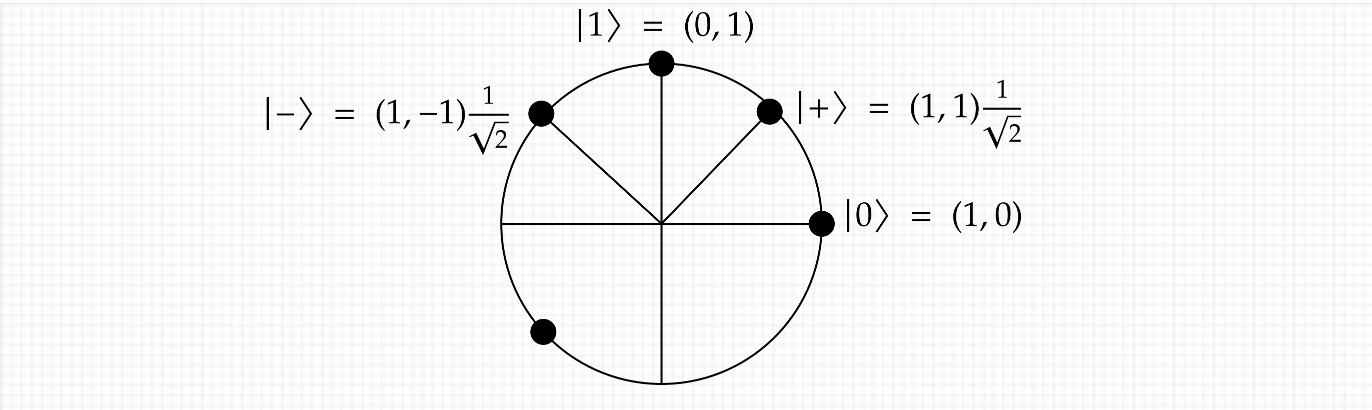
CSE491596 Lecture Wed. 12/07/22: Visualizing Quantum Circuits and Measurements

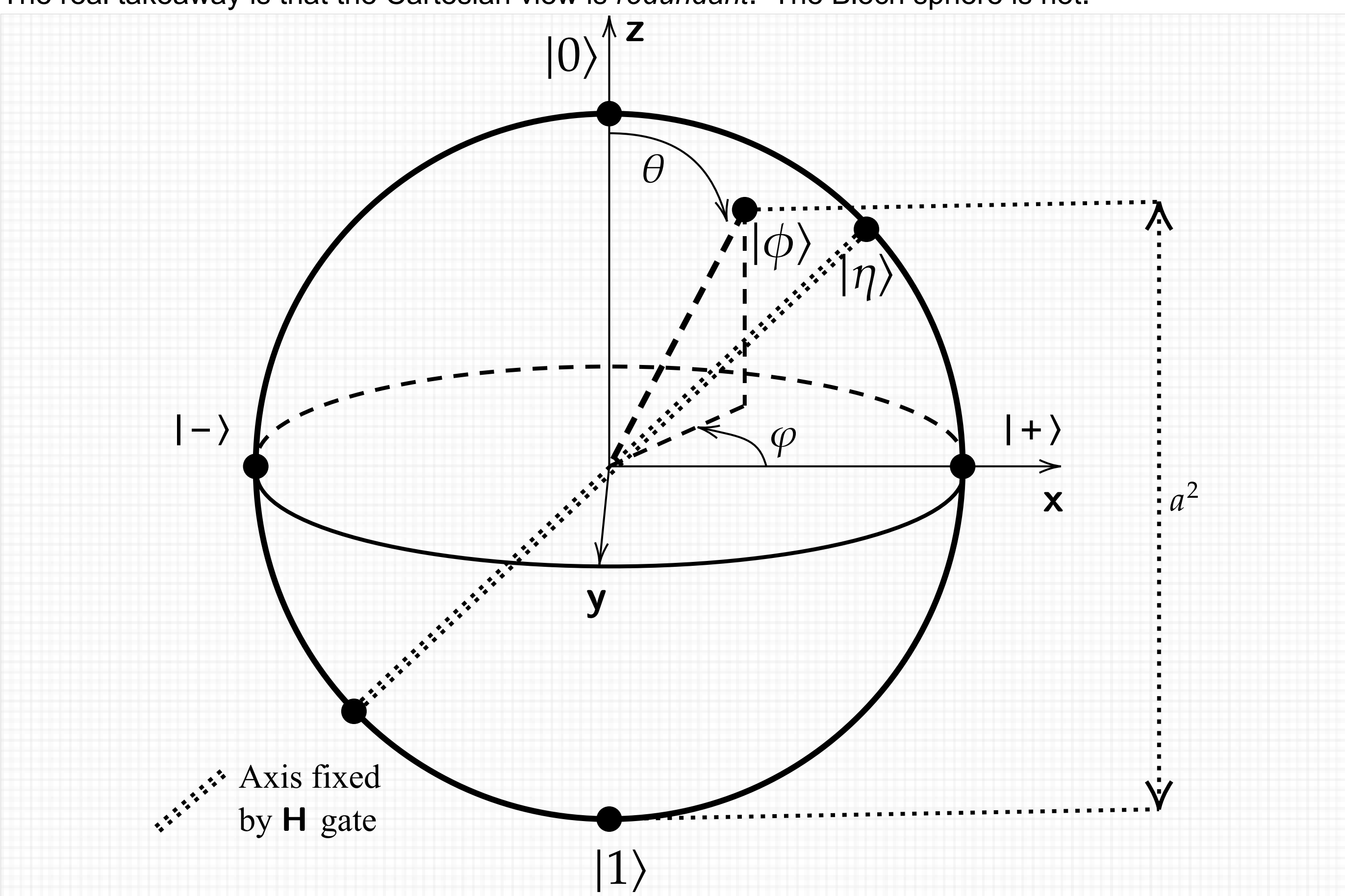
First some review of Monday's guest lecture, on the theme of "when is a typo not a typo?": A diagram in last Friday's posted course notes had a typo, where it said---

"Some algorithms, however, are IMHO easier to picture using the original planar diagram:"



The typo is $|-\rangle$ being graphed in the wrong place, a simple "sign error." But it is not really an error, because -1 is a unit complex number, so multiplying by it leaves **equivalent** quantum states. So $|-\rangle$, represented instead as $(-1,1)/\sqrt{2}$, could just as well be graphed "northwest."

The real takeaway is that the Cartesian view is *redundant*. The Bloch sphere is not.



Another rationale for the Bloch sphere is the following fact:

Theorem: Every 2×2 unitary matrix represents a rotation of the Bloch sphere around some axis.

The $\mathbf{NOT}$ gate turns the sphere upside down, so $|0\rangle$ goes to $|1\rangle$ and vice-versa. It does so by rotating 180 degrees around the x-axis. It leaves the states $|+\rangle$ and $|-\rangle$. Well, it actually maps $\frac{(1,-1)}{\sqrt{2}}$ to $\frac{(-1,1)}{\sqrt{2}}$ *but*---we just saw these are equivalent quantum states. So it fixes the x-axis, and that is why it has the alternative name $\mathbf{X}$. The Pauli $\mathbf{Y}$ gate fixes the y-axis, and $\mathbf{Z}$ fixes the z-axis. The Hadamard gate effects a 180-degree rotation around a diagonal axis, and that is how to visualize the way it switches $|0\rangle$ and $|+\rangle$, likewise $|1\rangle$ and $|-\rangle$.

Because DTIME$[t(n)]$ has Boolean circuits of size $\widetilde{O}(t(n))$, and because Toffoli gates are deterministic, we can state an immediate consequence:

Theorem: For fully time-constructible $t(n)$ between linear and exponential,

$$\mathsf{DTIME}[t(n)] \subseteq \mathsf{DQ}\left[\widetilde{O}(t(n))\right].$$

In particular, P $\subseteq$ DQP $\subseteq$ BQP, where BQP is to DQP as BPP is to P. (We define BQP formally after saying more about measurements.)

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 and is a presentation option.) 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 $y = 0$ the sum is over all binary strings z_j that have a 0 in the corresponding places. It is a postulate of quantum mechanics that we could do the measurement in such a way that the new state Ψ' stays "coherent" on qubit lines outside the r lines that were measured, but we will not care about this---we will be OK doing an all-qubits measurement (which "collapses" the system down to $|z_j\rangle$ for whatever basis state z_j is yielded) and then re-starting the whole circuit to do multiple trials, if necessary. What can make them necessary is the simple "unamplified" definition of BQP along lines of the definition given for BPP. 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.

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 $x \in \{0,1\}^n$,

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

With the help of ideas grouped under the term "**principle of deferred measurement**", the idea of amplifying the difference in probabilities by repeated trials and majority vote of the outcomes can be internalized within the circuits. This needs polynomially more ancilla qubits but allows doing only one measurement, which will then be guaranteed to give the correct answer with probability supremely close to 1 rather than probability $3/4$. However, it is (IMHO) more helpful to think instead of quantum circuits as objects that can be ***sampled***, and that a final classical post-processing routine gives the final answer as a function of the results of the samples. This is how Simon's algorithm, Shor's algorithm, and (general forms of) Grover's algorithm are usually conceived. The same approach of assembling a value $g(x)$ from multiple sample results can likewise be used for defining how functions g are computed.

With that said, the idea of computing a function $f(x) = y$ with y represented literally within a quantum (basis) state is often applied a different way. Given a circuit C computing y on lines $n+1, \dots, n+r$ that way---and using "copy-uncompute" to restore x on lines $1, \dots, n$---make C' by prepending $\mathbf{H}^{\otimes n}$ on the first n lines. Give C' $|0^n\rangle$ as the actual input. The resulting state is

$$s_f = \frac{1}{\sqrt{2^n}} \sum_{x \in \{0,1\}^n} |x\rangle |f(x)\rangle.$$

Although each individual term $|x\rangle|f(x)\rangle$ is separable---indeed, it is the basis state $\mathbf{e_x} \otimes \mathbf{e_y} = \mathbf{e_{xy}}$ where $y = f(x)$---the sum is usually majorly entangled. Our text calls this the **functional superposition** of f over the domain $\{0,1\}^n$. In **Shor's algorithm** for a product $M = pq$ of two primes, first a seed $a < M$ is chosen randomly from the $\rho = (p-1)(q-1)$ numbers that are not multiples of p or q. Then $f(x)$ is the function $a^x \bmod M$, where x is redundantly allowed to go as high as $Q-1$ with Q being a power of 2 between M^2 and $2M^2$. That makes enough room for the periodicity of the powering mod M to make enough waves for the QFT to do what Joseph Fourier knew it would 198 years ago: it transforms the waves' period, which divides ρ, into a peak. Repeated runs and measurements eventually give enough information about ρ to infer p and q.

Thus Shor's algorithm invokes *both* the "input x, output $f(x)$" view of what a quantum circuit does and the randomized sampling view. The latter is the external algorithm, and its input is not "x" but rather C, which in turn comes from the factoring problem instance M and the random seed a. In lieu of covering the full details in chapters 11 and 12, we can state:

Shor's Theorem: FACTORING is in BQP.

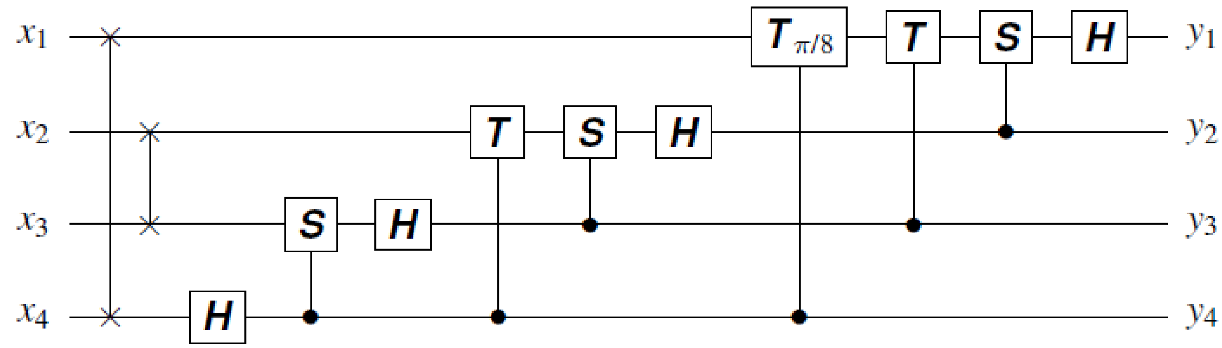
At present, I accept that s_f is feasible to build and the QFT is feasible to apply---at least with sufficient approximation for Shor's algorithm to work. However, I am chary of the account given under the Many Worlds Hypothesis. As told by David Deutsch and others, each Hadamard gate branches into two universes. If the n Hadamards stayed separate to make n pairs that might be reasonable, but building s_f seems to entail piggy-backing them to make 2^n universes, all harnessed together by the QFT.

Bigger Quantum Circuit Examples

We first 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 $\boldsymbol{H}^{\otimes n}$ is just n binary Hadamard gates laid out in parallel.
- The n-qubit **quantum Fourier transform** (QFT) can be built up out of $O(n^2)$ smaller gates---examples for $n = 3$ or 4 are a presentation option.

There is one thing that needs to be said about the QFT. The usual recursive way to build it via $O(n^2)$ unary and binary gates uses controlled rotations by exponentially tiny angles. This is already evident from the four-qubit illustration in the textbook (where the two gates on the left are :

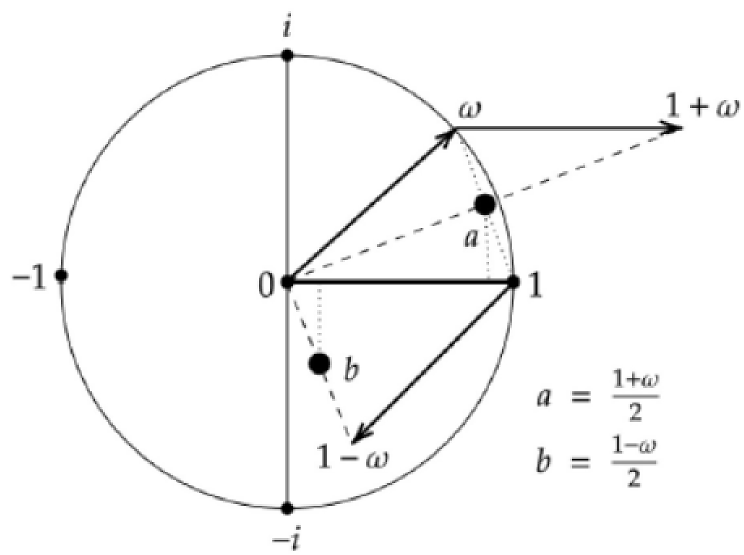


Here $T_{\pi/8} = \begin{bmatrix} 1 & 0 \\ 0 & \omega' \end{bmatrix}$ with $\omega' = e^{i\pi/8}$ not $\omega = e^{i\pi/4}$ as with the T-gate. So ω' has a phase angle one-sixteenth of a circle. For $n = 5$ the next bank uses $1/32$, then $1/64$, and soon the angles would be physically impossible so the gates could never be engineered. Those super-tiny angles are in the definition of the QFT itself. For any n, it takes $\omega_n = e^{2\pi i/N}$ where $N = 2^n$. With $n = 3$, the matrix together with its quantum coordinates is:

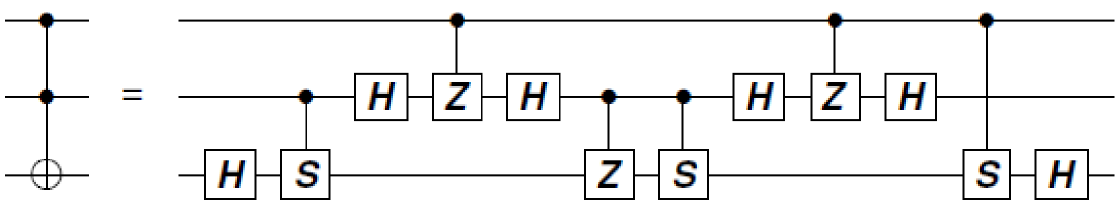
$$\left[\begin{array}{c|cccccccc} & 000 & 001 & 010 & 011 & 100 & 101 & 110 & 111 \\ \hline 000 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 001 & 1 & \omega & i & i\omega & -1 & -\omega & -i & -i\omega \\ 010 & 1 & i & -1 & -i & 1 & i & -1 & -i \\ 011 & 1 & i\omega & -i & \omega & -1 & -i\omega & i & -\omega \\ 100 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 101 & 1 & -\omega & i & -i\omega & -1 & \omega & -i & i\omega \\ 110 & 1 & -i & -1 & i & 1 & -i & -1 & i \\ 111 & 1 & -i\omega & -i & -\omega & -1 & i\omega & i & \omega \end{array}\right] = \left[\begin{array}{c|cccccccc} & 0 & 1 & 2 & 3 & 4 & 5 & 6 & 7 \\ \hline 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & \omega & \omega^2 & \omega^3 & -1 & \omega^5 & \omega^6 & \omega^7 \\ 2 & 1 & \omega^2 & \omega^4 & \omega^6 & 1 & \omega^2 & \omega^4 & \omega^6 \\ 3 & 1 & \omega^3 & \omega^6 & \omega & -1 & \omega^7 & \omega^2 & \omega^5 \\ 4 & 1 & -1 & 1 & -1 & 1 & -1 & 1 & -1 \\ 5 & 1 & \omega^5 & \omega^2 & \omega^7 & -1 & \omega & \omega^6 & \omega^3 \\ 6 & 1 & \omega^6 & \omega^4 & \omega^2 & 1 & \omega^6 & \omega^4 & \omega^2 \\ 7 & 1 & \omega^7 & \omega^6 & \omega^5 & -1 & \omega^3 & \omega^2 & \omega \end{array}\right]$$

$$\mathbf{QFT}[i,j] = \omega^{ij}$$

For $\mathbf{QFT}_N$ we raise ω_N with its tiny phase to exponentially many different powers. How can this possibly be feasible? Leonid Levin among others raised this objection. Here are several answers:



- Basic gates can fabricate quantum states having finer phases. This is already hinted by the diagram in the case of $\mathbf{HTH}$. Try composing $\mathbf{HTHT^*H}$ and $\mathbf{HTHT^*HTHT^*H}$. The *Solovay-Kitaev theorem* enables approximating operators with exponentially fine angles by polynomially many gates of phases that are multiples of ω (using **CNOT** to extend this to multiple-qubit operators).
- The Toffoli and Hadamard gates by themselves, which have phases only $+1$ and -1, can simulate the real parts and imaginary parts of quantum computations separately via binary code, in a way that allows re-creating all measurement probabilities. (This is undertaken in exercises 7.8--7.14 with a preview in the solved exercise 3.8.)
- The **CNOT** and Hadamard gates do not suffice for this, even when the so-called "phase gate" $\mathbf{S} = \mathbf{T}^2 = \begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$ is added. The Pauli $\mathbf{X}, \mathbf{Y}, \mathbf{Z}$ gates and also $\mathbf{CZ}$ can be built from these, but quantum circuits of these gates can be simulated in deterministic ("classical") polynomial time. However, $\mathbf{CS}$ suffices to build the Toffoli gate, per the diagram below (which is also a presentation option). So Hadamard + $\mathbf{CS}$ is a universal set using only quarter phases.
- The signature application of the QFT, which is *Shor's algorithm* showing that factoring belongs to BQP, may only require coarsed-grained approximations to $\mathbf{QFT}_N$.



For these reasons, $\mathbf{QFT}_N$ is considered feasible even though $N = 2^n$ is exponential. Not every $N \times N$ unitary matrix U is feasible---the Solovay-Kitaev theorem relies on U having a small exact formulation to begin with. But if we fix a finite **universal gate set** (such as $\mathbf{H} + \mathbf{T} + \mathbf{CNOT}$, $\mathbf{H} + \mathbf{Tof}$, or $\mathbf{H} + \mathbf{CS}$ above) and use only matrices that are compositions and tensor products of these gates, then we can use the simple gate-counting metric as the main complexity measure.

Reckoning and Visualizing Circuits and Measurements

There are basically three ways to "reckon" a quantum circuit computation:

1. Multiply the $Q \times Q$ matrices together---using *sparse-matrix techniques* as far as possible. If BQP $\neq$ P and you try this on a problem in the difference then the sparse-matrix techniques must blow up at some (early) point. The downside is that the exponential blowup is paid early; the upside is that once you pay it, the matrix multiplications don't get any worse, no matter how more complex the gates become. This is often called a "Schrödinger-style" simulation.
2. Any product of s-many $Q \times Q$ matrices can be written as a single big sum of s-fold products.

For instance, if A, B, C, D are four such matrices and u is a length-Q vector, then

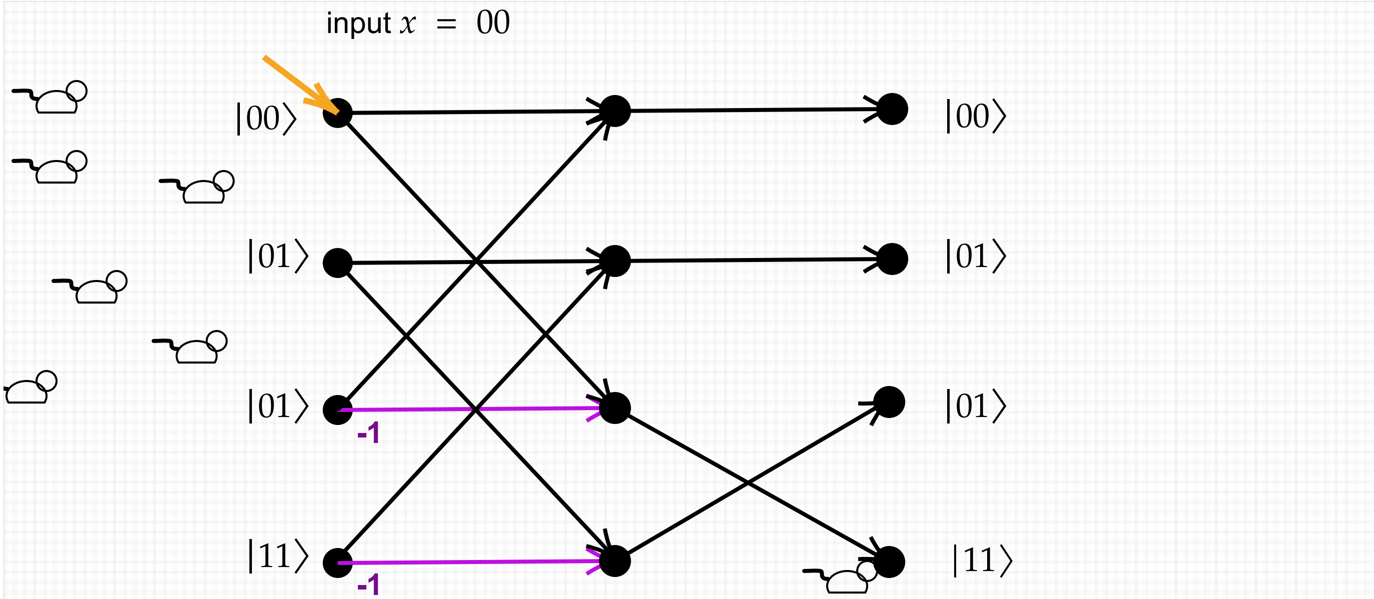
$$ABCDu[i] \;=\; \sum_{j,k,l,m=1}^{Q} A[i,j] \cdot B[j,k] \cdot C[k,l] \cdot D[l,m] \cdot u[m].$$

Every (*nonzero*) product of this form can be called a (*legal*) **path** through the system. [As hinted before, in a quantum circuit, u will be at left---on an input x, it will be the basis vector $\mathbf{e}_{x0^{r+m}} \;=\; |x0^{r+m}\rangle$ under the convention that 0s are used to initialize the output and ancilla lines---and D will be the first matrix from gate(s) in the circuit as you read left-to-right. Thus the output will come out of A, which is why it is best to visualize the path as coming in from the top of the column vector u, going out at some row m (where u_m is nonzero---for a standard basis vector, there is only one such m), then coming in at column m of D, choosing some row l to exit (where the entry $D[l,m]$ is nonzero), then coming in at column l of C, and so on until exiting at the designated row i of A. This is the discrete form of Richard Feynman's **sum-over-paths** formalism which he originally used to represent integrals over quantum fields (often with respect to infinite-dimensional Hilbert spaces). The upside is that each individual path has size $O(s)$ which is linear not exponential in the circuit size. The downside is that the number of nonzero terms in the sum can be far worse than Q and doubles each time a Hadamard gate (or other nondeterministic gate) is added to the circuit.

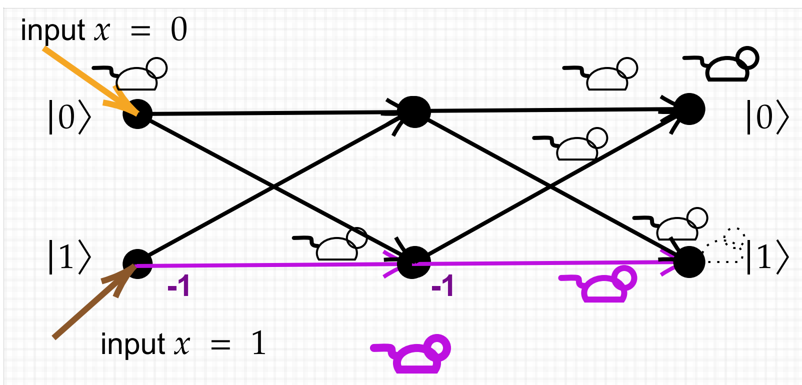
3. Find a way to formulate the matrix product so that the answer comes out of symbolic linear algebra---if possible!

For the textbook, I devised a way to combine the *downsides* of 1 and 2 by making an exponential-sized "maze diagram" up-front but evaluating it Feynman-style. Well, the book only uses it for $1 \;\leq\; Q \;\leq\; 3$ and I found that the brilliant Dorit Aharonov had the same idea. All the basic gate matrices have the property that all nonzero entries have the same magnitude---and when normalizing factors like $\frac{1}{\sqrt{2}}$ are collected and set aside, the Hadamard, **CNOT**, Toffoli, and Pauli gates (ignoring the global i factor in $\mathbf{Y}$) give just entries $+1$ or -1, which become the only possible values of any path. That makes it easier to sum the results of paths in a way that highlights the properties of **amplification** and **interference** in the "wave" view of what's going on. The index values $m, l, k, j, i, \;\ldots$ become "locations" in the wavefront as it flows for time s, and since it is discrete, the text pictures packs of...well...spectral lab mice running through the maze.

One nice thing is that you can read the mazes left-to-right, same as the circuits. Here is the $\mathbf{H} \;+\; \mathbf{CNOT}$ entangling circuit example:



No interference or amplification is involved here---the point is that if you enter at $|00\rangle$, then $|00\rangle$ and $|11\rangle$ are the only places you can come out---and they have equal weight. To see interference, you can string the "maze gadgets" for two Hadamard gates together:



$$\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} = \begin{array}{cc} |0\rangle & |1\rangle \\ \end{array} \begin{bmatrix} 2 & 0 \\ 0 & 2 \end{bmatrix}$$

In linear-algebra terms, all that happened at lower right was $1 \cdot 1 \ + \ -1 \cdot 1$ giving 0. But the wave interference being described that way is a real physical phenomenon. Even more, according to Deutsch the two serial Hadamard gates branch into 4 universes, each with its own "Phil the mouse" (which can be a photon after going through a beam-splitter). One of those universes has "Anti-Phil", who attacks a "Phil" that tries to occupy the same location (coming from a different universe) and they fight to mutual annihilation.