CSE439/510 Week 4: Quantum Circuits (chapters 4 and 5)

The end of the last lecture set up how we will visualize series of quantum operations. The first point is that despite all the complicated machinations when we calculate tensor products of matrices and vectors, they represent merely the act of laying those operations side-by-side, each "in its own lane". Two basic facts:

- The tensor product of two unit vectors is a always a unit vector.
- The tensor product of two unitary matrices is always a unitary matrix.

For the second one, suppose $A \cdot B$ and $C \cdot D$ are compatible matrix products. We can give A indices i, j, give B indices j, k, give C indices ℓ, m, and give D indices m, n. Then $A \otimes C$ has indices $i\ell, jm$ and $B \otimes D$ has indices jm, kn. Therefore $E = (A \otimes C) \cdot (B \otimes D)$ is a compatible matrix product, and it comes out with indices $i\ell, kn$. The value of any particular entry is

$$E[i\ell, kn] \;=\; \sum_{jm} (A \otimes C)[i\ell, jm](B \otimes D)[jm, kn].$$

By definition of tensor product, $(A \otimes C)[i\ell, jm] = A[i, j]C[\ell, m]$ and similarly for $B \otimes D$. So we get

$$E[i\ell, kn] \;=\; \sum_{jm} A[i, j]C[\ell, m]B[j, k]D[m, n] \;=\; \sum_{jm} A[i, j]B[j, k]C[\ell, m]D[m, n].$$

This in turn equals $\left(\sum_j A[i, j]B[j, k]\right)\left(\sum_m C[\ell, m]D[m, n]\right) \;=\; (AB)[i, k](CD)[\ell, n]$. Again by definition of tensor product, this is the same as $(AB \otimes CD)[i\ell, kn]$. So $E = (AB) \otimes (CD)$. (Note the distinction between the use of the comma for array rows-versus-columns versus not using a comma when tensored coordinates are "rammed together.") The second fact then follows because if A and C are unitary, then

$$(A \otimes C)^*(A \otimes C) \;=\; \left(A^* \otimes C^*\right)(A \otimes C) \;=\; \left(A^*A \;\otimes\; C^*C\right) \;=\; (I \otimes I) \;=\; I.$$

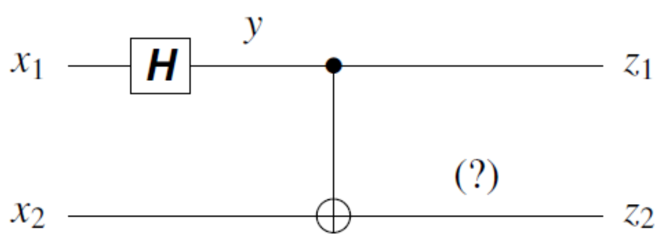
We will often use this silently when C itself is just the identity---a "non-gate" on one or more qubit "wires" running underneath the action of A on "upper" wires---and vice-versa when A is the identity. Despite all the fuss in our notation, for Nature this just means running things side-by-side. Now we will introduce the full formal notation for circuits of **qubits** and **gates**.

A First Quantum Circuit

At the end of the previous lecture, we did the computation of our basic entangled state:

$$\mathbf{CNOT}\cdot(\mathbf{H}\otimes\mathbf{I})|00\rangle = \begin{bmatrix}1&0&0&0\\0&1&0&0\\0&0&0&1\\0&0&1&0\end{bmatrix}\cdot\frac{1}{\sqrt{2}}\begin{bmatrix}1&0&1&0\\0&1&0&1\\1&0&-1&0\\0&1&0&-1\end{bmatrix} = \frac{1}{\sqrt{2}}\left[\begin{array}{c|ccc}1&0&1&0\\0&1&0&1\\0&1&0&-1\\1&0&-1&0\end{array}\right]\begin{bmatrix}1\\0\\0\\0\end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix}1\\0\\0\\1\end{bmatrix}.$$

When we do a quantum circuit left-to-right, however, the $(\mathbf{H}\otimes\mathbf{I})$ part comes first on the left. The symbol for a **CNOT** gate is to use a black dot to represent the control on the *source qubit* and $\oplus$ (which I have used as a symbol for XOR) on the *target qubit*. This is pictured by a **quantum circuit diagram**:



If $x_1 = |0\rangle$, then we can tell exactly what y is: it is the $|+\rangle$ state. And if $x_1 = |1\rangle$, then $y = |-\rangle$. If x_1 is any separate qubit state $(a,b) = ae_0 + be_1$, then by linearity we know that $y = a|+\rangle + b|-\rangle$. This expresses y over the transformed basis; in the standard basis it is

$$\frac{1}{\sqrt{2}}(a(1,1)+b(1,-1)) = \frac{1}{\sqrt{2}}(a+b, a-b)\ .$$

So we can say exactly what the input coming in to the first "wire" of the CNOT gate is. And the input to the second wire is just whatever x_2 is. But because that gate does entanglement, we cannot specify individual values for the wires coming *out*. The state is an inseparable 2-qubit state:

$$\frac{1}{\sqrt{2}}\big(|00\rangle + |11\rangle\big).$$

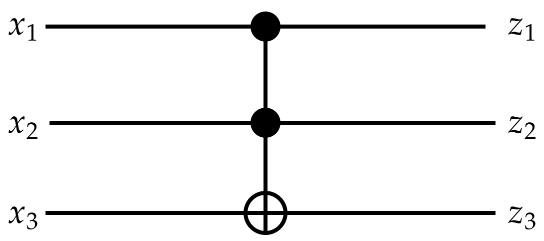
If you measure either qubit individually, you get 0 or 1 with equal probability. This is the same as if you measured the state $|{++}\rangle$. But that state is outwardly as well as inwardly different. When *both* qubits to be measured, it allows 01 and 10 as possible outcomes, whereas measuring the entangled state does not. I've seen papers telling ways to visualize entangled states of 2 or 3 qubits, but none implemented by an applet so far---Quirk just shows Bloch spheres with the yellow dot at the center for the "completely mixed state": | ¯_(ツ)_/¯ ⟩. (Note that you can save quantum circuits directly into URLs with Quirk---this convenience alone justifies the hassle of doing mental rotations and reversals to read its little-endian output.)

Three Qubits and More

The **CNOT** gate by itself has the logical description $z_1 = x_1$ and $z_2 = x_1 \oplus x_2$. This logical description is valid only for standard basis states. It means that if $x_1 = 0$ then $z_2 = x_2$, but if

$x_1 = 1$ then $z_2 = \neg x_2$. Since this description is complete for all of the standard basis inputs $x = x_1 x_2 = 00, 01, 10, 11$, it extends by linearity to all quantum states. We can use this idea to specify the 3-qubit **Toffoli gate** (**Tof**). It has inputs x_1, x_2, x_3 (representing the components in each basis state) and symbolic outputs z_1, z_2, z_3 (which, however, might not have individual values in non-basis cases owing to entanglement). Its spec in the basis quantum coordinates is:

$$z_1 = x_1, \qquad z_2 = x_2, \qquad z_3 = x_3 \oplus (x_1 \wedge x_2).$$



$$\left[\begin{array}{c|cccccccc}
 & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\
\hline
000 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
001 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\
010 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
011 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\
100 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\
101 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\
110 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\
111 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0
\end{array}\right]$$

Of particular note is that if x_3 is fixed to be a constant-1 input, then

$$z_3 = \neg(x_1 \wedge x_2) = NAND(x_1, x_2).$$

or rather

$$z_3 = x_3\, \boldsymbol{XOR}\, (x_1 \wedge x_2) = x_3\, XOR\, AND(x_1, x_2)$$

if $x_3 = 1$, then we get $1 \oplus (x_1 \wedge x_2) = \neg(x_1 \wedge x_2) = NAND(x_1 \wedge x_2)$.

Thus the Toffoli gate subsumes a classical NAND gate, except that you need an extra "helper wire" to put $x_3 = 1$ and you gate two extra output wires z_1, z_2 that only compute the identity on x_1, x_2 (in classical logic, that is---a non-basis quantum state can have knock-on effects even though all Toffoli does is switch the 7th and 8th components of the state vectors). If you have polynomially many Toffoli gates, then you get only polynomially much wastage of wires, and you can use the good ones to simulate any polynomial-size Boolean circuit of NAND gates.

We need to say more broadly what it means for quantum computations to be (polynomially) **feasible**. The community convention is simply to count up gates of 1, 2, or 3 qubits as constant cost. Gates involving more qubits are OK if they can be built up out of the small gates. We have already seen that $\mathbf{H}^{\otimes n}$ is just n binary Hadamard gates laid out in parallel. The n-qubit **quantum Fourier transform** can be built up out of $O(n^2)$ smaller gates---this actually has more "fine print" than sources usually say and is pursued in the chapter exercises of the textbook.

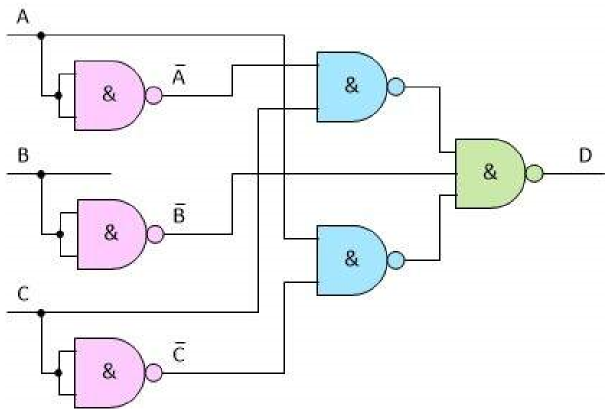
We should describe measurements in more detail and see smaller-scale deterministic and randomized examples first.

Quantum Circuit Examples

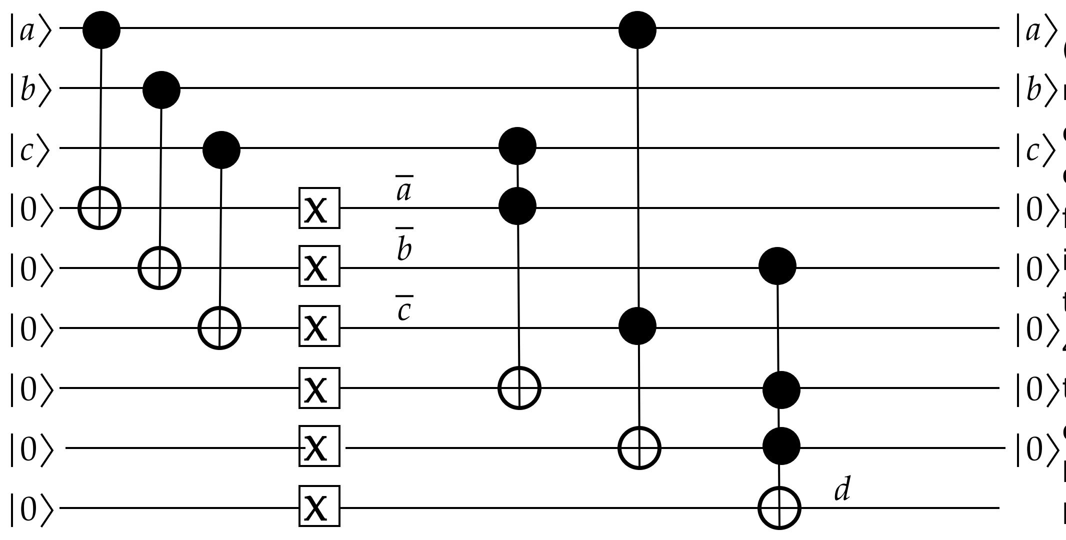
Theorem (cf. theorem 5.2 in section 5.3): Classical Boolean circuits can be efficiently simulated by quantum circuits that don't even use any superposition or entanglement or randomness.

The proof is basically that the Toffoli gate simulates NAND via $\mathrm{Tof}(x, y, 1) = (\overline{x} \vee \overline{y})$ and NAND is a universal gate. The extra lines for the constant 1 inputs also make the whole computation **reversible**. That is, $\mathrm{Tof}(x, y, z) = (x, y, z \oplus (\overline{x} \vee \overline{y}))$ is reversible. [RevNAND$(x, y) = (x, \overline{x} \vee \overline{y})$? (no, not reversible)]

Here is a sizable example of this theorem. Consider the following circuit of NAND gates from the blog article "Implementing Logic Functions Using Only NAND or NOR Gates" by Max Maxfield:



Here is the corresponding quantum circuit:



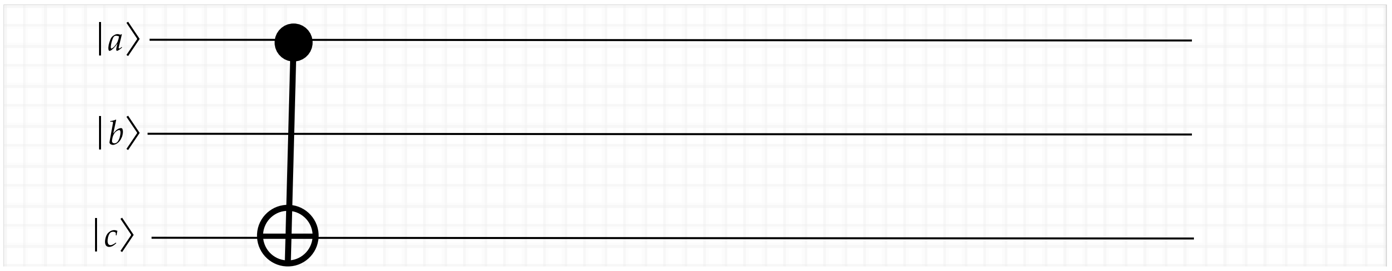
(We will later mirror the gates except the last one giving the function value d in order to reset the ancilla qubits 4--8 to $|0\rangle$, so that all qubits except the last keep their given basis values.)

Note also that the initial three CNOT gates effectively copy the Boolean values a, b, c so that they can be negated as $\bar{a}, \bar{b}, \bar{c}$ on the next three qubit lines. This is covered in section 6.2, and the last three qubit lines exemplify the trick in section 6.1 of using NOT gates to effectively initialize them to $|1\rangle$ rather than $|0\rangle$. *Caveat:* You can't copy an arbitrary quantum state using CNOT---the **No-Cloning Theorem** mentioned in section 6.2 shows there is no way to do this in general. But particular states in a known basis can be copied this way.

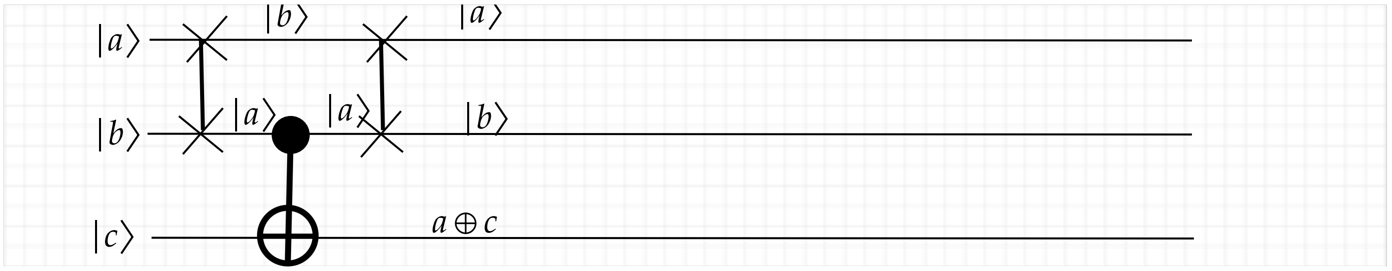
The "quantum extra", beginning with using the Hadamard gate to create superpositions, is what promises to take us beyond classical computing.

Circuits and Computations

Just like music can be divided into measures with a basic 'beat' unit, quantum gates going left to right are timesteps of a computation. If multiple gates are underneath each other, then they make a single tensor-producted operation---such as $\mathbf{X}^{\otimes 6}$ in the above diagram. If nothing happens on a certain qubit line at a given timestep, that is mathematically like tensoring with the identity matrix. A "squidgy" point has to do with the nearest-neighbor aspect of tensor products. Consider:



There is notation for $\mathbf{I} \otimes \mathrm{CNOT}$ and $\mathrm{CNOT} \otimes \mathbf{I}$, but not for "$\mathbf{I}$ in the middle." We can ignore this problem. Or---and often this has to be done with real tech---we can suppose the Swap gate is applied twice, e.g.



$$\mathrm{SWAP} = \left[\begin{array}{ccccc} & 00 & 01 & 10 & 11 \\ 00 & 1 & 0 & 0 & 0 \\ 01 & 0 & 0 & 1 & 0 \\ 10 & 0 & 1 & 0 & 0 \\ 11 & 0 & 0 & 0 & 1 \end{array}\right]$$

In such manner, we get the n-qubit circuit as a composition

$$C = U_t \circ U_{t-1} \circ \cdots \circ U_1$$

of $N \times N$ unitary matrices.

Principle of Linearity: For any quantum state $\boldsymbol{\Phi} = \sum_{i=0}^{N-1} a_i \mathbf{e}_i$,

$$C(\boldsymbol{\Phi}) = \sum_{i=0}^{N-1} a_i \, C\mathbf{e}_i \,.$$

In words, the action of a quantum circuit on any quantum state is determined by its actions on the (standard) basis states.

General Controlled Gates

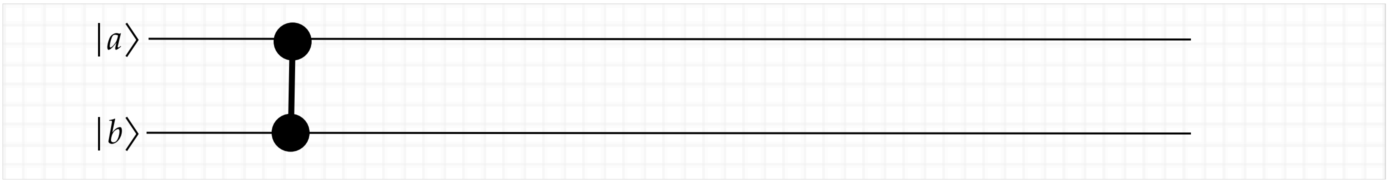
Related to the $\mathbf{CNOT}$ gate is the controlled version of the Z gate. Recall $\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$. The controlled version of any matrix A (in the standard basis) is the block matrix

$$\mathbf{C}A = \begin{array}{c|c|c} & 0u & 1u \\ \hline 0u & \mathbf{I} & 0 \\ \hline 1u & 0 & A \end{array},$$

where the hierarchical quantum indexing scheme is also shown. If the first qubit is 0 then the effect is the identity, while if it is 1, then the effect on the remainder $|u\rangle$ is to apply A. So

$$\mathbf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}.$$

Although the control is nominally on the first qubit, with $\mathbf{CZ}$ the effect on base states is to multiply the global state by -1 if and only if *both* qubits are 1. Hence it is really symmetric between qubits---the second qubit could equally be said to be controlling the first. The standard diagram for it is just two black dots connected by themselves:



Since a general vector $[u_1, u_2, u_3, u_4]^T$ becomes $[u_1, u_2, u_3, -u_4]^T$ after going through **CZ**, it follows, upon writing $|a\rangle = [a_1, a_2]^T$ and $|b\rangle = [b_1, b_2]^T$, that

$$\mathbf{CZ} \cdot \left(|a\rangle \otimes |b\rangle\right) = \mathbf{CZ} \cdot [a_1b_1, a_1b_2, a_2b_1, a_2b_2]^T = [a_1b_1, a_1b_2, a_2b_1, -a_2b_2]^T.$$

Is this ever entangled, and if so, when? Note that if $|a\rangle$ and $|b\rangle$ are both $|1\rangle$, then
$\mathbf{CZ} \cdot \left(|a\rangle \otimes |b\rangle\right) = \mathbf{CZ}|11\rangle = \mathbf{CZ} \cdot [0,0,0,1]^T = [0,0,0,-1] = -[0,0,0,1] = -|11\rangle.$

$$\mathbf{CZ} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix} \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \end{bmatrix} = \frac{1}{2} \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \end{bmatrix}$$

To try to represent this as a tensor product $\begin{bmatrix} e \\ f \end{bmatrix} \otimes \begin{bmatrix} g \\ h \end{bmatrix} = [eg, eh, fg, fh]^T$, we need both e and g to be 0, so we are left with $fh = -1$. This is easy to solve with $f = 1$ and $h = -1$, or even $f = h = i$ since we can use complex numbers.

But now let $|a\rangle$ and $|b\rangle$ both be $|+\rangle$. Then we get

$$\mathbf{CZ}|{+}{+}\rangle = \mathbf{CZ} \cdot \tfrac{1}{2}[1,1,1,1]^T = \tfrac{1}{2}[1,1,1,-1]^T.$$

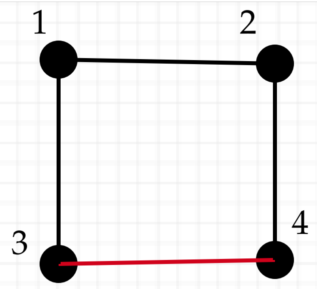
Can this be given the tensor-product form $[eg, eh, fg, fh]^T$? We can ignore the $\frac{1}{2}$ factor. So the equations become $eg = 1$, $eh = 1$, $fg = 1$, and $fh = -1$. The first three combine to give $g = \dfrac{1}{e} = h$, so $fg = fh = 1$, but that contradicts the fourth equation $fh = -1$. Thus $\mathbf{CZ}|{+}{+}\rangle$ is entangled.

We have exemplified a consequence that is important to bear in mind: Quantum operations do not simply flip a switch between "entangled" and "separable". Whether the output is entangled need not depend alone on whether the input is entangled or not:

It is possible for a quantum gate to leave one separable state separable while making another separable state become entangled.

Graphs and Classical and Quantum Representations

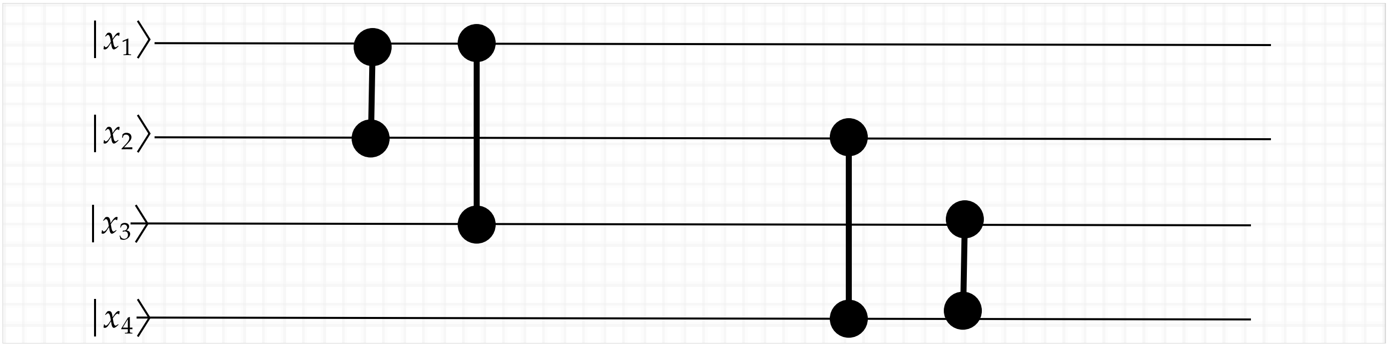
Now CZ gates are especially neat because they look like edges in a **graph** $G = (V, E)$, specifically an **undirected** graph because the gates are symmetric. Let's first see some examples of graphs. The *cycle graphs* C_k have k **vertices** (also called **nodes**) and k edges connecting them in a ring, for $k \geq 3$. The four-cycle graph has the following picture and **adjacency matrix**:



$$A = \begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \quad A' = \tfrac{1}{2}A \qquad A'' = \frac{1}{\sqrt{2}}\begin{bmatrix} 0 & 1 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & -1 \\ 0 & 1 & -1 & 0 \end{bmatrix}$$

Note: This differs from the text only in the labels 3 and 4. This makes it maybe easier to see that not only is A not unitary, it isn't even invertible: rows 2 and 3, and rows 1 and 4, are identical. But:

- A is a real symmetric matrix, so it is Hermitian.
- A' is a matrix of nonnegative entries each of whose rows and columns sums to 1, which makes it **doubly stochastic**. This is an analogue of "unitary" for classical probability.
- In fact, for any **regular** graph, meaning that all vertices have the same **degree** d, dividing the adjacency matrix by d always gives a doubly-stochastic matrix.
- We can in fact make a unitary matrix A'' by flipping the sign of the two 1s at lower right and dividing by $\sqrt{2}$ rather than by 2. This is, however, more of a coincidence than a general feature. The text shows that in the case of the regular *prism graph* ($n = 6$, $d = 3$), there is no sensible way to make it into a unitary matrix.
- The general way to encode graphs into quantum circuits via the $\mathbf{CZ}$ gate yield much bigger underlying matrices---and some surprises. Here we go:



When put on four qubits, the first gate gives the matrix $\mathbf{CZ} \otimes \mathbf{I} \otimes \mathbf{I}$, which we know how to build: replace every entry of the CZ matrix by the 4×4 identity matrix, to get the 16×16 matrix

$$\left|\begin{array}{c}0000\\0001\\0010\\0011\\0100\\0101\\0110\\0111\\1000\\1001\\1010\\1011\\1100\\1101\\1110\\1111\end{array}\right| : \quad \mathbf{CZ}\otimes\mathbf{I}\otimes\mathbf{I} = \begin{bmatrix}
1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&-1&0&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&-1&0&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&-1&0\\
0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&-1
\end{bmatrix} = diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\-1\\-1\end{bmatrix}\right)$$

At far left I've put the labels of the underlying coordinates by the sixteen basis strings of length 4. The point is that the -1 entries go in all the places where the first two bits of the string are 1 as shown in pink. This is because the first CZ gate is on the first two bits. Next, for the gate on qubits 1 and 3, we follow the same rule but for the coordinates where the first and third bit are 1:

$$\left|\begin{array}{c}0000\\0001\\0010\\0011\\0100\\0101\\0110\\0111\\1000\\1001\\1010\\1011\\1100\\1101\\1110\\1111\end{array}\right| : \quad diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\1\\1\\-1\\-1\end{bmatrix}\right).$$

$$= \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&-1&0&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&-1&0&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&-1&0\\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&-1 \end{bmatrix}$$

Here is the product of all four gate matrices that we get. I've "properly" put the matrix for the first gate on the right now, but actually this doesn't matter---they are all diagonal matrices so they commute with each other. To multiply them, we can just multiply the entries in each of the sixteen rows. The blue 1s show cases where an even number of -1 entries multiplied to give $+1$:

$$\begin{vmatrix} 0000\\0001\\0010\\0011\\0100\\0101\\0110\\0111\\1000\\1001\\1010\\1011\\1100\\1101\\1110\\1111 \end{vmatrix} : \; diag\left(\begin{bmatrix}1\\1\\1\\-1\\1\\1\\1\\-1\\1\\1\\1\\-1\\1\\1\\1\\-1\end{bmatrix}\right) \cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\-1\\1\\-1\\1\\1\\1\\1\\1\\-1\\1\\-1\end{bmatrix}\right) \; diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\1\\1\\-1\\-1\end{bmatrix}\right) \cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\-1\\-1\end{bmatrix}\right) = diag\left(\begin{bmatrix}1\\1\\1\\-1\\1\\-1\\1\\1\\1\\1\\-1\\1\\-1\\1\\1\\1\end{bmatrix}\right).$$

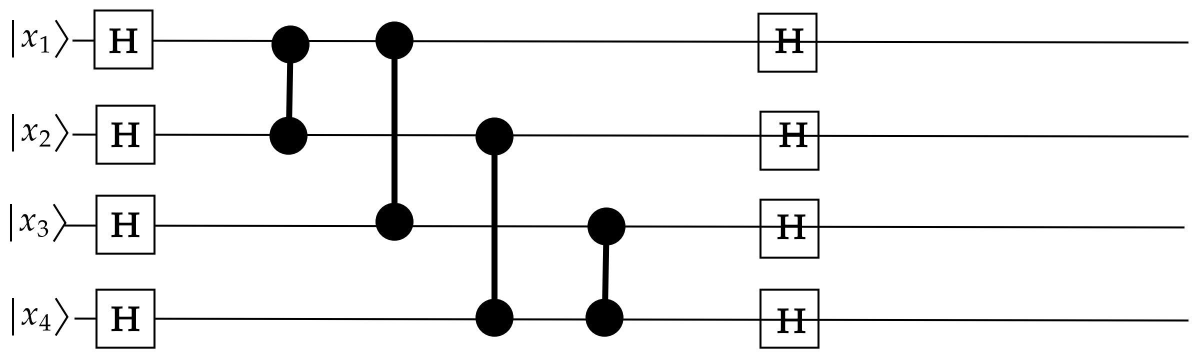
A word to the wise: The matrix for the fourth gate, which comes leftmost just above, is the tensor product $(\mathbf{I} \otimes \mathbf{I}) \times \mathbf{CZ}$. The matrices for the middle two gates, however, are technically not tensor products, because one identity comes "between the two arms" of the $\mathbf{CZ}$ gate. They are "morally" tensor products, though. The assigned exercise 4.11 makes a different case of this point. The rule about places with two particular 1s, however, applies in all cases. And the surviving -1 entries in the product at right mark four of the strings that gave exactly two 1s, the four corresponding to the edge set $E = \{(1,2),(1,3),(2,4),(3,4)\}$ of the graph.

If we apply our diagonal matrix to the all-1 unit vector, here

$$[1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,1,]\frac{1}{4} = |{+}{+}{+}{+}\rangle,$$

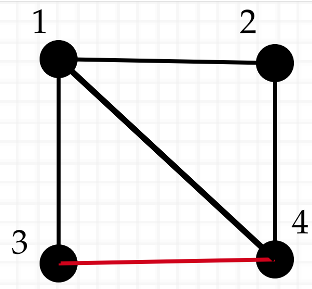
then we get the column vector of the diagonal entries at right (again, divided by 4 to normalize it). Does that column vector faithfully preserve all information about the given graph? A question to ponder...

A **graph-state circuit** conventionally includes an all-qubits Hadamard transform before and after it:

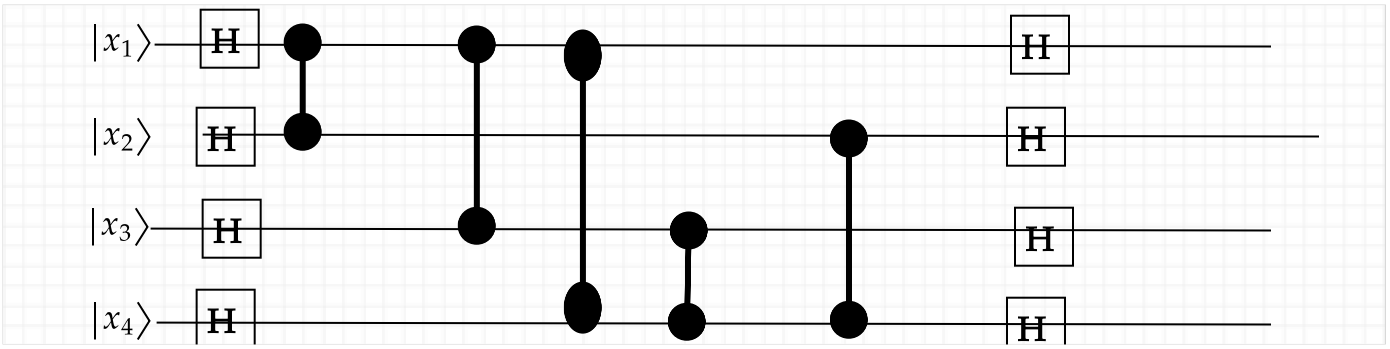


[Fall 2025: Lecture will pick up here. MathCha allows moving things around for illustration's sake. I think I've put them back the way they were to begin with. The relevant fact is that the CZ gates all commute with each other, which is to say that it doesn't matter what order you list the edges in.]

Lecture reviewed the logic of the -1 values in the 4-node graph state example above, and then added an edge between nodes 1 and 4 to make:



$$A = \begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 \end{bmatrix}$$



The edge added in the middle inserted another diagonal matrix in the middle below. That in turn updated the status of products of rows to product eight $+1$ and eight -1 values at far right:

$$diag\left(\begin{bmatrix}1\\1\\1\\-1\\1\\1\\1\\-1\\1\\1\\1\\-1\\1\\1\\1\\-1\end{bmatrix}\right)\cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\-1\\1\\-1\\1\\1\\1\\1\\1\\-1\\1\\-1\end{bmatrix}\right)\cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\1\\-1\\1\\-1\\1\\-1\end{bmatrix}\right)\cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\1\\1\\-1\\-1\end{bmatrix}\right)\cdot diag\left(\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\1\\-1\\-1\\-1\\-1\end{bmatrix}\right)= diag\left(\begin{bmatrix}1\\1\\1\\-1\\1\\-1\\1\\1\\1\\-1\\-1\\-1\\-1\\-1\\1\\-1\end{bmatrix}\right).$$

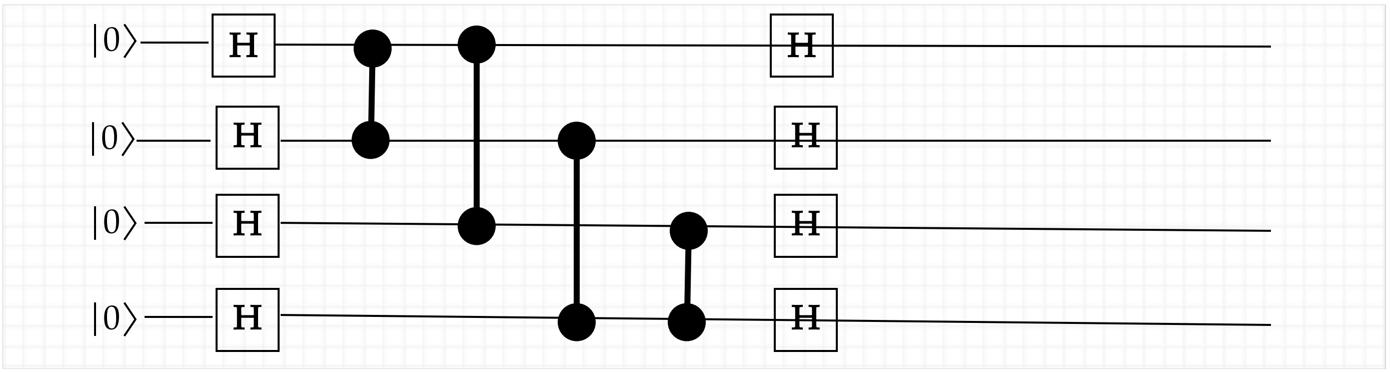
The final Hadamard transform effectively sums up the values in the rightmost column. Unlike above when it was twelve $1s$ to only four -1s, these cancel. The upshot (seen when doing this example on the Wybiral quantum circuit app at the end of lecture) is that the above circuit on input e_{0000} cannot give e_{0000} as output. We will pay attention to which graphs do this cancelling and which do not. Let's say more about those Hadamard transforms...

General Quantum Circuits and Computations

If there are n qubits, then the underlying matrices we get are $N \times N$ with $N = 2^n$. It is much harder to handle 2^n-sized stuff than n-sized stuff. Happily, we can always break the basic gates down to constant size---3 at most with the Toffoli gate in practice---and there are theorems that guarantee constant size gates working in general. One important case of using n single-qubit gates is the **Hadamard transform** $\mathbf{H} \otimes \mathbf{H} \otimes \cdots \otimes \mathbf{H}$ (n times), which can be abbreviated $\mathbf{H}^{\otimes n}$:

$$H^{\otimes 2} = \frac{1}{2}\begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix}, \quad H^{\otimes 3} = \frac{1}{2\sqrt{2}}\left[\begin{array}{cccc|cccc} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 & 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 & 1 & -1 & -1 & 1 \\ \hline 1 & 1 & 1 & 1 & -1 & -1 & -1 & -1 \\ 1 & -1 & 1 & -1 & -1 & 1 & -1 & 1 \\ 1 & 1 & -1 & -1 & -1 & -1 & 1 & 1 \\ 1 & -1 & -1 & 1 & -1 & 1 & 1 & -1 \end{array}\right]$$

We always have $H^{\otimes n}|0^n\rangle = |+\rangle^{\otimes n} = |+^n\rangle =$ the all-1 vector of length $N = 2^n$ divided by $\sqrt{N} = \sqrt{2^n} = 2^{n/2}$. Often this is the first step of a quantum circuit, for example:



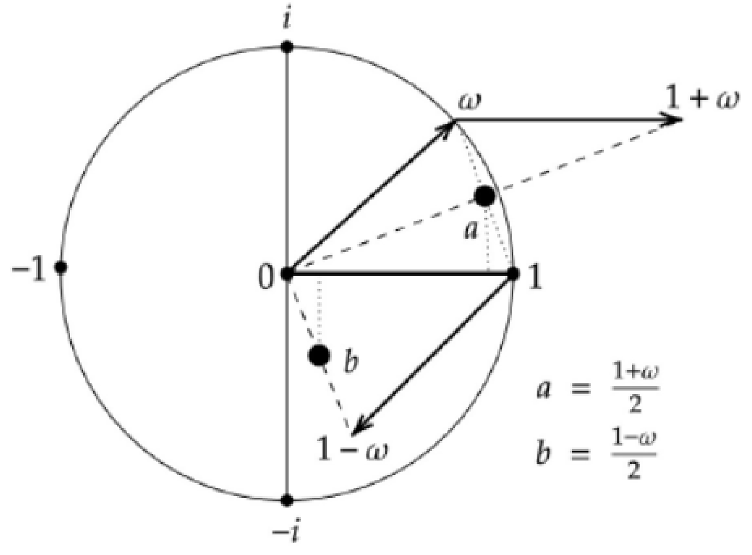
Putting the same Hadamard transform also after the edges of the graph G creates the graph state circuit C_G. One question we will soon address is whether C_G on all-$|0\rangle$ input can possibly produce all-$|0\rangle$ output. The "possibly" part here involves randomized results coming from *measurements*, which we will shortly define.

We will call an $N \times N$ matrix that arises from a single small gate---or a tensor product of small gates---a **succinct** matrix. Thus a **quantum computation** of **length** s is formally a composition of s succinct matrices applied to some input vector. The text draws allusion to a classical computation on a binary string x of length n, such as $x = 10100010$, say. The quantum circuit starts with input the basis state $|x\rangle = |10100010\rangle$. We could actually start with $|0^8\rangle$ but then **prepare** the state $|x\rangle$ by making the first column of the circuit be the tensor product

$$\mathbf{X} \otimes \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{I} \otimes \mathbf{X} \otimes \mathbf{I},$$

which has a NOT gate where x has a 1. This is why we often suppose ("without loss of generality") that the circuit starts with the all-zero basis vector. Most quantum algorithms begin with a Hadamard transform on all n qubits anyway, thus "really" starting with the equal and completely superposed separable state $|+\rangle^n$.

The $\mathbf{Z}$ and $\mathbf{CZ}$ gates are the heads of an important family of basic gates having to do with rotations of **phase**, which is a curious but definitely physical property. When a complex number $x + iy$ is rewritten in polar form as $re^{i\theta}$, the angle θ is the phase. The magnitude is r, so when $r = 1$ we have a unit complex number. Note that i itself is the same as $e^{i\pi/2}$ since $\frac{\pi}{2}$ means 90° phase. Then $i^2 = e^{i\pi} = -1$ and if we put $\omega = e^{i\pi/4}$ then $\omega^2 = i$. In Cartesian coordinates, $\omega = \frac{1+i}{\sqrt{2}}$. Here is some more geometry:



The vector $\mathbf{u} = [a, b]^T$ is a funky unit vector. To see that it is a unit vector, note that

$$||\mathbf{u}||^2 = \langle \mathbf{u}, \mathbf{u} \rangle = \mathbf{u}^* \cdot \mathbf{u} = a^*a + b^*b = \left(\frac{1+\overline{\omega}}{2}\right)\left(\frac{1+\omega}{2}\right) + \left(\frac{1-\overline{\omega}}{2}\right)\left(\frac{1-\omega}{2}\right).$$

In polar form, the complex conjugate of $e^{i\theta}$ is always $e^{-i\theta} = e^{i(2\pi-\theta)}$, so $\overline{\omega} = e^{i7\pi/4} = \omega^7$. In Cartesian coordinates,

$$\frac{1+\omega}{2} = \frac{1}{2}\left(1 + \frac{1+i}{\sqrt{2}}\right) = \frac{\sqrt{2}+1+i}{2\sqrt{2}} \quad \text{and} \quad \frac{1-\omega}{2} = \frac{1}{2}\left(1 - \frac{1+i}{\sqrt{2}}\right) = \frac{\sqrt{2}-1-i}{2\sqrt{2}}$$

So

$$\frac{1+\overline{\omega}}{2} = \frac{1}{2}\left(1 + \frac{1-i}{\sqrt{2}}\right) = \frac{\sqrt{2}+1-i}{2\sqrt{2}} \quad \text{and} \quad \frac{1-\overline{\omega}}{2} = \frac{1}{2}\left(1 - \frac{1-i}{\sqrt{2}}\right) = \frac{\sqrt{2}-1+i}{2\sqrt{2}}.$$

Then

$$\left(\frac{1+\overline{\omega}}{2}\right)\left(\frac{1+\omega}{2}\right) = \frac{1}{8}\left(\sqrt{2}+1+i\right)\left(\sqrt{2}+1-i\right) = \frac{1}{8}\left[\left(\sqrt{2}+1\right)^2+1\right] = \frac{1}{8}\left(2+1+2\sqrt{2}+1\right) = \frac{2+\sqrt{2}}{4}$$

and

$$\left(\frac{1-\overline{\omega}}{2}\right)\left(\frac{1-\omega}{2}\right) = \frac{1}{8}\left(\sqrt{2}-1-i\right)\left(\sqrt{2}-1+i\right) = \frac{1}{8}\left[\left(\sqrt{2}-1\right)^2+1\right] = \frac{1}{8}\left(2+1-2\sqrt{2}+1\right) = \frac{2-\sqrt{2}}{4}.$$

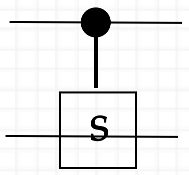
These squared values add to 1 as promised, so $\mathbf{u} = [a,b]^T$ is a unit vector. How do we get it? Here is the start of an infinite family of gates:

$$\mathbf{Z} = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}, \quad \mathbf{T} = \begin{bmatrix} 1 & 0 \\ 0 & \omega \end{bmatrix}, \quad \mathbf{T}_{\pi/8} = \begin{bmatrix} 1 & 0 \\ 0 & e^{i\pi/8} \end{bmatrix}.$$

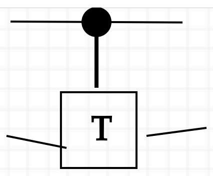
The controlled versions to go with $\mathbf{CZ}$ are $\mathbf{CS}$, $\mathbf{CT}$, etc. They, too, are symmetric---indeed, all of these gates are controlled phase shifts conditioned on the basis-state 1 of all of the (one or two) qubits involved. (Here I must note global inconsistency and confusion in notation, especially about rotations, which we will try to resolve when we cover the **Bloch Sphere** next week.)

Two other 2-qubit gates and their matrix and circuit representations are:

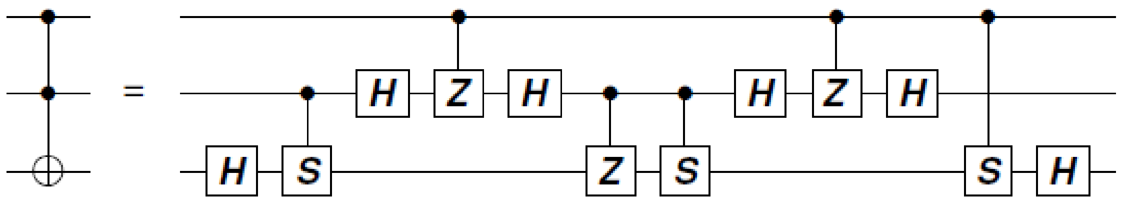
$$\mathbf{CS} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & i \end{bmatrix}$$



$$\mathbf{CT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & \omega \end{bmatrix}$$



Here is a famous circuit equation (from this 1996 paper) that uses $\mathbf{CS}$:



If we had to "go to the matrices" we'd be multiplying *twelve* 8×8 matrices together. Happily we can verify this for the eight standard basis vectors and then apply the principle of linearity to conclude that it works for any three-qubit quantum state given as input. When given standard-basis vectors, it is kosher to "reduce" controlled gates and then carry out further cancellations that result. But we can't assume that *non*-standard-basis states pass unchanged through controls. [Lecture did the logic: If the first qubit is 0, then the two "HZH" blocks go away and also the controlled S at far right. Because SZS=I, those gates also cancel regardless of the second qubit. Then the bottom two H gates cancel. The result is a no-op, like the Toffoli gate at left becomes. If the first qubit is 1 but the second is 0, then the first controlled S goes away, but the first HZH turns the second qubit *on,* so that ZS is active on line 3. The second HZH flips it the second qubit back to 0, so again we have a no-op on the first two qubits. But because the final controlled-S is active, we get ZSS on the bottom, which cancels---and then the two H gates cancel leaving a no-op again. But if both top qubits are 1, then the ZS in the middle disappears, but the other two S gates controlled from lines 2 and 1 are active. So we get HSSH = HZH on the bottom, which flips the third bit exactly as Toffoli then does, while the first two bits stay 1.]

Outputs and Measurements

There are various conventions about what it means for a family $[C_n]$ of quantum circuits to compute a function f on $\{0,1\}^*$, where f is an ensemble of functions f_n on $\{0,1\}^n$ and each C_n computes f_n. I like supposing that $f(x)$ is coded in $\{0,1\}^r$ where r depends only on n and giving C_n r-many output qubits separate from the n input qubits, plus some number m of ancilla qubits. (It is traditional, IMHO weirdly, to consider that the primordial input is always $\mathbf{0}^n$ and that for any other x, **NOT** gates are prepended onto the circuit for those lines i where $x_i = 1$.)

For *languages*, this means that the yes/no verdict comes on qubit $n+1$. Many references say to measure line 1 instead. (Using a swap gate between lines 1 and $n+1$ can show these conventions to be equivalent, but I prefer reserving lines 1 to n for *potential* use of the "copy-uncompute" trick, which is covered in section 6.3.) Even for languages, however, one evidently cannot get the most power if you need always to rig the circuit so that on any input $x \in \{0,1\}^n$, the output line always has a (standard-)basis value, i.e., is $\mathbf{0}$ with certainty or is $\mathbf{1}$ with certainty. Instead, one must **measure** it, whereupon the value $\mathbf{0}$ is given with some probability p, $\mathbf{1}$ with probability $1-p$.

The math of measurements (at least of the kind of *pure states* we get in completely-specified circuits) is simple. At the end we have a quantum state Ψ of $n+r+m$ qubits, counting the output and any ancilla lines. It "is" a vector $(v_1, v_2, \dots v_Q) \in \mathbb{C}^Q$ where $Q = 2^{n+r+m}$. Numbering $\{0,1\}^{n+r+m}$ in canonical order as $z_1, \dots, z_S$, an **all-qubits measurement** gives any z_j with probability $|v_j|^2$. If we focus on just the r output lines, then any $y \in \{0,1\}^r$ occurs with probability

$$\sum_{j:\, z_j \text{ agrees with } y \text{ on the } r \text{ output lines}} |v_j|^2.$$

When $r = 1$ and $y = \mathbf{0}$ the sum is over all binary strings z_j that have a 0 in the corresponding places. To simplify the notartion, let p_x denote the probability of measuring 1 on the output qubit line. The notion of uniformity is similar to that for ordinary Boolean circuits: it means that C_n can be written down in $n^{O(1)}$ (classical) time. We can finally define:

Definition: A language L belongs to BQP if there is a uniform family $[C_n]$ of polynomial-sized quantum circuits such that for all n and inputs $x \in \{0,1\}^n$,

$$\begin{aligned} x \in L &\implies p_x \geq 3/4; \\ x \notin L &\implies p_x \leq 1/4. \end{aligned}$$

[Lecture ended with demos of a one-qubit circuit with three gates in the order HTH and then the above two graph-state circuits on four qubits.]