Last time we talked about systems with few qubits, mostly 1, 2, or 3. We found that, although qubits look continuous, you cannot extract all of the information contained therein.

We showed in "super-dense coding" (entanglement-assisted), that if you take a qubit and transmit it from a person A to a person B, then you can trasmit two classical bits of information.

This bound is tight and you cannot do any more than this. With quantum teleportation, we showed that if A wants to send her qubit to B, then only two bits of classical information need to be transferred. So the amount of information we can pack into a qubit is quite a subtle notion.

Today we will talk about:

1. How to deal with an arbitrary numbers of qubits.
2. Quantum circuits.
3. Complexity Classes.

The first bit is setup, and there's a lot of details that need to be covered first in order to have the right language to describe the more interesting things.

1 Systems of n qubits

Classically, a system of n bits would be described by an n-bit string, which we denote as $x \in \{0,1\}^n$. In quantum mechanics, a system of n qubits is described by a vector space with one basis element for each classical configuration x; the state is written as

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

Naively, this is why quantum computers are so great — because we have an exponentially larger space to work with, behind the scenes. Today we'll see how to think about this.

For this we need some formalism which many of us are already aware of from physics, but we'll review it for everyone else. The state of a single qubit is a unit vector $|\psi_1\rangle \in \mathbb{C}^2$. If we have two, then we have a unit vector $|\psi_1, \psi_2\rangle \in \mathbb{C}^4$. To combine these, we take the tensor product.

Classically, if one can specifiy the state of one system with k parameters and the state of a second system with ℓ parameters, then $k+\ell$ parameters is enough to completely specify the state of the composite system. However, in quantum mehcanics the amount of information *multiplies* instead. If we have quantum systems with k and ℓ levels, then their states are elements of $\mathbb{C}^k$ and $\mathbb{C}^\ell$ respectively, and states of the composite system are elements of $\mathbb{C}^k \otimes \mathbb{C}^\ell \cong \mathbb{C}^{k\times\ell}$.

More formally, suppose $\mathcal{H}_1 = \text{span}\{|v_1\rangle, |v_2\rangle, \ldots, |v_k\rangle\}$ and $\mathcal{H}_2 = \text{span}\{|w_1\rangle, |w_2\rangle, \ldots, |w_\ell\rangle\}$, then we can form *elementary tensors* $|v_i\rangle \otimes |w_j\rangle$, and their span gives us then new tensor product space. If the old bases were orthonormal, then the new one will be as well.

In terms of matrix notation, we will write kets as column vectors

$$|v\rangle = \alpha_1 |v_1\rangle + \cdots + \alpha_k |v_k\rangle = \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_k \end{bmatrix} \tag{2}$$

and

$$\langle w| = \langle w_1| \beta_1^* + \langle w_2| \beta_2^* + \cdots + \langle v_k| \beta_k^* = \begin{bmatrix} \beta_1 & \beta_2 & \cdots & \beta_k \end{bmatrix} \tag{3}$$

We represent the inner product as

$$\langle w|v\rangle = \begin{bmatrix} \beta_1 & \beta_2 & \cdots & \beta_k \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \alpha_k \end{bmatrix} = \sum_{i=1}^{k} \beta_k^* \alpha_k \tag{4}$$

The tensor product satisfies several properties:

1. Bilinear: $\alpha(|v\rangle \otimes |w\rangle) = \alpha |v\rangle \otimes \alpha |w\rangle$.

2. Distributive with the tensor product: $(|u\rangle + |v\rangle) \otimes |w\rangle = |u\rangle \otimes |w\rangle + |v\rangle \otimes |w\rangle$.

3. The inner product of the tensor product is the product of the inner products:

$$\Big(\langle u| \otimes \langle v| \Big) \Big(|x\rangle \otimes |y\rangle \Big) = \langle u|x\rangle \langle v|y\rangle . \tag{5}$$

Often we will save space by writing $|v\rangle \otimes |w\rangle = |vw\rangle$.

So, as an example, the basis vectors for a system of two qubits are $|00\rangle$, $|01\rangle$, $|10\rangle$, $|11\rangle$. Then a general state of two qubits can be written as

$$|\psi\rangle = \alpha_{00} |00\rangle + \alpha_{10} |10\rangle + \alpha_{01} |01\rangle + \alpha_{11} |11\rangle \tag{6}$$

where $\sum_{ij} |\alpha_{ij}|^2 = 1$.

Finally, we want to know how operators work with tensor products. Suppose we have two systems represented by the spaces $\mathcal{H}_1 = \mathbb{C}^k$ and $\mathcal{H}_2 = \mathbb{C}^\ell$. Suppose we have operators $A : H_1 \to H_1$ on the first space and $B : H_2 \to H_2$. We can then form a composite operator $A \otimes B$ that acts on the tensor product space $H_1 \otimes H_2$ simply by extending things by linearity:

$$(A \otimes B) \Big(|v_i\rangle \otimes |w_j\rangle \Big) = A |v_i\rangle \otimes B |w_j\rangle . \tag{7}$$

One can write tensor products with matrix notation using block vectors and matricies. For example, if

$$|v_2\rangle = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} \text{ and } |w_1\rangle = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \tag{8}$$

then

$$|v_2\rangle \otimes |w_1\rangle = \begin{bmatrix} 0 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \\ 1 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \\ 0 \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} \end{bmatrix} \tag{9}$$

Given two operators

$$A = \begin{bmatrix} a_{11} & \cdots & a_{1k} \\ \vdots & & \vdots \\ a_{k1} & \cdots & a_{kk} \end{bmatrix} \text{ and } B = \begin{bmatrix} b_{11} & \cdots & b_{1\ell} \\ \vdots & & \vdots \\ b_{\ell 1} & \cdots & b_{\ell\ell} \end{bmatrix} \tag{10}$$

the tensor product of operators can be written as

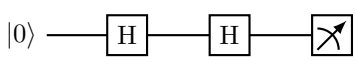
$$A \otimes B = \begin{bmatrix} a_{11}B & \cdots & a_{1k}B \\ \vdots & & \vdots \\ a_{k1}B & \cdots & a_{kk}B \end{bmatrix}, \tag{11}$$

where each of the elements is now a $\ell \times \ell$ block, giving an $(k\ell) \times (k\ell)$ matrix.

2 Quantum Circuits

2.1 Safe Storage

Let's consider two circuits. First, two Hadamard gates in a row, followed by a measurement.

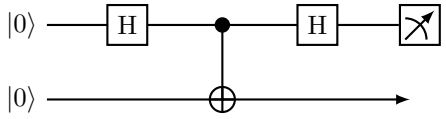


The result of this measurement will be 0 with probability 1 because $H^2 = I$, so the final state is $H^2 |0\rangle = |0\rangle$.

Alternatively, we could first apply a Hadamard gate, then measure, apply another Hadamard gate, and measure again.

After the first measurement, the state collapses into either $|0\rangle$ or $|1\rangle$ (in the standard basis), then after the second Hadamard gate, it will become $\frac{1}{\sqrt{2}} |0\rangle \pm \frac{1}{\sqrt{2}} |1\rangle$. In either case, the final measurement will yield 0 or 1 with equal probability. So as soon as we have measured, any initial information we had is destroyed. This could be a problem — what if we want to design an algorithm where we want to know the outcome of both the second and seventh steps? We can't measure the output after the second step because this would destroy the information we need for step three.

The technique of "safe storage" can bypass this limitation. Consider a third slightly different circuit with a CNOT gate instead of a measurement.



Before the CNOT gate the state of the system is $|0\rangle \otimes \left(\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle\right)$, and after the CNOT it becomes

$$\frac{1}{\sqrt{2}} |00\rangle + \frac{1}{\sqrt{2}} |11\rangle . \tag{12}$$

Once the final Hadamard gate is applied, it becomes

$$\frac{1}{\sqrt{2}} \left[\left(\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle \right) \otimes |0\rangle \right] + \frac{1}{\sqrt{2}} \left[\left(\frac{1}{\sqrt{2}} |0\rangle - \frac{1}{\sqrt{2}} |1\rangle \right) \otimes |1\rangle \right] . \tag{13}$$

If the top qubit is measured it will yield 0 or 1 with equal probabilities. Any time after the CNOT gate, the state of the bottom qubit is $\frac{1}{\sqrt{2}} |0\rangle + \frac{1}{\sqrt{2}} |1\rangle$, i.e. the state of the top qubit before the CNOT gate. Any further gates applied to the top qubit do not change this, so any measurement on the bottom would yield 0 or 1 with equal probabilities. Thus the effect of the CNOT gate is to create a "safe" version of the information contained in the top qubit, that can be measured at the end (or any other time of our convenience) without destroying the state.

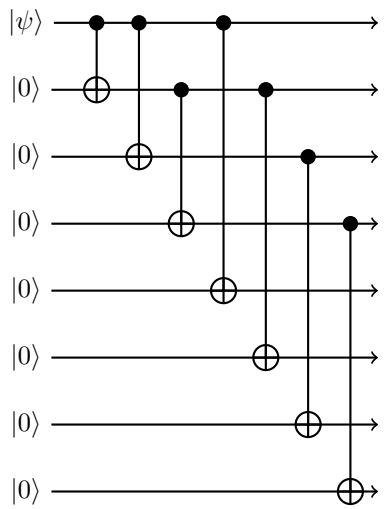
To sum up: we cannot measure the state of the system in the middle of the computation, so we instead join the state we want to measure to an ancilla qubit in the ground state with a CNOT gate, and that information will be safely preserved until the end without affecting anything else. If you want a slogan, we can say "interactions between systems are the same as measurements".

2.2 Measurements

Measurements are quite subtle, so here's a naive way to think about them. There probably exist situations where this fails, but this is a decent intuitive guide. Suppose we have a qubit $|\psi\rangle = \alpha |0\rangle + \beta |1\rangle$.

Take a qubit and apply CNOT gates over and over. This will replicate the state of your qubit first to two, then four, next to eight qubits, and so on, giving the state

$$\alpha |000000 \cdots 0\rangle + \beta |111111 \cdots 1\rangle . \tag{14}$$

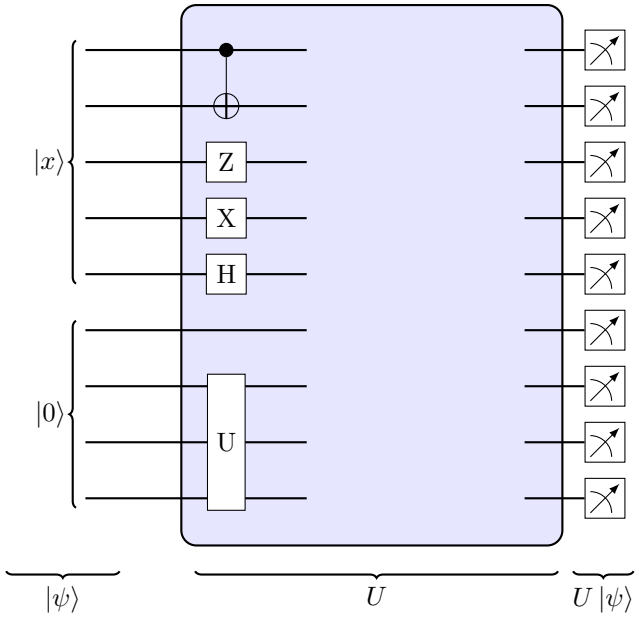


If you keep doing this doubling, perhaps 60 times, you'll get to Avagadro's number of entangled qubits. Any measurement device must include something which has this "repeated doubling" so you can go from a quantum state to classical one, where you can tick a geiger counter or push a needle or something. This is a cartoon picture, of course — one doesn't actually have CNOTs, but you do exponentially add entanglement. Eventually the macroscopic state will decohere and find itself as either zeros or ones in all of the qubits.

What exactly goes on in this "collapse"? Some people have various ideas, and some remain agnostic. For this class, we'll say "some magic happens" during the collapse and leave it at that.

2.3 General Quantum Circuits

Our cartoon of a general quantum circuit will be the following. Lots of lines going in, each representing a qubit. Some will be initialized to $|x\rangle$ as input data, and some will be ancillary $|0\rangle$'s as workspace. Since quantum computations must be reversible, we need to include all the storage we'll every need from the beginning. In the middle we'll have some arbitrary collection of gates, and then at the end we'll measure everything. For consistency, one thinks of measuring everything in the standard basis. One could in principle do some complicated joint basis, but this is in itself a computation, so we insist on measuring in the standard basis and having all the complexity in the middle. If the initial state is $|\psi\rangle$, we can collect all the gates as unitary transformation U, and then the end result will be $U|\psi\rangle$.



So now there's something very important to think about: if we have m qubits, then U can have $2^m \times 2^m = 2^{2m}$ complex parameters. Each gate corresponds to a constant number of parameters. So to be able to apply

any U we need to have *exponentially* many circuit elements. This could be an issue, because now some unitary transformations take exponentially more gates to implement. Effectively, this has given us a notion of how "complex" a unitary transformation is — the more gates needed to make it, the greater its complexity. Now not all unitary transformations are created equal.

But surely this depends on which circuit elements we chose! It turns out that there is a family of gates called *universal*, that allow you to approximate any arbitrary unitary transformation. An example of such as family is

$$\{\text{CNOT}, H, X, Z, R(\pi/8)\} \tag{15}$$

where the last corresponds to a rotation by an angle $\pi/8$. There is an important theorem that gives a *constructive* way to do this, which we state informally.

Solovay-Kitaev Theorem. Suppose we have a quantum system with states in $\mathbb{C}^d$. Take a universal family of gates, such as `CNOT`, H, X, Z, and $R(\pi/8)$. Then one can approximate any unitary operator U with accuracy ε with $O(d^2 \log^4 \frac{1}{\varepsilon})$ gates.

Here, "accuracy" is distance between the two operators in the operator norm. This turns out to be good enough! Since we're getting probabilistic answers anyway, it's usually fine if we're ε off in our transformation.

To sum up: most unitary transformations are too complex to actually use, so we should only simple ones. The theorem says we can — in principle — make anything by putting together a small number of simple gates.

2.4 Implementing Gates

It's worth noting that there's a careful point which historically hung the subject up for a long time. Initially the way we thought about quantum computing was as a very fragile thing. We thought the computation had to be completely isolated and the first time you interact is when you measure at the end. So we thought this meant the *controls* for the circuit had to be built into the circuit itself. If you sent a laser pulse at your qubit to flip the bit (implementing an X gate), then you would entangle your computation with the outside world and destroy it. People came up with schemes to embed a Turing machine or finite automata into the circuit so it would control itself, and other such ideas.

Today, however, the circuit model is envisaged as something very different. We have a set of qubits, and then a *classical* control which controls how, when, and which gates are applied. Why is it ok to do this?

Suppose we have hydrogen atom qubits, and we want to apply the X gate. This can be done by sending in a laser pulse with frequency tuned to take excited states down and ground states up. This looks like

$$(\alpha |0\rangle + \beta |1\rangle) |n\rangle \to (\alpha |1\rangle |n-1\rangle + \beta |0\rangle |n+1\rangle) . \tag{16}$$

The laser states are completely orthogonal, which is a problem; after the pulse hits, it looks like any measurement of the laser pulse (e.g. hitting an air molecule) will collapse the computer into either state $|0\rangle$ or $|1\rangle$, destroying the state of the system.

However, this is not actually what the laser's state looks like! Since the state of the laser is classical, it will be found in a coherent state that looks like $\sum_n \gamma_n |n\rangle$ where γ_n is a Gaussian peaked at some value n, which gives us, in the end,

$$\alpha |1\rangle \otimes \sum_n \gamma_n |n-1\rangle + \beta |0\rangle \otimes \sum_n \gamma_n |n+1\rangle \approx (\alpha |1\rangle + \beta |0\rangle) \otimes \left(\sum_n \gamma_n |n\rangle \right) . \tag{17}$$

The two states of the laser $\sum_n \gamma_n |n-1\rangle$ and $\sum_n \gamma_n |n+1\rangle$ are much closer to parallel than perpendicular. This means that if the state of the laser is measured, it will collapse the state of the system to $(\alpha \pm \varepsilon) |1\rangle + (\beta \pm \varepsilon) |0\rangle$, which is approximately the state we wanted. Therefore the classical part effectively stays separate and we are justified in controlling the gates from the outside.

This doesn't mean life is easy! We still absolutely need to prevent the qubits themselves from decohering through interactions, but at least we can implement the gates without worrying too much.

3 Complexity Theory

3.1 Overview of Complexity Classes

We now have a notion of how complex a unitary matrix is. We will now use this notion to begin to classify how hard various types of computation are. We now want to answer the following question: on input $x \in \{0,1\}^n$, we compute $f_n(x) \{0,1\}$. How hard is this? How many gates are necessary?

Let us informally define some classes of answers to this problem, some classical and some quantum.

Polynomial (P) Problems that can be answered by (classical) circuits with at most $P(n)$ gates, for some polynomial P (of arbitrary degree) are in this class.

Bounded-error Probabilistic Polynomial (BPP) Problems with inputs of length n that can be answered by (classical) circuits with at most $P(n)$ gates, but the answer is only correct with probability at least $2/3$. This is "fine", in the sense that you could run the algorithm k times and take the most common answer; after k runs, the probability of knowing the correct answer is at least $1 - 2^{-ck}$ for some $c > 0$.

Nondeterministic Polynomial (NP) Problems where one must decide if there exists a solution to a problem that can be verified quickly. A common example is the travelling salesperson problem: given a list of cities and the distances between each pair of cities, is there a route that visits each city exactly once before returning to the starting city, and has a length at most d? There are many, many possible routes to check, but checking if each route satisfies the criteria is easy.

NP Complete Problems that are in NP and such that every other type of problem in NP can be mapped to them. If there was an easy to solution to any one of these problems, then there would be an easy solution to all NP problems.

Co-NP Problems where one must decide if there does *not* exist a solution to a problem that can be verified in P.

Bounded-error Quantum Polynomial (BQP) Problems that can be answer by quantum circuits with at most $P(n)$ gates that give the correct answer with probability at least $2/3$. We further specifiy the circuits we use must be uniform: we should be able to quickly write down what they are.

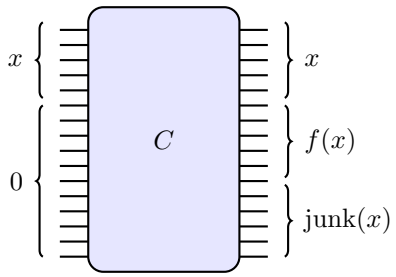
Showing which classes are contained in other classes is a difficult problem. One of the Millennium Problems is to determine if $\mathbf{P} = \mathbf{NP}$ or not.

This was a very quick overview. For more information on computational complexity, one could consult the Complexity Zoo at `https://complexityzoo.uwaterloo.ca/Complexity_Zoo`.

3.2 BPP $\subseteq$ BQP

The first people who thought about quantum computing worried that quantum effects would mean that quantum computation is actually *less* powerful and that with polynomially-many quantum gates, you couldn't even solve all the problems in **P**. Because all quantum computation involves unitary operations, it must necessarily be reversible, and we can always get the input back from the output. Naively, it seems like this imposes a strict constraint on what can actually be computed; in a classical circuit, as soon as we use an AND gate, we can't go backwards. It turns out, however, that every classical computation can be done in a reversible manner. One can then replace this reversible classical computation with its quantum analogue and, for any problem with a solution in **BPP**, find a solution in **BQP**. Let's see how this works.

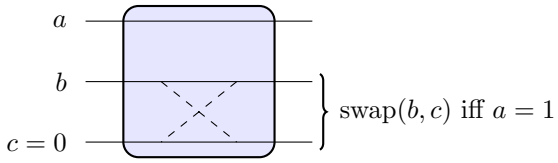
Let's suppose we have some classical circuit C. It might look like the following:



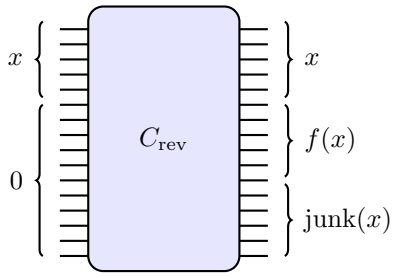
The best we might hope for is to replace "junk(x)" by zeros. The first step is to replace C by a reversible version. It turns out that, for any boolean gate, we can always make a reversible version. For instance, the AND gate, whose action on classical bits $a \in \{0, 1\}$ and $b \in \{0, 1\}$ is

$$a \text{ AND } b = \begin{cases} 1 & a = 1 \,\&\, b = 1 \\ 1 & a = 1 \,\&\, b = 0 \\ 1 & a = 0 \,\&\, b = 1 \\ 0 & a = 0 \,\&\, b = 0. \end{cases} \tag{18}$$

We can replaced this by a (classical) reversible AND gate (called a Fredkin gate).

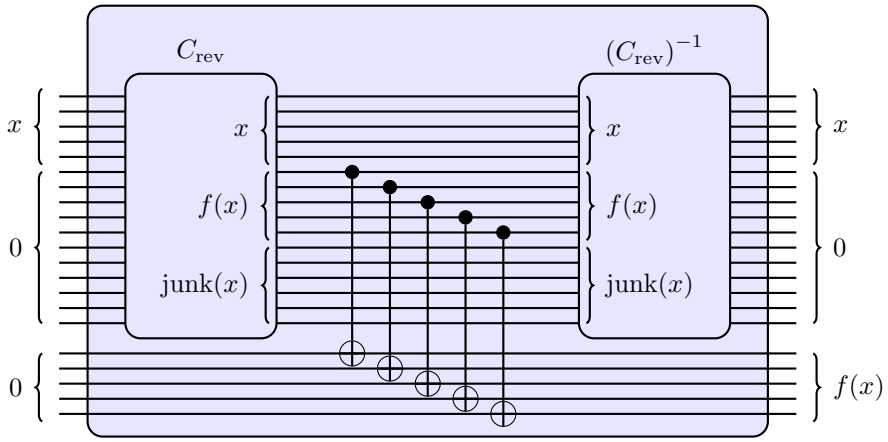


One can show that this type of gate is *universal*, in the sense that any boolean circuit can be replaced by an equivalent only made up only of reversible AND gates — and this new circuit must be reversible! Our original circuit C can then be replaced by a reversible version C_{rev} with the same inputs and outputs.

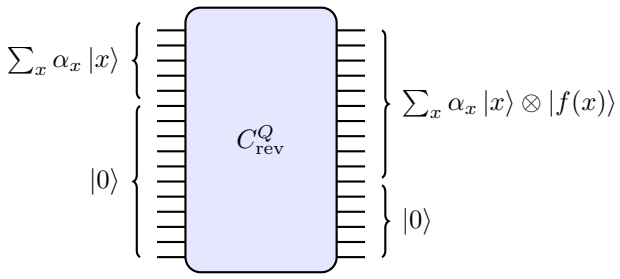


The junk(x) is still a problem. If we were to make a quantum version of this, then our answer would be something like $|f(x)\rangle \otimes |\text{junk}(x)\rangle$, and it wouldn't be possible to disentangle our answer from the junk. We need to zero it out.

To do this, we use the following gadget: first we get our answer reversibly, then copy the answer to waiting empty ancilae bits, then undo the computation reversibly to get the answer, the original input, and lots of zeros. All of this is reversible, so we have replaced our original circuit with a reversible circuit with no junk!



Once we've done this, it's easy to see how this could be performed by a quantum algorithm in BQP. Simply replace all of the reversible AND gates (or whatever we used) by a quantum analogue. This gives the following — it works the same way, except now we can do any linear combination of inputs at once!



This essentially shows **BPP** $\subseteq$ **BQP**.