1$.² Measuring the state of the system now reveals 2 bits

²Recall that $\{0, 1\}^2$ denotes the set consisting of the four 2-bit binary strings and in general $\{0, 1\}^n$ denotes the set of all n -bit binary strings.

Entanglement

Suppose we have two qubits, the first in the state $\alpha_0|0\rangle + \alpha_1|1\rangle$ and the second in the state $\beta_0|0\rangle + \beta_1|1\rangle$. What is the joint state of the two qubits? The answer is, the (tensor) product of the two: $\alpha_0\beta_0|00\rangle + \alpha_0\beta_1|01\rangle + \alpha_1\beta_0|10\rangle + \alpha_1\beta_1|11\rangle$.

Given an arbitrary state of two qubits, can we specify the state of each individual qubit in this way? No, in general the two qubits are *entangled* and cannot be decomposed into the states of the individual qubits. For example, consider the state $|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$, which is one of the famous Bell states. It cannot be decomposed into states of the two individual qubits (see Exercise 10.1). Entanglement is one of the most mysterious aspects of quantum mechanics and is ultimately the source of the power of quantum computation.

of information, and the probability of outcome $x \in \{0, 1\}^2$ is $|\alpha_x|^2$. Moreover, as before, if the outcome of measurement is jk , then the new state of the system is $|jk\rangle$: if $jk = 10$, for example, then the first electron is in the excited state and the second electron is in the ground state.

An interesting question comes up here: what if we make a *partial measurement*? For instance, if we measure just the first qubit, what is the probability that the outcome is 0? This is simple. It is exactly the same as it would have been had we measured both qubits, namely, $\Pr\{\text{1st bit} = 0\} = \Pr\{00\} + \Pr\{01\} = |\alpha_{00}|^2 + |\alpha_{01}|^2$. Fine, but how much does this partial measurement disturb the state of the system?

The answer is elegant. If the outcome of measuring the first qubit is 0, then the new superposition is obtained by crossing out all terms of $|\alpha\rangle$ that are inconsistent with this outcome (that is, whose first bit is 1). Of course the sum of the squares of the amplitudes is no longer 1, so we must renormalize. In our example, this new state would be

$$|\alpha_{\text{new}}\rangle = \frac{\alpha_{00}}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}|00\rangle + \frac{\alpha_{01}}{\sqrt{|\alpha_{00}|^2 + |\alpha_{01}|^2}}|01\rangle.$$

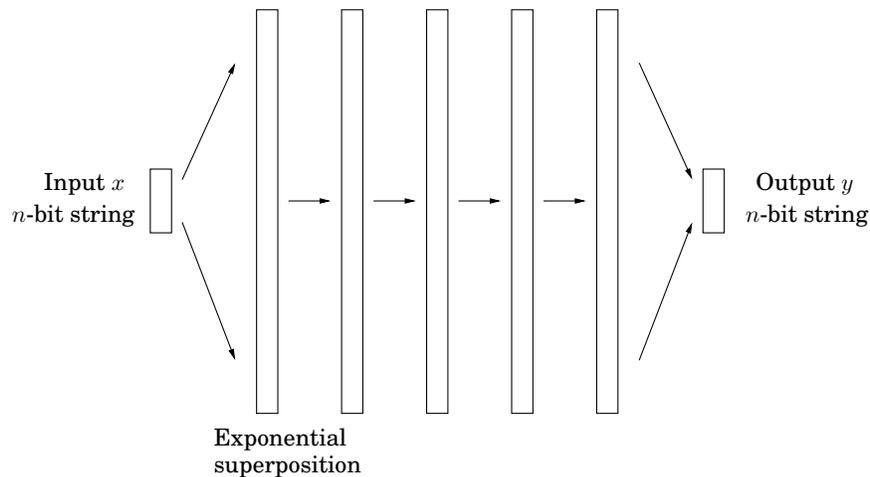
Finally, let us consider the general case of n hydrogen atoms. Think of n as a fairly small number of atoms, say $n = 500$. Classically the states of the 500 electrons could be used to store 500 bits of information in the obvious way. But the quantum state of the 500 qubits is a linear superposition of all 2^{500} possible classical states:

$$\sum_{x \in \{0,1\}^n} \alpha_x |x\rangle.$$

It is as if Nature has 2^{500} scraps of paper on the side, each with a complex number written on it, just to keep track of the state of this system of 500 hydrogen atoms! Moreover, at each moment, as the state of the system evolves in time, it is as though Nature crosses out the complex number on each scrap of paper and replaces it with its new value.

Let us consider the effort involved in doing all this. The number 2^{500} is much larger than estimates of the number of elementary particles in the universe. Where, then, does Nature store this information? How could microscopic quantum systems of a few hundred atoms

Figure 10.3 A quantum algorithm takes n “classical” bits as its input, manipulates them so as to create a superposition of their 2^n possible states, manipulates this exponentially large superposition to obtain the final quantum result, and then measures the result to get (with the appropriate probability distribution) the n output bits. For the middle phase, there are elementary operations which count as one step and yet manipulate all the exponentially many amplitudes of the superposition.



contain more information than we can possibly store in the entire classical universe? Surely this is a most extravagant theory about the amount of effort put in by Nature just to keep a tiny system evolving in time.

In this phenomenon lies the basic motivation for quantum computation. After all, if Nature is so extravagant at the quantum level, why should we base our computers on classical physics? Why not tap into this massive amount of effort being expended at the quantum level?

But there is a fundamental problem: this exponentially large linear superposition is the private world of the electrons. Measuring the system only reveals n bits of information. As before, the probability that the outcome is a particular 500-bit string x is $|\alpha_x|^2$. And the new state after measurement is just $|x\rangle$.

10.2 The plan

A quantum algorithm is unlike any you have seen so far. Its structure reflects the tension between the exponential “private workspace” of an n -qubit system and the mere n bits that can be obtained through measurement.

The input to a quantum algorithm consists of n classical bits, and the output also consists of n classical bits. It is while the quantum system is not being watched that the quantum effects take over and we have the benefit of Nature working exponentially hard on our behalf.

If the input is an n -bit string x , then the quantum computer takes as input n qubits in

state $|x\rangle$. Then a series of quantum operations are performed, by the end of which the state of the n qubits has been transformed to some superposition $\sum_y \alpha_y |y\rangle$. Finally, a measurement is made, and the output is the n -bit string y with probability $|\alpha_y|^2$. Observe that this output is *random*. But this is not a problem, as we have seen before with randomized algorithms such as the one for primality testing. As long as y corresponds to the right answer with high enough probability, we can repeat the whole process a few times to make the chance of failure miniscule.

Now let us look more closely at the quantum part of the algorithm. Some of the key quantum operations (which we will soon discuss) can be thought of as looking for certain kinds of *patterns* in a superposition of states. Because of this, it is helpful to think of the algorithm as having two stages. In the first stage, the n classical bits of the input are “unpacked” into an exponentially large superposition, which is expressly set up so as to have an underlying pattern or regularity that, if detected, would solve the task at hand. The second stage then consists of a suitable set of quantum operations, followed by a measurement, which reveals the hidden pattern.

All this probably sounds quite mysterious at the moment, but more details are on the way. In Section 10.3 we will give a high-level description of the most important operation that can be efficiently performed by a quantum computer: a quantum version of the fast Fourier transform (FFT). We will then describe certain patterns that this quantum FFT is ideally suited to detect, and will show how to recast the problem of factoring an integer N in terms of detecting precisely such a pattern. Finally we will see how to set up the initial stage of the quantum algorithm, which converts the input N into an exponentially large superposition with the right kind of pattern.

The algorithm to factor a large integer N can be viewed as a sequence of reductions (and everything shown here in italics will be defined in good time):

- FACTORING is reduced to finding a *nontrivial square root of 1 modulo N* .
- Finding such a root is reduced to computing the *order* of a random integer modulo N .
- The order of an integer is precisely the *period* of a particular *periodic superposition*.
- Finally, periods of superpositions can be found by the *quantum FFT*.

We begin with the last step.

10.3 The quantum Fourier transform

Recall the fast Fourier transform (FFT) from Chapter 2. It takes as input an M -dimensional, complex-valued vector α (where M is a power of 2, say $M = 2^m$), and outputs an M -dimensional

complex-valued vector β :

$$\begin{bmatrix} \beta_0 \\ \beta_1 \\ \beta_2 \\ \vdots \\ \beta_{M-1} \end{bmatrix} = \frac{1}{\sqrt{M}} \begin{bmatrix} 1 & 1 & 1 & \dots & 1 \\ 1 & \omega & \omega^2 & \dots & \omega^{M-1} \\ 1 & \omega^2 & \omega^4 & \dots & \omega^{2(M-1)} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \omega^j & \omega^{2j} & \dots & \omega^{(M-1)j} \\ \vdots & \vdots & \vdots & \dots & \vdots \\ 1 & \omega^{(M-1)} & \omega^{2(M-1)} & \dots & \omega^{(M-1)(M-1)} \end{bmatrix} \begin{bmatrix} \alpha_0 \\ \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_{M-1} \end{bmatrix},$$

where ω is a complex M th root of unity (the extra factor of \sqrt{M} is new and has the effect of ensuring that if the $|\alpha_i|^2$ add up to 1, then so do the $|\beta_i|^2$). Although the preceding equation suggests an $O(M^2)$ algorithm, the classical FFT is able to perform this calculation in just $O(M \log M)$ steps, and it is this speedup that has had the profound effect of making digital signal processing practically feasible. We will now see that quantum computers can implement the FFT *exponentially* faster, in $O(\log^2 M)$ time!

But wait, how can any algorithm take time less than M , the length of the input? The point is that we can encode the input in a superposition of just $m = \log M$ qubits: after all, this superposition consists of 2^m amplitude values. In the notation we introduced earlier, we would write the superposition as $|\alpha\rangle = \sum_{j=0}^{M-1} \alpha_j |j\rangle$ where α_i is the amplitude of the m -bit binary string corresponding to the number i in the natural way. This brings up an important point: the $|j\rangle$ notation is really just another way of writing a vector, where the index of each entry of the vector is written out explicitly in the special bracket symbol.

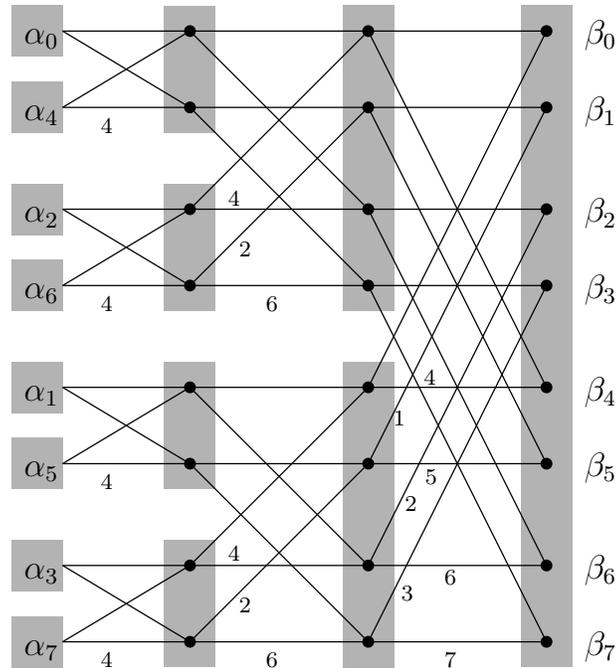
Starting from this input superposition $|\alpha\rangle$, the *quantum Fourier transform (QFT)* manipulates it appropriately in $m = \log M$ stages. At each stage the superposition evolves so that it encodes the intermediate results at the same stage of the classical FFT (whose circuit, with $m = \log M$ stages, is reproduced from Chapter 2 in Figure 10.4). As we will see in Section 10.5, this can be achieved with m quantum operations per stage. Ultimately, after m such stages and $m^2 = \log^2 M$ elementary operations, we obtain the superposition $|\beta\rangle$ that corresponds to the desired output of the QFT.

So far we have only considered the good news about the QFT: its amazing speed. Now it is time to read the fine print. The classical FFT algorithm actually *outputs* the M complex numbers $\beta_0, \dots, \beta_{M-1}$. In contrast, the QFT only prepares a superposition $|\beta\rangle = \sum_{j=0}^{M-1} \beta_j |j\rangle$. And, as we saw earlier, these amplitudes are part of the “private world” of this quantum system.

Thus the only way to get our hands on this result is by measuring it! And measuring the state of the system only yields $m = \log M$ classical bits: specifically, the output is index j with probability $|\beta_j|^2$.

So, instead of QFT, it would be more accurate to call this algorithm *quantum Fourier sampling*. Moreover, even though we have confined our attention to the case $M = 2^m$ in this section, the algorithm can be implemented for arbitrary values of M , and can be summarized as follows:

Figure 10.4 The classical FFT circuit from Chapter 2. Input vectors of M bits are processed in a sequence of $m = \log M$ levels.



Input: A superposition of $m = \log M$ qubits, $|\alpha\rangle = \sum_{j=0}^{M-1} \alpha_j |j\rangle$.

Method: Using $O(m^2) = O(\log^2 M)$ quantum operations perform the quantum FFT to obtain the superposition $|\beta\rangle = \sum_{j=0}^{M-1} \beta_j |j\rangle$.

Output: A random m -bit number j (that is, $0 \leq j \leq M - 1$), from the probability distribution $Pr[j] = |\beta_j|^2$.

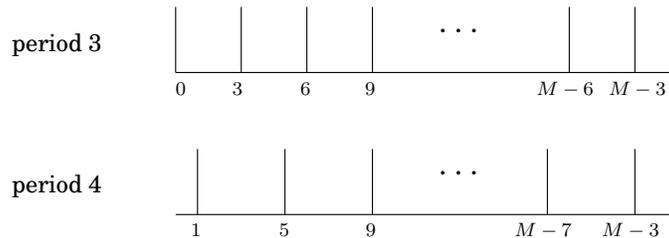
Quantum Fourier sampling is basically a quick way of getting a very rough idea about the output of the classical FFT, just detecting one of the larger components of the answer vector. In fact, we don't even see the value of that component—we only see its index. How can we use such meager information? In which applications of the FFT is just the index of the large components enough? This is what we explore next.

10.4 Periodicity

Suppose that the input to the QFT, $|\alpha\rangle = (\alpha_0, \alpha_1, \dots, \alpha_{M-1})$, is such that $\alpha_i = \alpha_j$ whenever $i \equiv j \pmod k$, where k is a particular integer that divides M . That is, the array α consists of M/k repetitions of some sequence $(\alpha_0, \alpha_1, \dots, \alpha_{k-1})$ of length k . Moreover, suppose that

exactly one of the k numbers $\alpha_0, \dots, \alpha_{k-1}$ is nonzero, say α_j . Then we say that $|\alpha\rangle$ is periodic with period k and offset j .

Figure 10.5 Examples of periodic superpositions.



It turns out that if the input vector is periodic, we can use quantum Fourier sampling to compute its period! This is based on the following fact, proved in the next box:

Suppose the input to quantum Fourier sampling is periodic with period k , for some k that divides M . Then the output will be a multiple of M/k , and it is equally likely to be any of the k multiples of M/k .

Now a little thought tells us that by repeating the sampling a few times (repeatedly preparing the periodic superposition and doing Fourier sampling), and then taking the greatest common divisor of all the indices returned, we will with very high probability get the number M/k —and from it the period k of the input!

The Fourier transform of a periodic vector

Suppose the vector $|\alpha\rangle = (\alpha_0, \alpha_1, \dots, \alpha_{M-1})$ is periodic with period k and with no offset (that is, the nonzero terms are $\alpha_0, \alpha_k, \alpha_{2k}, \dots$). Thus,

$$|\alpha\rangle = \sum_{j=0}^{M/k-1} \sqrt{\frac{k}{M}} |jk\rangle.$$

We will show that its Fourier transform $|\beta\rangle = (\beta_0, \beta_1, \dots, \beta_{M-1})$ is also periodic, with period M/k and no offset.

Claim $|\beta\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} | \frac{jM}{k} \rangle$.

Proof. In the input vector, the coefficient α_ℓ is $\sqrt{k/M}$ if k divides ℓ , and is zero otherwise. We can plug this into the formula for the j th coefficient of $|\beta\rangle$:

$$\beta_j = \frac{1}{\sqrt{M}} \sum_{\ell=0}^{M-1} \omega^{j\ell} \alpha_\ell = \frac{\sqrt{k}}{M} \sum_{i=0}^{M/k-1} \omega^{j ik}.$$

The summation is a geometric series, $1 + \omega^{jk} + \omega^{2jk} + \omega^{3jk} + \dots$, containing M/k terms and with ratio ω^{jk} (recall that ω is a complex M th root of unity). There are two cases. If the ratio is exactly 1, which happens if $jk \equiv 0 \pmod{M}$, then the sum of the series is simply the number of terms. If the ratio isn't 1, we can apply the usual formula for geometric series to find that the sum is $\frac{1 - \omega^{jk(M/k)}}{1 - \omega^{jk}} = \frac{1 - \omega^{Mj}}{1 - \omega^{jk}} = 0$.

Therefore β_j is $1/\sqrt{k}$ if M divides jk , and is zero otherwise. ■

More generally, we can consider the original superposition to be periodic with period k , but with some offset $l < k$:

$$|\alpha\rangle = \sum_{j=0}^{M/k-1} \sqrt{\frac{k}{M}} |jk + l\rangle.$$

Then, as before, the Fourier transform $|\beta\rangle$ will have nonzero amplitudes precisely at multiples of M/k :

Claim $|\beta\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} \omega^{ljM/k} | \frac{jM}{k} \rangle$.

The proof of this claim is very similar to the preceding one (Exercise 10.5).

We conclude that *the QFT of any periodic superposition with period k is an array that is everywhere zero, except at indices that are multiples of M/k , and all these k nonzero coefficients have equal absolute values*. So if we sample the output, we will get an index that is a multiple of M/k , and each of the k such indices will occur with probability $1/k$.

Let's make this more precise.

Lemma Suppose s independent samples are drawn uniformly from

$$0, \frac{M}{k}, \frac{2M}{k}, \dots, \frac{(k-1)M}{k}.$$

Then with probability at least $1 - k/2^s$, the greatest common divisor of these samples is M/k .

Proof. The only way this can fail is if all the samples are multiples of $j \cdot M/k$, where j is some integer greater than 1. So, fix any integer $j \geq 2$. The chance that a particular sample is a multiple of jM/k is at most $1/j \leq 1/2$; and thus the chance that *all* the samples are multiples of jM/k is at most $1/2^s$.

So far we have been thinking about a particular number j ; the probability that this bad event will happen for *some* $j \leq k$ is at most equal to the *sum* of these probabilities over the different values of j , which is no more than $k/2^s$. ■

We can make the failure probability as small as we like by taking s to be an appropriate multiple of $\log M$.

10.5 Quantum circuits

So quantum computers can carry out a Fourier transform exponentially faster than classical computers. But what do these computers actually look like? What is a *quantum circuit* made up of, and exactly how does it compute Fourier transforms so quickly?

10.5.1 Elementary quantum gates

An elementary quantum operation is analogous to an elementary gate like the AND or NOT gate in a classical circuit. It operates upon either a single qubit or two qubits. One of the most important examples is the Hadamard gate, denoted by H, which operates on a single qubit. On input $|0\rangle$, it outputs $H(|0\rangle) = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle$. And for input $|1\rangle$, $H(|1\rangle) = \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$. In pictures:

$$|0\rangle \text{---} \boxed{\text{H}} \text{---} \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|1\rangle \qquad |1\rangle \text{---} \boxed{\text{H}} \text{---} \frac{1}{\sqrt{2}}|0\rangle - \frac{1}{\sqrt{2}}|1\rangle$$

Notice that in either case, measuring the resulting qubit yields 0 with probability $1/2$ and 1 with probability $1/2$. But what happens if the input to the Hadamard gate is an arbitrary superposition $\alpha_0|0\rangle + \alpha_1|1\rangle$? The answer, dictated by the linearity of quantum physics, is the superposition $\alpha_0 H(|0\rangle) + \alpha_1 H(|1\rangle) = \frac{\alpha_0 + \alpha_1}{\sqrt{2}}|0\rangle + \frac{\alpha_0 - \alpha_1}{\sqrt{2}}|1\rangle$. And so, if we apply the Hadamard gate to the output of a Hadamard gate, it restores the qubit to its original state!

Another basic gate is the controlled-NOT, or CNOT. It operates upon two qubits, with the first acting as a control qubit and the second as the target qubit. The CNOT gate flips the second bit if and only if the first qubit is a 1. Thus $\text{CNOT}(|00\rangle) = |00\rangle$ and $\text{CNOT}(|10\rangle) = |11\rangle$:



Yet another basic gate, the controlled phase gate, is described below in the subsection describing the quantum circuit for the QFT.

Now let us consider the following question: Suppose we have a quantum state on n qubits, $|\alpha\rangle = \sum_{x \in \{0,1\}^n} \alpha_x |x\rangle$. How many of these 2^n amplitudes change if we apply the Hadamard gate to only the first qubit? The surprising answer is—all of them! The new superposition becomes $|\beta\rangle = \sum_{x \in \{0,1\}^n} \beta_x |x\rangle$, where $\beta_{0y} = \frac{\alpha_{0y} + \alpha_{1y}}{\sqrt{2}}$ and $\beta_{1y} = \frac{\alpha_{0y} - \alpha_{1y}}{\sqrt{2}}$. Looking at the results more closely, the quantum operation on the first qubit deals with each $n - 1$ bit suffix y separately. Thus the pair of amplitudes α_{0y} and α_{1y} are transformed into $(\alpha_{0y} + \alpha_{1y})/\sqrt{2}$ and $(\alpha_{0y} - \alpha_{1y})/\sqrt{2}$. This is exactly the feature that will give us an exponential speedup in the quantum Fourier transform.

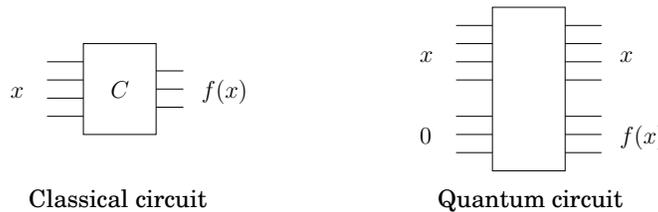
10.5.2 Two basic types of quantum circuits

A quantum circuit takes some number n of qubits as input, and outputs the same number of qubits. In the diagram these n qubits are carried by the n wires going from left to right. The quantum circuit consists of the application of a sequence of elementary quantum gates (of the kind described above) to single qubits and pairs of qubits.

At a high level, there are two basic functionalities of quantum circuits that we use in the design of quantum algorithms:

Quantum Fourier Transform These quantum circuits take as input n qubits in some state $|\alpha\rangle$ and output the state $|\beta\rangle$ resulting from applying the QFT to $|\alpha\rangle$.

Classical Functions Consider a function f with n input bits and m output bits, and suppose we have a classical circuit that outputs $f(x)$. Then there is a quantum circuit that, on input consisting of an n -bit string x padded with m 0's, outputs x and $f(x)$:

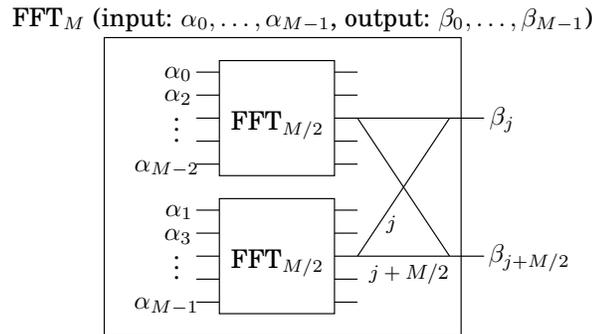


Now the input to this quantum circuit could be a superposition over the n bit strings x , $\sum_x |x, 0^k\rangle$, in which case the output has to be $\sum_x |x, f(x)\rangle$. Exercise 10.7 explores the construction of such circuits out of elementary quantum gates.

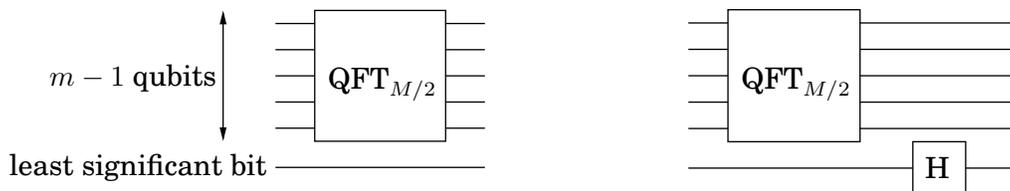
Understanding quantum circuits at this high level is sufficient to follow the rest of this chapter. The next subsection on quantum circuits for the QFT can therefore be safely skipped by anyone not wanting to delve into these details.

10.5.3 The quantum Fourier transform circuit

Here we have reproduced the diagram (from Section 2.6.4) showing how the classical FFT circuit for M -vectors is composed of two FFT circuits for $(M/2)$ -vectors followed by some simple gates.



Let's see how to simulate this on a quantum system. The input is now encoded in the 2^m amplitudes of $m = \log M$ qubits. Thus the decomposition of the inputs into evens and odds, as shown in the preceding figure, is clearly determined by one of the qubits—the least significant qubit. How do we separate the even and odd inputs and apply the recursive circuits to compute $FFT_{M/2}$ on each half? The answer is remarkable: just apply the quantum circuit $QFT_{M/2}$ to the remaining $m - 1$ qubits. The effect of this is to apply $QFT_{M/2}$ to the superposition of all the m -bit strings of the form $x0$ (of which there are $M/2$), and separately to the superposition of all the m -bit strings of the form $x1$. Thus the two recursive classical circuits can be emulated by a single quantum circuit—an exponential speedup when we unwind the recursion!



Let us now consider the gates in the classical FFT circuit *after* the recursive calls to $FFT_{M/2}$: the wires pair up j with $M/2 + j$, and ignoring for now the phase that is applied to the contents of the $(M/2 + j)$ th wire, we must add and subtract these two quantities to obtain the j th and the $(M/2 + j)$ th outputs, respectively. How would a quantum circuit achieve the result of these M classical gates? Simple: just perform the Hadamard gate on the first qubit! Recall from the preceding discussion (Section 10.5.1) that for every possible configuration of the remaining $m - 1$ qubits x , this pairs up the strings $0x$ and $1x$. Translating from binary, this means we are pairing up x and $M/2+x$. Moreover the result of the Hadamard gate is that for each such pair, the amplitudes are replaced by the sum and difference (normalized by $1/\sqrt{2}$), respectively. So far the QFT requires almost no gates at all!

The phase that must be applied to the $(M/2 + j)$ th wire for each j requires a little more work. Notice that the phase of ω^j must be applied only if the first qubit is 1. Now if j is

represented by the $m - 1$ bits $j_1 \dots j_{m-1}$, then $\omega^j = \prod_{l=1}^{m-1} \omega^{2^l j_l}$. Thus the phase ω^j can be applied by applying for the l th wire (for each l) a phase of ω^{2^l} if the l th qubit is a 1 and the first qubit is a 1. This task can be accomplished by another two-qubit quantum gate—the conditional phase gate. It leaves the two qubits unchanged unless they are both 1, in which case it applies a specified phase factor.

The QFT circuit is now specified. The number of quantum gates is given by the formula $S(m) = S(m-1) + O(m)$, which works out to $S(m) = O(m^2)$. The QFT on inputs of size $M = 2^m$ thus requires $O(m^2) = O(\log^2 M)$ quantum operations.

10.6 Factoring as periodicity

We have seen how the quantum Fourier transform can be used to find the period of a periodic superposition. Now we show, by a sequence of simple reductions, how factoring can be recast as a period-finding problem.

Fix an integer N . A *nontrivial square root of 1 modulo N* (recall Exercises 1.36 and 1.40) is any integer $x \not\equiv \pm 1 \pmod N$ such that $x^2 \equiv 1 \pmod N$. If we can find a nontrivial square root of 1 mod N , then it is easy to decompose N into a product of two nontrivial factors (and repeating the process would factor N):

Lemma *If x is a nontrivial square root of 1 modulo N , then $\gcd(x+1, N)$ is a nontrivial factor of N .*

Proof. $x^2 \equiv 1 \pmod N$ implies that N divides $(x^2 - 1) = (x+1)(x-1)$. But N does not divide either of these individual terms, since $x \not\equiv \pm 1 \pmod N$. Therefore N must have a nontrivial factor in common with each of $(x+1)$ and $(x-1)$. In particular, $\gcd(N, x+1)$ is a nontrivial factor of N . ■

Example. Let $N = 15$. Then $4^2 \equiv 1 \pmod{15}$, but $4 \not\equiv \pm 1 \pmod{15}$. Both $\gcd(4-1, 15) = 3$ and $\gcd(4+1, 15) = 5$ are nontrivial factors of 15.

To complete the connection with periodicity, we need one further concept. Define the *order* of x modulo N to be the smallest positive integer r such that $x^r \equiv 1 \pmod N$. For instance, the order of 2 mod 15 is 4.

Computing the order of a *random* number $x \pmod N$ is closely related to the problem of finding nontrivial square roots, and thereby to factoring. Here's the link.

Lemma *Let N be an odd composite, with at least two distinct prime factors, and let x be chosen uniformly at random between 0 and $N - 1$. If $\gcd(x, N) = 1$, then with probability at least $1/2$, the order r of $x \pmod N$ is even, and moreover $x^{r/2}$ is a nontrivial square root of 1 mod N .*

The proof of this lemma is left as an exercise. What it implies is that if we could compute the order r of a randomly chosen element $x \pmod N$, then there's a good chance that this order is even and that $x^{r/2}$ is a nontrivial square root of 1 modulo N . In which case $\gcd(x^{r/2} + 1, N)$ is a factor of N .

Example. If $x = 2$ and $N = 15$, then the order of 2 is 4 since $2^4 \equiv 1 \pmod{15}$. Raising 2 to half this power, we get a nontrivial root of 1: $2^2 \equiv 4 \not\equiv \pm 1 \pmod{15}$. So we get a divisor of 15 by computing $\gcd(4 + 1, 15) = 5$.

Hence we have reduced FACTORING to the problem of ORDER FINDING. The advantage of this latter problem is that it has a natural periodic function associated with it: fix N and x , and consider the function $f(a) = x^a \pmod{N}$. If r is the order of x , then $f(0) = f(r) = f(2r) = \dots = 1$, and similarly, $f(1) = f(r + 1) = f(2r + 1) = \dots = x$. Thus f is periodic, with period r . And we can compute it efficiently by the repeated squaring algorithm from Section 1.2.2. So, in order to factor N , all we need to do is to figure out how to use the function f to set up a periodic superposition with period r ; whereupon we can use quantum Fourier sampling as in Section 10.3 to find r . This is described in the next box.

Setting up a periodic superposition

Let us now see how to use our periodic function $f(a) = x^a \pmod{N}$ to set up a periodic superposition. Here is the procedure:

- We start with two quantum registers, both initially 0.
- Compute the quantum Fourier transform of the first register modulo M , to get a superposition over all numbers between 0 and $M - 1$: $\frac{1}{\sqrt{M}} \sum_{a=0}^{M-1} |a, 0\rangle$. This is because the initial superposition can be thought of as periodic with period M , so the transform is periodic with period 1.
- We now compute the function $f(a) = x^a \pmod{N}$. The quantum circuit for doing this regards the contents of the first register a as the input to f , and the second register (which is initially 0) as the answer register. After applying this quantum circuit, the state of the two registers is: $\sum_{a=0}^{M-1} \frac{1}{\sqrt{M}} |a, f(a)\rangle$.
- We now measure the second register. This gives a periodic superposition on the first register, with period r , the period of f . Here's why:

Since f is a periodic function with period r , for every r th value in the first register, the contents of the second register are the same. The measurement of the second register therefore yields $f(k)$ for some random k between 0 and $r - 1$. What is the state of the first register after this measurement? To answer this question, recall the rules of partial measurement outlined earlier in this chapter. The first register is now in a superposition of only those values a that are compatible with the outcome of the measurement on the second register. But these values of a are exactly $k, k + r, k + 2r, \dots, k + M - r$. So the resulting state of the first register is a periodic superposition $|\alpha\rangle$ with period r , which is exactly the order of x that we wish to find!

10.7 The quantum algorithm for factoring

We can now put together all the pieces of the quantum algorithm for FACTORING (see Figure 10.6). Since we can test in polynomial time whether the input is a prime or a prime power, we'll assume that we have already done that and that the input is an odd composite number with at least two distinct prime factors.

Input: an odd composite integer N .

Output: a factor of N .

1. Choose x uniformly at random in the range $1 \leq x \leq N - 1$.
2. Let M be a power of 2 near N (for reasons we cannot get into here, it is best to choose $M \approx N^2$).
3. Repeat $s = 2 \log N$ times:
 - (a) Start with two quantum registers, both initially 0, the first large enough to store a number modulo M and the second modulo N .
 - (b) Use the periodic function $f(a) \equiv x^a \pmod{N}$ to create a periodic superposition $|\alpha\rangle$ of length M as follows (see box for details):
 - i. Apply the QFT to the first register to obtain the superposition $\sum_{a=0}^{M-1} \frac{1}{\sqrt{M}} |a, 0\rangle$.
 - ii. Compute $f(a) = x^a \pmod{N}$ using a quantum circuit, to get the superposition $\sum_{a=0}^{M-1} \frac{1}{\sqrt{M}} |a, x^a \pmod{N}\rangle$.
 - iii. Measure the second register. Now the first register contains the periodic superposition $|\alpha\rangle = \sum_{j=0}^{M/r-1} \sqrt{\frac{r}{M}} |jr + k\rangle$ where k is a random offset between 0 and $r - 1$ (recall that r is the order of x modulo N).
 - (c) Fourier sample the superposition $|\alpha\rangle$ to obtain an index between 0 and $M - 1$.

Let g be the gcd of the resulting indices j_1, \dots, j_s .

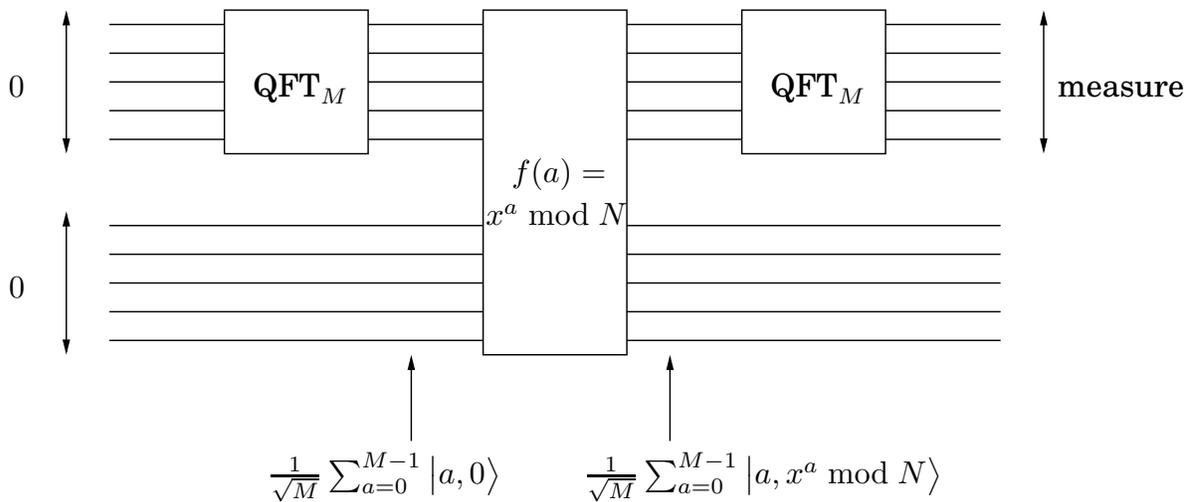
4. If M/g is even, then compute $\gcd(N, x^{M/2g} + 1)$ and output it if it is a nontrivial factor of N ; otherwise return to step 1.

From previous lemmas, we know that this method works for at least half the choices of x , and hence the entire procedure has to be repeated only a couple of times on average before a factor is found.

But there is one aspect of this algorithm, having to do with the number M , that is still quite unclear: M , the size of our FFT, must be a power of 2. And for our period-detecting idea to work, the period must divide M —hence it should also be a power of 2. But the period in our case is the order of x , definitely not a power of 2!

The reason it all works anyway is the following: *the quantum Fourier transform can detect the period of a periodic vector even if it does not divide M* . But the derivation is not as clean as in the case when the period does divide M , so we shall not go any further into this.

Figure 10.6 Quantum factoring.



Let $n = \log N$ be the number of bits of the input N . The running time of the algorithm is dominated by the $2 \log N = O(n)$ repetitions of step 3. Since modular exponentiation takes $O(n^3)$ steps (as we saw in Section 1.2.2) and the quantum Fourier transform takes $O(n^2)$ steps, the total running time for the quantum factoring algorithm is $O(n^3 \log n)$.

Quantum physics meets computation

In the early days of computer science, people wondered whether there were much more powerful computers than those made up of circuits composed of elementary gates. But since the seventies this question has been considered well settled. Computers implementing the von Neumann architecture on silicon were the obvious winners, and it was widely accepted that any other way of implementing computers is polynomially equivalent to them. That is, a T -step computation on any computer takes at most some polynomial in T steps on another. This fundamental principle is called *the extended Church-Turing thesis*. Quantum computers violate this fundamental thesis and therefore call into question some of our most basic assumptions about computers.

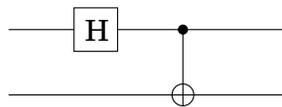
Can quantum computers be built? This is the challenge that is keeping busy many research teams of physicists and computer scientists around the world. The main problem is that quantum superpositions are very fragile and need to be protected from any inadvertent measurement by the environment. There is progress, but it is very slow: so far, the most ambitious reported quantum computation was the factorization of the number 15 into its factors 3 and 5 using nuclear magnetic resonance (NMR). And even in this experiment, there are questions about how faithfully the quantum factoring algorithm was really implemented. The next decade promises to be really exciting in terms of our ability to physically manipulate quantum bits and implement quantum computers.

But there is another possibility: What if all these efforts at implementing quantum computers fail? This would be even more interesting, because it would point to some fundamental flaw in quantum physics, a theory that has stood unchallenged for a century.

Quantum computation is motivated as much by trying to clarify the mysterious nature of quantum physics as by trying to create novel and superpowerful computers.

Exercises

- 10.1. $|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$ is one of the famous “Bell states,” a highly entangled state of its two qubits. In this question we examine some of its strange properties.
- (a) Suppose this Bell state could be decomposed as the (tensor) product of two qubits (recall the box on page 314), the first in state $\alpha_0|0\rangle + \alpha_1|1\rangle$ and the second in state $\beta_0|0\rangle + \beta_1|1\rangle$. Write four equations that the amplitudes $\alpha_0, \alpha_1, \beta_0,$ and β_1 must satisfy. Conclude that the Bell state cannot be so decomposed.
 - (b) What is the result of measuring the first qubit of $|\psi\rangle$?
 - (c) What is the result of measuring the second qubit after measuring the first qubit?
 - (d) If the two qubits in state $|\psi\rangle$ are very far from each other, can you see why the answer to (c) is surprising?
- 10.2. Show that the following quantum circuit prepares the Bell state $|\psi\rangle = \frac{1}{\sqrt{2}}|00\rangle + \frac{1}{\sqrt{2}}|11\rangle$ on input $|00\rangle$: apply a Hadamard gate to the first qubit followed by a CNOT with the first qubit as the control and the second qubit as the target.



What does the circuit output on input 10, 01, and 11? These are the rest of the Bell basis states.

- 10.3. What is the quantum Fourier transform modulo M of the uniform superposition $\frac{1}{\sqrt{M}} \sum_{j=0}^{M-1} |j\rangle$?
- 10.4. What is the QFT modulo M of $|j\rangle$?
- 10.5. *Convolution-Multiplication.* Suppose we shift a superposition $|\alpha\rangle = \sum_j \alpha_j |j\rangle$ by l to get the superposition $|\alpha'\rangle = \sum_j \alpha_j |j+l\rangle$. If the QFT of $|\alpha\rangle$ is $|\beta\rangle$, show that the QFT of α' is β' , where $\beta'_j = \beta_j \omega^{lj}$. Conclude that if $|\alpha'\rangle = \sum_{j=0}^{M/k-1} \sqrt{\frac{k}{M}} |jk+l\rangle$, then $|\beta'\rangle = \frac{1}{\sqrt{k}} \sum_{j=0}^{k-1} \omega^{ljM/k} |jM/k\rangle$.
- 10.6. Show that if you apply the Hadamard gate to the inputs and outputs of a CNOT gate, the result is a CNOT gate with control and target qubits switched:



- 10.7. The CONTROLLED SWAP (C-SWAP) gate takes as input 3 qubits and swaps the second and third if and only if the first qubit is a 1.
- (a) Show that each of the NOT, CNOT, and C-SWAP gates are their own inverses.
 - (b) Show how to implement an AND gate using a C-SWAP gate, i.e., what inputs a, b, c would you give to a C-SWAP gate so that one of the outputs is $a \wedge b$?
 - (c) How would you achieve fanout using just these three gates? That is, on input a and 0, output a and a .

- (d) Conclude therefore that for any classical circuit C there is an equivalent quantum circuit Q using just NOT and C-SWAP gates in the following sense: if C outputs y on input x , then Q outputs $|x, y, z\rangle$ on input $|x, 0, 0\rangle$. (Here z is some set of junk bits that are generated during this computation).
 - (e) Now show that there is a quantum circuit Q^{-1} that outputs $|x, 0, 0\rangle$ on input $|x, y, z\rangle$.
 - (f) Show that there is a quantum circuit Q' made up of NOT, CNOT, and C-SWAP gates that outputs $|x, y, 0\rangle$ on input $|x, 0, 0\rangle$.
- 10.8. In this problem we will show that if $N = pq$ is the product of two odd primes, and if x is chosen uniformly at random between 0 and $N - 1$, such that $\gcd(x, N) = 1$, then with probability at least $3/8$, the order r of $x \bmod N$ is even, and moreover $x^{r/2}$ is a nontrivial square root of 1 mod N .
- (a) Let p be an odd prime and let x be a uniformly random number modulo p . Show that the order of $x \bmod p$ is even with probability at least $1/2$. (*Hint*: Use Fermat's little theorem (Section 1.3).)
 - (b) Use the Chinese remainder theorem (Exercise 1.37) to show that with probability at least $3/4$, the order r of $x \bmod N$ is even.
 - (c) If r is even, prove that the probability that $x^{r/2} \equiv \pm 1$ is at most $1/2$.