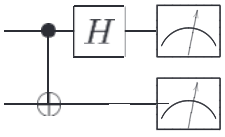
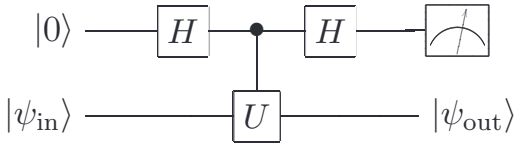
in the computational basis. However, often we want to perform a measurement in some other basis, defined by a complete set of orthonormal states. To perform this measurement, simply unitarily transform from the basis we wish to perform the measurement in to the computational basis, then measure. For example, show that the circuit

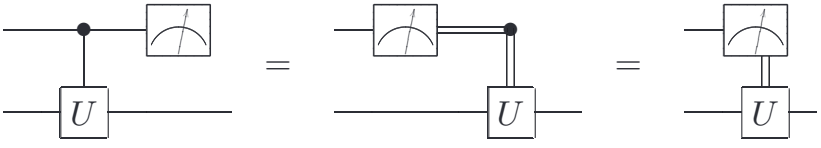


performs a measurement in the basis of the Bell states. More precisely, show that this circuit results in a measurement being performed with corresponding POVM elements the four projectors onto the Bell states. What are the corresponding measurement operators?

Exercise 4.34: (Measuring an operator) Suppose we have a single qubit operator U with eigenvalues ± 1, so that U is both Hermitian and unitary, so it can be regarded both as an observable and a quantum gate. Suppose we wish to measure the observable U. That is, we desire to obtain a measurement result indicating one of the two eigenvalues, and leaving a post-measurement state which is the corresponding eigenvector. How can this be implemented by a quantum circuit? Show that the following circuit implements a measurement of U:



Exercise 4.35: (Measurement commutes with controls) A consequence of the principle of deferred measurement is that measurements commute with quantum gates when the qubit being measured is a control qubit, that is:



(Recall that the double lines represent classical bits in this diagram.) Prove the first equality. The rightmost circuit is simply a convenient notation to depict the use of a measurement result to classically control a quantum gate.

4.5 Universal quantum gates

A small set of gates (e.g. AND, OR, NOT) can be used to compute an arbitrary classical function, as we saw in Section 3.1.2. We say that such a set of gates is *universal* for classical computation. In fact, since the Toffoli gate is universal for classical computation, quantum circuits subsume classical circuits. A similar universality result is true for quantum computation, where a set of gates is said to be *universal for quantum computation* if any unitary operation may be approximated to arbitrary accuracy by a quantum circuit

involving only those gates. We now describe three universality constructions for quantum computation. These constructions build upon each other, and culminate in a proof that any unitary operation can be approximated to arbitrary accuracy using Hadamard, phase, CNOT, and $\pi/8$ gates. You may wonder why the phase gate appears in this list, since it can be constructed from two $\pi/8$ gates; it is included because of its natural role in the fault-tolerant constructions described in Chapter 10.

The first construction shows that an arbitrary unitary operator may be expressed *exactly* as a product of unitary operators that each acts non-trivially only on a subspace spanned by two computational basis states. The second construction combines the first construction with the results of the previous section to show that an arbitrary unitary operator may be expressed *exactly* using single qubit and CNOT gates. The third construction combines the second construction with a proof that single qubit operation may be approximated to arbitrary accuracy using the Hadamard, phase, and $\pi/8$ gates. This in turn implies that any unitary operation can be approximated to arbitrary accuracy using Hadamard, phase, CNOT, and $\pi/8$ gates.

Our constructions say little about efficiency – how many (polynomially or exponentially many) gates must be composed in order to create a given unitary transform. In Section 4.5.4 we show that there *exist* unitary transforms which require exponentially many gates to approximate. Of course, the goal of quantum computation is to find interesting families of unitary transformations that *can* be performed efficiently.

Exercise 4.36: Construct a quantum circuit to add two two-bit numbers x and y modulo 4. That is, the circuit should perform the transformation $|x, y\rangle \rightarrow |x, x + y \bmod 4\rangle$.

4.5.1 Two-level unitary gates are universal

Consider a unitary matrix U which acts on a d-dimensional Hilbert space. In this section we explain how U may be decomposed into a product of *two-level unitary matrices*; that is, unitary matrices which act non-trivially only on two-or-fewer vector components. The essential idea behind this decomposition may be understood by considering the case when U is 3×3, so suppose that U has the form

$$U = \begin{bmatrix} a & d & g \\ b & e & h \\ c & f & j \end{bmatrix}. \tag{4.41}$$

We will find two-level unitary matrices $U_1, \ldots, U_3$ such that

$$U_3 U_2 U_1 U = I. \tag{4.42}$$

It follows that

$$U = U_1^\dagger U_2^\dagger U_3^\dagger. \tag{4.43}$$

U_1, U_2 and U_3 are all two-level unitary matrices, and it is easy to see that their inverses, $U_1^\dagger, U_2^\dagger$ and $U_3^\dagger$ are also two-level unitary matrices. Thus, if we can demonstrate (4.42), then we will have shown how to break U up into a product of two-level unitary matrices.

Use the following procedure to construct U_1: if $b = 0$ then set

$$U_1 \equiv \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.44}$$

If $b \neq 0$ then set

$$U_1 \equiv \begin{bmatrix} \frac{a^*}{\sqrt{|a|^2+|b|^2}} & \frac{b^*}{\sqrt{|a|^2+|b|^2}} & 0 \\ \frac{b}{\sqrt{|a|^2+|b|^2}} & \frac{-a}{\sqrt{|a|^2+|b|^2}} & 0 \\ 0 & 0 & 1 \end{bmatrix}. \tag{4.45}$$

Note that in either case U_1 is a two-level unitary matrix, and when we multiply the matrices out we get

$$U_1 U = \begin{bmatrix} a' & d' & g' \\ 0 & e' & h' \\ c' & f' & j' \end{bmatrix}. \tag{4.46}$$

The key point to note is that the middle entry in the left hand column is zero. We denote the other entries in the matrix with a generic prime $'$; their actual values do not matter.

Now apply a similar procedure to find a two-level matrix U_2 such that $U_2 U_1 U$ has no entry in the *bottom left* corner. That is, if $c' = 0$ we set

$$U_2 \equiv \begin{bmatrix} a'^* & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \tag{4.47}$$

while if $c' \neq 0$ then we set

$$U_2 \equiv \begin{bmatrix} \frac{a'^*}{\sqrt{|a'|^2+|c'|^2}} & 0 & \frac{c'^*}{\sqrt{|a'|^2+|c'|^2}} \\ 0 & 1 & 0 \\ \frac{c'}{\sqrt{|a'|^2+|c'|^2}} & 0 & \frac{-a'}{\sqrt{|a'|^2+|c'|^2}} \end{bmatrix}. \tag{4.48}$$

In either case, when we carry out the matrix multiplication we find that

$$U_2 U_1 U = \begin{bmatrix} 1 & d'' & g'' \\ 0 & e'' & h'' \\ 0 & f'' & j'' \end{bmatrix}. \tag{4.49}$$

Since U, U_1 and U_2 are unitary, it follows that $U_2 U_1 U$ is unitary, and thus $d'' = g'' = 0$, since the first row of $U_2 U_1 U$ must have norm 1. Finally, set

$$U_3 \equiv \begin{bmatrix} 1 & 0 & 0 \\ 0 & e''^* & f''^* \\ 0 & h''^* & j''^* \end{bmatrix}. \tag{4.50}$$

It is now easy to verify that $U_3 U_2 U_1 U = I$, and thus $U = U_1^\dagger U_2^\dagger U_3^\dagger$, which is a decomposition of U into two-level unitaries.

More generally, suppose U acts on a d-dimensional space. Then, in a similar fashion to the 3×3 case, we can find two-level unitary matrices $U_1, \ldots, U_{d-1}$ such that the matrix

$U_{d-1}U_{d-2}\ldots U_1U$ has a one in the top left hand corner, and all zeroes elsewhere in the first row and column. We then repeat this procedure for the $d-1$ by $d-1$ unitary submatrix in the lower right hand corner of $U_{d-1}U_{d-2}\ldots U_1U$, and so on, with the end result that an arbitrary $d\times d$ unitary matrix may be written

$$U = V_1 \ldots V_k, \tag{4.51}$$

where the matrices V_i are two-level unitary matrices, and $k \leq (d-1)+(d-2)+\cdots+1 = d(d-1)/2$.

Exercise 4.37: Provide a decomposition of the transform

$$\frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & i & -1 & -i \\ 1 & -1 & 1 & -1 \\ 1 & -i & -1 & i \end{bmatrix} \tag{4.52}$$

into a product of two-level unitaries. This is a special case of the quantum Fourier transform, which we study in more detail in the next chapter.

A corollary of the above result is that an arbitrary unitary matrix on an n qubit system may be written as a product of at most $2^{n-1}(2^n-1)$ two-level unitary matrices. For specific unitary matrices, it may be possible to find much more efficient decompositions, but as you will now show there exist matrices which *cannot* be decomposed as a product of fewer than $d-1$ two-level unitary matrices!

Exercise 4.38: Prove that there exists a $d\times d$ unitary matrix U which cannot be decomposed as a product of fewer than $d-1$ two-level unitary matrices.

4.5.2 Single qubit and CNOT gates are universal

We have just shown that an arbitrary unitary matrix on a d-dimensional Hilbert space may be written as a product of two-level unitary matrices. Now we show that single qubit and CNOT gates together can be used to implement an arbitrary two-level unitary operation on the state space of n qubits. Combining these results we see that single qubit and CNOT gates can be used to implement an arbitrary unitary operation on n qubits, and therefore are universal for quantum computation.

Suppose U is a two-level unitary matrix on an n qubit quantum computer. Suppose in particular that U acts non-trivially on the space spanned by the computational basis states $|s\rangle$ and $|t\rangle$, where $s = s_1 \ldots s_n$ and $t = t_1 \ldots t_n$ are the binary expansions for s and t. Let $\tilde{U}$ be the non-trivial 2×2 unitary submatrix of U; $\tilde{U}$ can be thought of as a unitary operator on a single qubit.

Our immediate goal is to construct a circuit implementing U, built from single qubit and CNOT gates. To do this, we need to make use of *Gray codes*. Suppose we have distinct binary numbers, s and t. A *Gray code* connecting s and t is a sequence of binary numbers, starting with s and concluding with t, such that adjacent members of the list differ in exactly one bit. For instance, with $s = 101001$ and $t = 110011$ we have the Gray

code

$$\begin{array}{cccccc} 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 0 & 1 & 1 \end{array} \tag{4.53}$$

Let g_1 through g_m be the elements of a Gray code connecting s and t, with $g_1 = s$ and $g_m = t$. Note that we can always find a Gray code such that $m \leq n+1$ since s and t can differ in at most n locations.

The basic idea of the quantum circuit implementing U is to perform a sequence of gates effecting the state changes $|g_1\rangle \rightarrow |g_2\rangle \rightarrow \ldots \rightarrow |g_{m-1}\rangle$, then to perform a controlled-$\tilde{U}$ operation, with the target qubit located at the single bit where g_{m-1} and g_m differ, and then to undo the first stage, transforming $|g_{m-1}\rangle \rightarrow |g_{m-2}\rangle \rightarrow \ldots \rightarrow |g_1\rangle$. Each of these steps can be easily implemented using operations developed earlier in this chapter, and the final result is an implementation of U.

A more precise description of the implementation is as follows. The first step is to swap the states $|g_1\rangle$ and $|g_2\rangle$. Suppose g_1 and g_2 differ at the ith digit. Then we accomplish the swap by performing a controlled bit flip on the ith qubit, conditional on the values of the other qubits being identical to those in both g_1 and g_2. Next we use a controlled operation to swap $|g_2\rangle$ and $|g_3\rangle$. We continue in this fashion until we swap $|g_{m-2}\rangle$ with $|g_{m-1}\rangle$. The effect of this sequence of $m-2$ operations is to achieve the operation

$$|g_1\rangle \rightarrow |g_{m-1}\rangle \tag{4.54}$$

$$|g_2\rangle \rightarrow |g_1\rangle \tag{4.55}$$

$$|g_3\rangle \rightarrow |g_2\rangle \tag{4.56}$$

$$\ldots\ldots\ldots$$

$$|g_{m-1}\rangle \rightarrow |g_{m-2}\rangle. \tag{4.57}$$

All other computational basis states are left unchanged by this sequence of operations. Next, suppose g_{m-1} and g_m differ in the jth bit. We apply a controlled-$\tilde{U}$ operation with the jth qubit as target, conditional on the other qubits having the same values as appear in both g_m and g_{m-1}. Finally, we complete the U operation by undoing the swap operations: we swap $|g_{m-1}\rangle$ with $|g_{m-2}\rangle$, then $|g_{m-2}\rangle$ with $|g_{m-3}\rangle$ and so on, until we swap $|g_2\rangle$ with $|g_1\rangle$.

A simple example illuminates the procedure further. Suppose we wish to implement the two-level unitary transformation

$$U = \begin{bmatrix} a & 0 & 0 & 0 & 0 & 0 & 0 & c \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ b & 0 & 0 & 0 & 0 & 0 & 0 & d \end{bmatrix}. \tag{4.58}$$

Here, a, b, c and d are any complex numbers such that $\tilde{U} \equiv \begin{bmatrix} a & c \\ b & d \end{bmatrix}$ is a unitary matrix.

Notice that U acts non-trivially only on the states $|000\rangle$ and $|111\rangle$. We write a Gray code connecting 000 and 111:

$$\begin{array}{ccc} A & B & C \\ 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 1 \end{array} . \tag{4.59}$$

From this we read off the required circuit, shown in Figure 4.16. The first two gates shuffle the states so that $|000\rangle$ gets swapped with $|011\rangle$. Next, the operation $\tilde{U}$ is applied to the first qubit of the states $|011\rangle$ and $|111\rangle$, conditional on the second and third qubits being in the state $|11\rangle$. Finally, we unshuffle the states, ensuring that $|011\rangle$ gets swapped back with the state $|000\rangle$.

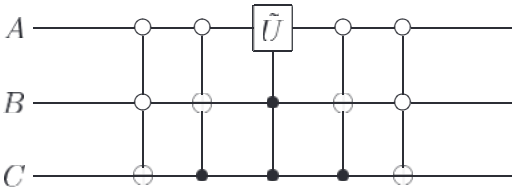


Figure 4.16. Circuit implementing the two-level unitary operation defined by (4.58).

Returning to the general case, we see that implementing the two-level unitary operation U requires at most $2(n-1)$ controlled operations to swap $|g_1\rangle$ with $|g_{m-1}\rangle$ and then back again. Each of these controlled operations can be realized using $O(n)$ single qubit and CNOT gates; the controlled-$\tilde{U}$ operation also requires $O(n)$ gates. Thus, implementing U requires $O(n^2)$ single qubit and CNOT gates. We saw in the previous section that an arbitrary unitary matrix on the 2^n-dimensional state space of n qubits may be written as a product of $O(2^{2n}) = O(4^n)$ two-level unitary operations. Combining these results, we see that an arbitrary unitary operation on n qubits can be implemented using a circuit containing $O(n^2 4^n)$ single qubit and CNOT gates. Obviously, this construction does not provide terribly efficient quantum circuits! However, we show in Section 4.5.4 that the construction is close to optimal in the sense that there are unitary operations that require an exponential number of gates to implement. Thus, to find fast quantum algorithms we will clearly need a different approach than is taken in the universality construction.

Exercise 4.39: Find a quantum circuit using single qubit operations and CNOTs to implement the transformation

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & a & 0 & 0 & 0 & 0 & c \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & b & 0 & 0 & 0 & 0 & d \end{bmatrix} , \tag{4.60}$$

where $\tilde{U} = \begin{bmatrix} a & c \\ b & d \end{bmatrix}$ is an arbitrary 2×2 unitary matrix.

4.5.3 A discrete set of universal operations

In the previous section we proved that the CNOT and single qubit unitaries together form a universal set for quantum computation. Unfortunately, no straightforward method is known to implement all these gates in a fashion which is resistant to errors. Fortunately, in this section we'll find a discrete set of gates which can be used to perform universal quantum computation, and in Chaper 10 we'll show how to perform these gates in an error-resistant fashion, using quantum error-correcting codes.

Approximating unitary operators

Obviously, a discrete set of gates can't be used to implement an arbitrary unitary operation *exactly*, since the set of unitary operations is continuous. Rather, it turns out that a discrete set can be used to *approximate* any unitary operation. To understand how this works, we first need to study what it means to approximate a unitary operation. Suppose U and V are two unitary operators on the same state space. U is the target unitary operator that we wish to implement, and V is the unitary operator that is actually implemented in practice. We define the *error* when V is implemented instead of U by

$$E(U,V) \equiv \max_{|\psi\rangle} \|(U-V)|\psi\rangle\|, \tag{4.61}$$

where the maximum is over all normalized quantum states $|\psi\rangle$ in the state space. In Box 4.1 on page 195 we show that this measure of error has the interpretation that if $E(U,V)$ is small, then any measurement performed on the state $V|\psi\rangle$ will give approximately the same measurement statistics as a measurement of $U|\psi\rangle$, for any initial state $|\psi\rangle$. More precisely, we show that if M is a POVM element in an arbitrary POVM, and P_U (or P_V) is the probability of obtaining this outcome if U (or V) were performed with a starting state $|\psi\rangle$, then

$$|P_U - P_V| \leq 2E(U,V)\,. \tag{4.62}$$

Thus, if $E(U,V)$ is small, then measurement outcomes occur with similar probabilities, regardless of whether U or V were performed. Also shown in Box 4.1 is that if we perform a sequence of gates $V_1,\ldots,V_m$ intended to approximate some other sequence of gates $U_1,\ldots,U_m$, then the errors add at most linearly,

$$E(U_m U_{m-1}\ldots U_1, V_m V_{m-1}\ldots V_1) \leq \sum_{j=1}^{m} E(U_j,V_j)\,. \tag{4.63}$$

The approximation results (4.62) and (4.63) are extremely useful. Suppose we wish to perform a quantum circuit containing m gates, U_1 through U_m. Unfortunately, we are only able to approximate the gate U_j by the gate V_j. In order that the probabilities of different measurement outcomes obtained from the approximate circuit be within a tolerance $\Delta > 0$ of the correct probabilities, it suffices that $E(U_j,V_j) \leq \Delta/(2m)$, by the results (4.62) and (4.63).

Universality of Hadamard + phase + CNOT + $\pi/8$ gates

We're now in a good position to study the approximation of arbitrary unitary operations by discrete sets of gates. We're going to consider two different discrete sets of gates, both

Box 4.1: Approximating quantum circuits

Suppose a quantum system starts in the state $|\psi\rangle$, and we perform either the unitary operation U, or the unitary operation V. Following this, we perform a measurement. Let M be a POVM element associated with the measurement, and let P_U (or P_V) be the probability of obtaining the corresponding measurement outcome if the operation U (or V) was performed. Then

$$|P_U - P_V| = \left|\langle\psi|U^\dagger M U|\psi\rangle - \langle\psi|V^\dagger M V|\psi\rangle\right|. \tag{4.64}$$

Let $|\Delta\rangle \equiv (U - V)|\psi\rangle$. Simple algebra and the Cauchy–Schwarz inequality show that

$$\begin{aligned} |P_U - P_V| &= \left|\langle\psi|U^\dagger M|\Delta\rangle + \langle\Delta|MV|\psi\rangle\right|. && (4.65)\\ &\leq |\langle\psi|U^\dagger M|\Delta\rangle| + |\langle\Delta|MV|\psi\rangle| && (4.66)\\ &\leq \||\Delta\rangle\| + \||\Delta\rangle\| && (4.67)\\ &\leq 2E(U, V). && (4.68) \end{aligned}$$

The inequality $|P_U - P_V| \leq 2E(U, V)$ gives quantitative expression to the idea that when the error $E(U, V)$ is small, the difference in probabilities between measurement outcomes is also small.

Suppose we perform a sequence $V_1, V_2, \ldots, V_m$ of gates intended to approximate some other sequence of gates, $U_1, U_2, \ldots, U_m$. Then it turns out that the error caused by the entire sequence of imperfect gates is at most the sum of the errors in the individual gates,

$$E(U_m U_{m-1} \ldots U_1, V_m V_{m-1} \ldots V_1) \leq \sum_{j=1}^{m} E(U_j, V_j). \tag{4.69}$$

To prove this we start with the case $m = 2$. Note that for some state $|\psi\rangle$ we have

$$\begin{aligned} E(U_2U_1, V_2V_1) &= \|(U_2U_1 - V_2V_1)|\psi\rangle\| && (4.70)\\ &= \|(U_2U_1 - V_2U_1)|\psi\rangle + (V_2U_1 - V_2V_1)|\psi\rangle\|. && (4.71) \end{aligned}$$

Using the triangle inequality $\||a\rangle + |b\rangle\| \leq \||a\rangle\| + \||b\rangle\|$, we obtain

$$\begin{aligned} E(U_2U_1, V_2V_1) &\leq \|(U_2 - V_2)U_1|\psi\rangle\| + \|V_2(U_1 - V_1)|\psi\rangle\| && (4.72)\\ &\leq E(U_2, V_2) + E(U_1, V_1), && (4.73) \end{aligned}$$

which was the desired result. The result for general m follows by induction.

of which are universal. The first set, the *standard set* of universal gates, consists of the Hadamard, phase, controlled-NOT and $\pi/8$ gates. We provide fault-tolerant constructions for these gates in Chapter 10; they also provide an exceptionally simple universality construction. The second set of gates we consider consists of the Hadamard gate, phase gate, the controlled-NOT gate, and the Toffoli gate. These gates can also all be done fault-tolerantly; however, the universality proof and fault-tolerance construction for these gates is a little less appealing.

We begin the universality proof by showing that the Hadamard and $\pi/8$ gates can be

used to approximate any single qubit unitary operation to arbitrary accuracy. Consider the gates T and HTH. T is, up to an unimportant global phase, a rotation by $\pi/4$ radians around the $\hat{z}$ axis on the Bloch sphere, while HTH is a rotation by $\pi/4$ radians around the $\hat{x}$ axis on the Bloch sphere (Exercise 4.14). Composing these two operations gives, up to a global phase,

$$\exp\left(-i\frac{\pi}{8}Z\right)\exp\left(-i\frac{\pi}{8}X\right) = \left[\cos\frac{\pi}{8}I - i\sin\frac{\pi}{8}Z\right]\left[\cos\frac{\pi}{8}I - i\sin\frac{\pi}{8}X\right] \quad (4.74)$$

$$= \cos^2\frac{\pi}{8}I - i\left[\cos\frac{\pi}{8}(X+Z) + \sin\frac{\pi}{8}Y\right]\sin\frac{\pi}{8}. \quad (4.75)$$

This is a rotation of the Bloch sphere about an axis along $\vec{n} = (\cos\frac{\pi}{8}, \sin\frac{\pi}{8}, \cos\frac{\pi}{8})$ with corresponding unit vector $\hat{n}$, and through an angle θ defined by $\cos(\theta/2) \equiv \cos^2\frac{\pi}{8}$. That is, using only the Hadamard and $\pi/8$ gates we can construct $R_{\hat{n}}(\theta)$. Moreover, this θ can be shown to be an irrational multiple of 2π. Proving this latter fact is a little beyond our scope; see the end of chapter 'History and further reading'.

Next, we show that repeated iteration of $R_{\hat{n}}(\theta)$ can be used to approximate to arbitrary accuracy any rotation $R_{\hat{n}}(\alpha)$. To see this, let $\delta > 0$ be the desired accuracy, and let N be an integer larger than $2\pi/\delta$. Define θ_k so that $\theta_k \in [0, 2\pi)$ and $\theta_k = (k\theta) \bmod 2\pi$. Then the pigeonhole principle implies that there are distinct j and k in the range $1, \ldots, N$ such that $|\theta_k - \theta_j| \leq 2\pi/N < \delta$. Without loss of generality assume that $k > j$, so we have $|\theta_{k-j}| < \delta$. Since $j \neq k$ and θ is an irrational multiple of 2π we must have $\theta_{k-j} \neq 0$. It follows that the sequence $\theta_{l(k-j)}$ fills up the interval $[0, 2\pi)$ as l is varied, so that adjacent members of the sequence are no more than δ apart. It follows that for any $\epsilon > 0$ there exists an n such that

$$E(R_{\hat{n}}(\alpha), R_{\hat{n}}(\theta)^n) < \frac{\epsilon}{3}. \quad (4.76)$$

Exercise 4.40: For arbitrary α and β show that

$$E(R_{\hat{n}}(\alpha), R_{\hat{n}}(\alpha+\beta)) = |1 - \exp(i\beta/2)|, \quad (4.77)$$

and use this to justify (4.76).

We are now in position to verify that any single qubit operation can be approximated to arbitrary accuracy using the Hadamard and $\pi/8$ gates. Simple algebra implies that for any α

$$HR_{\hat{n}}(\alpha)H = R_{\hat{m}}(\alpha), \quad (4.78)$$

where $\hat{m}$ is a unit vector in the direction $(\cos\frac{\pi}{8}, -\sin\frac{\pi}{8}, \cos\frac{\pi}{8})$, from which it follows that

$$E(R_{\hat{m}}(\alpha), R_{\hat{m}}(\theta)^n) < \frac{\epsilon}{3}. \quad (4.79)$$

But by Exercise 4.11 an arbitrary unitary U on a single qubit may be written as

$$U = R_{\hat{n}}(\beta)R_{\hat{m}}(\gamma)R_{\hat{n}}(\delta), \quad (4.80)$$

up to an unimportant global phase shift. The results (4.76) and (4.79), together with the

chaining inequality (4.63) therefore imply that for suitable positive integers n_1, n_2, n_3,

$$E(U, R_{\hat{n}}(\theta)^{n_1} H R_{\hat{n}}(\theta)^{n_2} H R_{\hat{n}}(\theta)^{n_3}) < \epsilon \,. \tag{4.81}$$

That is, given any single qubit unitary operator U and any $\epsilon > 0$ it is possible to approximate U to within ϵ using a circuit composed of Hadamard gates and $\pi/8$ gates alone.

Since the $\pi/8$ and Hadamard gates allow us to approximate any single qubit unitary operator, it follows from the arguments of Section 4.5.2 that we can approximate any m gate quantum circuit, as follows. Given a quantum circuit containing m gates, either CNOTs or single qubit unitary gates, we may approximate it using Hadamard, controlled-NOT and $\pi/8$ gates (later, we will find that phase gates make it possible to do the appoximation fault-tolerantly, but for the present universality argument they are not strictly necessary). If we desire an accuracy of ϵ for the entire circuit, then this may be achieved by approximating each single qubit unitary using the above procedure to within ϵ/m and applying the chaining inequality (4.63) to obtain an accuracy of ϵ for the entire circuit.

How efficient is this procedure for approximating quantum circuits using a discrete set of gates? This is an important question. Suppose, for example, that approximating an arbitrary single qubit unitary to within a distance ϵ were to require $\Omega(2^{1/\epsilon})$ gates from the discrete set. Then to approximate the m gate quantum circuit considered in the previous paragraph would require $\Omega(m2^{m/\epsilon})$ gates, an exponential increase over the original circuit size! Fortunately, the rate of convergence is much better than this. Intuitively, it is plausible that the sequence of angles θ_k 'fills in' the interval $[0, 2\pi)$ in a more or less uniform fashion, so that to approximate an arbitrary single qubit gate ought to take roughly $\Theta(1/\epsilon)$ gates from the discrete set. If we use this estimate for the number of gates required to approximate an arbitrary single qubit gate, then the number required to approximate an m gate circuit to accuracy ϵ becomes $\Theta(m^2/\epsilon)$. This is a quadratic increase over the original size of the circuit, m, which for many applications may be sufficient.

Rather remarkably, however, a much faster rate of convergence can be proved. The *Solovay–Kitaev theorem*, proved in Appendix 3, implies that an arbitrary single qubit gate may be approximated to an accuracy ϵ using $O(\log^c(1/\epsilon))$ gates from our discrete set, where c is a constant approximately equal to 2. The Solovay–Kitaev theorem therefore implies that to approximate a circuit containing m CNOTs and single qubit unitaries to an accuracy ϵ requires $O(m \log^c(m/\epsilon))$ gates from the discrete set, a polylogarithmic increase over the size of the original circuit, which is likely to be acceptable for virtually all applications.

To sum up, we have shown that the Hadamard, phase, controlled-NOT and $\pi/8$ gates are universal for quantum computation in the sense that given a circuit containing CNOTs and arbitrary single qubit unitaries it is possible to simulate this circuit to good accuracy using only this discrete set of gates. Moreover, the simulation can be performed efficiently, in the sense that the overhead required to perform the simulation is polynomial in $\log(m/\epsilon)$, where m is the number of gates in the original circuit, and ϵ is the desired accuracy of the simulation.

Exercise 4.41: This and the next two exercises develop a construction showing that the Hadamard, phase, controlled-NOT and Toffoli gates are universal. Show that

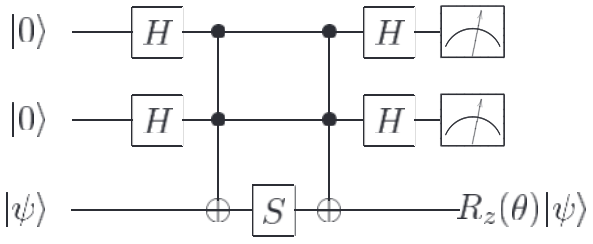


Figure 4.17. Provided both measurement outcomes are 0 this circuit applies $R_z(\theta)$ to the target, where $\cos\theta = 3/5$. If some other measurement outcome occurs then the circuit applies Z to the target.

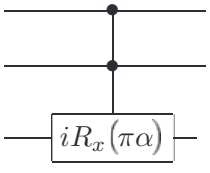
the circuit in Figure 4.17 applies the operation $R_z(\theta)$ to the third (target) qubit if the measurement outcomes are both 0, where $\cos\theta = 3/5$, and otherwise applies Z to the target qubit. Show that the probability of both measurement outcomes being 0 is $5/8$, and explain how repeated use of this circuit and $Z = S^2$ gates may be used to apply a $R_z(\theta)$ gate with probability approaching 1.

Exercise 4.42: (**Irrationality of θ**) Suppose $\cos\theta = 3/5$. We give a proof by contradiction that θ is an irrational multiple of 2π.

(1) Using the fact that $e^{i\theta} = (3 + 4i)/5$, show that if θ is rational, then there must exist a positive integer m such that $(3 + 4i)^m = 5^m$.
(2) Show that $(3 + 4i)^m = 3 + 4i \pmod 5$ for all $m > 0$, and conclude that no m such that $(3 + 4i)^m = 5^m$ can exist.

Exercise 4.43: Use the results of the previous two exercises to show that the Hadamard, phase, controlled-NOT and Toffoli gates are universal for quantum computation.

Exercise 4.44: Show that the three qubit gate G defined by the circuit:



is universal for quantum computation whenever α is irrational.

Exercise 4.45: Suppose U is a unitary transform implemented by an n qubit quantum circuit constructed from H, S, CNOT and Toffoli gates. Show that U is of the form $2^{-k/2}M$, for some integer k, where M is a $2^n \times 2^n$ matrix with only complex integer entries. Repeat this exercise with the Toffoli gate replaced by the $\pi/8$ gate.

4.5.4 Approximating arbitrary unitary gates is generically hard

We've seen that any unitary transformation on n qubits can be built up out of a small set of elementary gates. Is it always possible to do this efficiently? That is, given a unitary transformation U on n qubits does there always exist a circuit of size polynomial in n approximating U? The answer to this question turns out to be a resounding no: in fact, most unitary transformations can only be implemented very inefficiently. One way to see

this is to consider the question: how many gates does it take to generate an arbitrary state of n qubits? A simple counting argument shows that this requires exponentially many operations, in general; it immediately follows that there are unitary operations requiring exponentially many operations. To see this, suppose we have g different types of gates available, and each gate works on at most f input qubits. These numbers, f and g, are fixed by the computing hardware we have available, and may be considered to be constants. Suppose we have a quantum circuit containing m gates, starting from the computational basis state $|0\rangle^{\otimes n}$. For any particular gate in the circuit there are therefore at most $\left[\begin{array}{c} n \\ f \end{array} \right]^g = O(n^{fg})$ possible choices. It follows that at most $O(n^{fgm})$ different states may be computed using m gates.

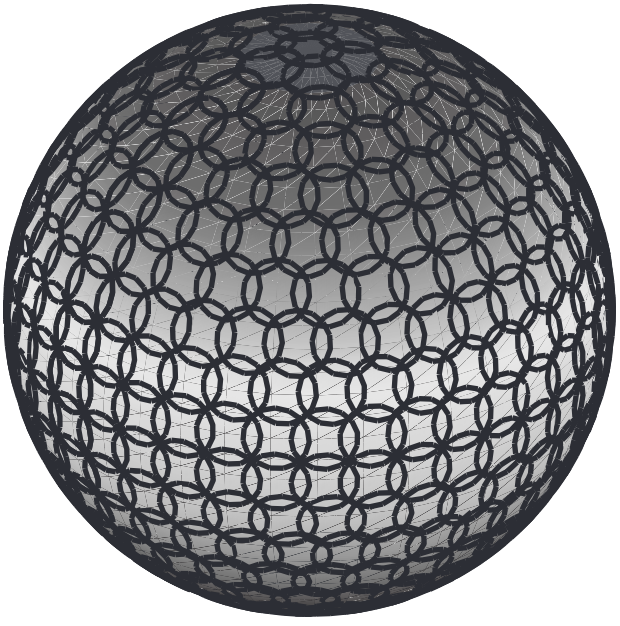


Figure 4.18. Visualization of covering the set of possible states with patches of constant radius.

Suppose we wish to approximate a particular state, $|\psi\rangle$, to within a distance ϵ. The idea of the proof is to cover the set of all possible states with a collection of 'patches,' each of radius ϵ (Figure 4.18), and then to show that the number of patches required rises doubly exponentially in n; comparing with the exponential number of different states that may be computed using m gates will imply the result. The first observation we need is that the space of state vectors of n qubits can be regarded as just the unit $(2^{n+1}-1)$-sphere. To see this, suppose the n qubit state has amplitudes $\psi_j = X_j + iY_j$, where X_j and Y_j are the real and imaginary parts, respectively, of the jth amplitude. The normalization condition for quantum states can be written $\sum_j (X_j^2 + Y_j^2) = 1$, which is just the condition for a point to be on the unit sphere in 2^{n+1} real dimensions, that is, the unit $(2^{n+1}-1)$-sphere. Similarly, the surface area of radius ϵ near $|\psi\rangle$ is approximately the same as the volume of a $(2^{n+1}-2)$-sphere of radius ϵ. Using the formula $S_k(r) = 2\pi^{(k+1)/2} r^k / \Gamma((k+1)/2)$ for the surface area of a k-sphere of radius r, and $V_k(r) = 2\pi^{(k+1)/2} r^{k+1} / [(k+1)\Gamma((k+1)/2)]$ for the volume of a k-sphere of radius r, we see that the number of patches needed to

cover the state space goes like

$$\frac{S_{2^{n+1}-1}(1)}{V_{2^{n+1}-2}(\epsilon)} = \frac{\sqrt{\pi}\Gamma(2^n - \frac{1}{2})(2^{n+1} - 1)}{\Gamma(2^n)\epsilon^{2^{n+1}-1}}, \tag{4.82}$$

where Γ is the usual generalization of the factorial function. But $\Gamma(2^n - 1/2) \geq \Gamma(2^n)/2^n$, so the number of patches required to cover the space is at least

$$\Omega\left(\frac{1}{\epsilon^{2^{n+1}-1}}\right). \tag{4.83}$$

Recall that the number of patches which can be reached in m gates is $O(n^{fgm})$, so in order to reach all the ϵ-patches we must have

$$O\left(n^{fgm}\right) \geq \Omega\left(\frac{1}{\epsilon^{2^{n+1}-1}}\right) \tag{4.84}$$

which gives us

$$m = \Omega\left(\frac{2^n \log(1/\epsilon)}{\log(n)}\right). \tag{4.85}$$

That is, there are states of n qubits which take $\Omega(2^n \log(1/\epsilon)/\log(n))$ operations to approximate to within a distance ϵ. This is exponential in n, and thus is 'difficult', in the sense of computational complexity introduced in Chapter 3. Furthermore, this immediately implies that there are unitary transformations U on n qubits which take $\Omega(2^n \log(1/\epsilon)/\log(n))$ operations to approximate by a quantum circuit implementing an operation V such that $E(U, V) \leq \epsilon$. By contrast, using our universality constructions and the Solovay–Kitaev theorem it follows that an arbitrary unitary operation U on n qubits may be approximated to within a distance ϵ using $O(n^2 4^n \log^c(n^2 4^n/\epsilon))$ gates. Thus, to within a polynomial factor the construction for universality we have given is optimal; unfortunately, it does not address the problem of determining which families of unitary operations can be computed efficiently in the quantum circuits model.

4.5.5 Quantum computational complexity

In Chapter 3 we described a theory of 'computational complexity' for classical computers that classified the resource requirements to solve computational problems on classical computers. Not surprisingly there is considerable interest in developing a theory of quantum computational complexity, and relating it to classical computational complexity theory. Although only first steps have been taken in this direction, it will doubtless be an enormously fruitful direction for future researchers. We content ourselves with presenting one result about quantum complexity classes, relating the quantum complexity class **BQP** to the classical complexity class **PSPACE**. Our discussion of this result is rather informal; for more details you are referred to the paper of Bernstein and Vazirani referenced in the end of chapter 'History and further reading'.

Recall that **PSPACE** was defined in Chapter 3 as the class of decision problems which can be solved on a Turing machine using space polynomial in the problem size and an arbitrary amount of time. **BQP** is an essentially quantum complexity class consisting of those decision problems that can be solved with bounded probability of error using a polynomial size quantum circuit. Slightly more formally, we say a language L is in **BQP** if there is a family of polynomial size quantum circuits which decides the language,

accepting strings in the language with probability at least $3/4$, and rejecting strings which aren't in the language with probability at least $3/4$. In practice, what this means is that the quantum circuit takes as input binary strings, and tries to determine whether they are elements of the language or not. At the conclusion of the circuit one qubit is measured, with 0 indicating that the string has been accepted, and 1 indicating rejection. By testing the string a few times to determine whether it is in L, we can determine with very high probability whether a given string is in L.

Of course, a quantum circuit is a fixed entity, and any given quantum circuit can only decide whether strings up to some finite length are in L. For this reason, we use an entire family of circuits in the definition of **BQP**; for every possible input length there is a different circuit in the family. We place two restrictions on the circuit in addition to the acceptance / rejection criterion already described. First, the size of the circuits should only grow polynomially with the size of the input string x for which we are trying to determine whether $x \in L$. Second, we require that the circuits be *uniformly generated*, in a sense similar to that described in Section 3.1.2. This uniformity requirement arises because, in practice, given a string x of some length n, somebody will have to build a quantum circuit capable of deciding whether x is in L. To do so, they will need to have a clear set of instructions – an algorithm – for building the circuit. For this reason, we require that our quantum circuits be uniformly generated, that is, there is a Turing machine capable of efficiently outputting a description of the quantum circuit. This restriction may seem rather technical, and in practice is nearly always satisfied trivially, but it does save us from pathological examples such as that described in Section 3.1.2. (You might also wonder if it matters whether the Turing machine used in the uniformity requirement is a quantum or classical Turing machine; it turns out that it doesn't matter – see 'History and further reading'.)

One of the most significant results in quantum computational complexity is that **BQP** $\subseteq$ **PSPACE**. It is clear that **BPP** $\subseteq$ **BQP**, where **BPP** is the classical complexity class of decision problems which can be solved with bounded probability of error using polynomial time on a classical Turing machine. Thus we have the chain of inclusions **BPP** $\subseteq$ **BQP** $\subseteq$ **PSPACE**. Proving that **BQP** $\neq$ **BPP** – intuitively the statement that quantum computers are more powerful than classical computers – will therefore imply that **BPP** $\neq$ **PSPACE**. However, it is not presently known whether **BPP** $\neq$ **PSPACE**, and proving this would represent a major breakthrough in classical computer science! So proving that quantum computers are more powerful than classical computers would have some very interesting implications for classical computational complexity! Unfortunately, it also means that providing such a proof may be quite difficult.

Why is it that **BQP** $\subseteq$ **PSPACE**? Here is an intuitive outline of the proof (a rigorous proof is left to the references in 'History and further reading'). Suppose we have an n qubit quantum computer, and do a computation involving a sequence of $p(n)$ gates, where $p(n)$ is some polynomial in n. Supposing the quantum circuit starts in the state $|0\rangle$ we will explain how to evaluate in polynomial space on a classical computer the probability that it ends up in the state $|y\rangle$. Suppose the gates that are executed on the quantum computer are, in order, $U_1, U_2, \ldots, U_{p(n)}$. Then the probability of ending up in the state $|y\rangle$ is the modulus squared of

$$\langle y|U_{p(n)} \cdots U_2 U_1|0\rangle \, . \tag{4.86}$$

This quantity may be estimated in polynomial space on a classical computer. The basic

idea is to insert the completeness relation $\sum_x |x\rangle\langle x| = I$ between each term in (4.86), obtaining

$$\langle y|U_{p(n)} \cdots U_2 U_1|0\rangle = \sum_{x_1,\ldots,x_{p(n)-1}} \langle y|U_{p(n)}|x_{p(n)-1}\rangle\langle x_{p(n)-1}|U_{p(n)-2} \ldots U_2|x_1\rangle\langle x_1|U_1|0\rangle \,. \tag{4.87}$$

Given that the individual unitary gates appearing in this sum are operations such as the Hadamard gate, CNOT, and so on, it is clear that each term in the sum can be calculated to high accuracy using only polynomial space on a classical computer, and thus the sum as a whole can be calculated using polynomial space, since individual terms in the sum can be erased after being added to the running total. Of course, this algorithm is rather slow, since there are exponentially many terms in the sum which need to be calculated and added to the total; however, only polynomially much space is consumed, and thus **BQP** $\subseteq$ **PSPACE**, as we set out to show.

A similar procedure can be used to simulate an *arbitrary* quantum computation on a classical computer, no matter the length of the quantum computation. Therefore, the class of problems solvable on a quantum computer with unlimited time and space resources is no larger than the class of problems solvable on a classical computer. Stated another way, this means that quantum computers do not violate the Church–Turing thesis that any algorithmic process can be simulated efficiently using a Turing machine. Of course, quantum computers may be much more *efficient* than their classical counterparts, thereby challenging the *strong* Church–Turing thesis that any algorithmic process can be simulated *efficiently* using a probabilistic Turing machine.

4.6 Summary of the quantum circuit model of computation

In this book the term 'quantum computer' is synonymous with the quantum circuit model of computation. This chapter has provided a detailed look at quantum circuits, their basic elements, universal families of gates, and some applications. Before we move on to more sophisticated applications, let us summarize the key elements of the quantum circuit model of computation:

(1) **Classical resources:** A quantum computer consists of two parts, a classical part and a quantum part. In principle, there is no need for the classical part of the computer, but in practice certain tasks may be made much easier if parts of the computation can be done classically. For example, many schemes for quantum error-correction (Chapter 10) are likely to involve classical computations in order to maximize efficiency. While classical computations can always be done, in principle, on a quantum computer, it may be more convenient to perform the calculations on a classical computer.

(2) **A suitable state space:** A quantum circuit operates on some number, n, of qubits. The state space is thus a 2^n-dimensional complex Hilbert space. Product states of the form $|x_1, \ldots, x_n\rangle$, where $x_i = 0, 1$, are known as *computational basis states* of the computer. $|x\rangle$ denotes a computational basis state, where x is the number whose binary representation is $x_1 \ldots x_n$.

(3) **Ability to prepare states in the computational basis:** It is assumed that any computational basis state $|x_1, \ldots, x_n\rangle$ can be prepared in at most n steps.