Chapter 15
Quantum information and quantum computing: the new kids on the block

Information theory is the new kid on the block in the physics community. Physicists these days study black holes using ideas first conjured by communications engineers studying information theory, trying to figure out how to send messages most efficiently across telephone lines. Complexity theory, born in computer science, now is contributing to work in quantum gravity. The goal of this article is to provide some background for understanding these developments.

Two problems

Physicists as of this writing (2023) are trying to solve two outstanding problems. (There are others, but two especially stand out.) First, what is the physical structure of space and time? Second, how can we reconcile general relativity and quantum mechanics, the two pillars of physics? It turns out those two problems are closely related.

Physical theory historically assumes that events occur on a pre-existing background of space and time. With Newton's laws we can calculate the orbits of planets around stars as if they were projected on a coordinate system of meter sticks and clocks. Einstein's relativity theory says that meter sticks stretch or shrink depending on the concentration of mass in their vicinity, and clocks slow down in a gravitational field. But the theory still assumes that clocks and meter sticks or comparable measuring tools exist in the structure of the universe. Only recently have physicists begun to tackle what the clocks and meter sticks are really made of, i.e. what is the underlying fabric of the "emptiness" out there between the galaxies.

The second problem, reconciling inconsistencies between quantum mechanics and general relativity, is more subtle. Quantum field theory (QFT is the most accurate physical theory we have. It describes processes at the very smallest scales: why quarks collect in protons and neutrons to form atomic nuclei, why electrons are attracted to nuclei to form atoms, why atoms absorb and emit light at particular wavelengths, etc. The mathematical equations of QFT agree with experimental measurement out to one part in a thousand million million, the limits of our current capacity to calculate and measure such things. General relativity (GR) is comparable in the accuracy of its predictions, precise confirmations limited only by the fact that GR deals with the very largest and most extreme structures in the universe – neutron stars, black holes, galaxies, and the universe itself – where measurements get messier just because of the enormous scale. Over one hundred years of observations, general relativity has passed all tests with flying colors: masses warp spacetime and bend the path of light; spinning masses drag spacetime around with them; black holes exist, as predicted; gravitational waves exist, as predicted. The problem is that there's no mathematical theory (except maybe string theory under special conditions) that includes both QFT and GR in its framework. We know there are circumstances where both theories should apply. For example, particles are produced at the event horizon of a black hole (the Hawking radiation). QFT can describe the particle production. GR can describe the black hole and its horizon. But no single theory yet exists that encompasses both in a single mathematical framework.

Enter quantum computation and information theory. Recent collaboration between computer scientists and the physics community has generated a lot of excitement, chipping away at these problems. That will be the purpose of the rest of this paper, to report new ideas relating the science of information to our traditional understanding of physics. First some background in what the information stuff is all about.

Information

Information was given a precise definition by communications researchers including Claude Shannon at ATT Bell Labs and Charles Bennett at IBM Research. At its essence, information is ones and zeros, yes vs. no, heads vs. tails. Information is the response to a yes-no question, and it is most easily contained in bits, either 1 or 0. Is direct sunlight coming through your window right now? Yes or no? If yes, label that a 1. That's the relevant *bit* of information. Is there an electron in register AF10H24B of your computer memory? Yes or no? If no, label that a zero.

All information can be encoded in 1's and 0's. It's as if nature plays an ongoing game of twenty questions. Is there a hydrogen atom at this particular location x, y, z at this particular time t? Yes or no. Is it moving at 10 m/sec? Is its spin up? Of course, we simplify our description of nature by consolidating information into standard measurements: what are the measured values of position, momentum and spin of the electron. That saves a whole lot of yes-no entries into our data books. But in principle, we could describe the world in ones and zeros for what's happening at every location in space and time, including yes-no questions for all possible spacetime events.

With this definition of information, it is convenient to encode messages as strings of ones and zeros – bit strings. Computers encode the alphabet in strings of eight bits. For example, to say "hi" send 01101000 01101001. (01101000 is the bit string for "h," and 01101001 is the bit string for "i.") If you want to be more enthusiastic, send "Hi!", 01001000 01101001 00100001. (01001000 is the bit string for "H," etc.) Even better, this convention allows you to process the message using standard mathematical operators (from the realm of linear algebra). For example, if you wanted to convert all the lower case letters "h" in a text to uppercase "H" you could scan the text for "h" (itself a bitwise operation) then carry out a matrix operation on the bit string to flip the third bit in "h" from 1 to 0. That particular matrix operation is kind of messy (involving an 8×8 matrix) so we'll look at a simpler operation in a minute. (If you are not familiar with linear algebra, suffice to say it's a mathematical system that allows us to convert one set of numbers (a vector) into another set of numbers using mathematical operators (matrices) governed by certain rules. Linear algebra is a hi-falutin' extension of standard arithmetic that allows us to model changes in complex systems, e.g. how does the path of an electron (a vector) change as it moves through an electric field (another vector).)

This bit-wise idea of information caught the attention of physicists toward the end of the last century. What is physics anyway? We're trying to understand nature, what the world is made of and how it works. That is, we're trying to extract information about nature. And if information is, indeed, 1's and 0's, then we should be able to understand it as such. One thing led to another (that's the rest of the story in this paper) and pretty quickly physicists starting calculating in ones and zeros. Suppose a photon, for example, flips the spin of an electron from spin down to spin up. Here's what that looks like in linear algebra bit notation.

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

where

$$down \equiv \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \ \ up \equiv \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

The matrix $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ represents a photon acting on the electron to flip the spin. More generally, matrices like $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ transform – stretch or shrink or rotate – vectors, represented e.g. by the original vector $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ and the flipped vector $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ in the equation above.

We model physical phenomena in mathematical equations. Math is a convenient (read that "indispensable") tool to represent nature. It not only describes what's going on, but its logical rules allow us to predict things we haven't yet seen. For some deep reason, nature herself follows those same rules. (Or perhaps nature <u>is</u> mathematics, and we're just discovering those maths.) These particular mathematical operations come from standard linear algebra. Sal Khan's video, vector transformation, reviews the math behind matrix multiplication if you'd like further explanation. Here's the link to Khan's Linear Algebra. Anyway, we'll explain the math as we go along.

Once they caught the bit bug, (or the bug bit them) physicists found more and more nutrition in information theory. One of the foundations of information theory that attracted their attention is the conservation law for information: information is strictly conserved. No information is ever lost. It might become inaccessible (for example if your computer hard drive fails). But it's never lost. Nature keeps strict accounting, and every last bit of information is recorded in nature's books. Forever.

Here physicists encounter familiar territory. Physics is built on the great conservation laws – conservation of energy, conservation of momentum, conservation of angular momentum, conservation of electric charge. (There are others as well, and all derive from a deeper principle, locality.) From the fresh perspective of information theory, we can re-interpret those classical conservation laws in terms of information, maybe even make some new headway. Conservation of energy implies conservation of information about the state of a physical system, say a collection of atoms, over time. Conservation of momentum implies conservation of information about the position of an object over time.

We find conservation of information most impressively in the quantum unitarity . Unitarity is a fancy term referring to probabilities. Flip a fair coin and there's 50% chance you'll get heads, 50% probability tails. But it is certain, 100% probability, that you'll get one or the other. That's unitarity. Probabilities all have to add up to one, 100%. Given that any particular outcome out of a number of possible outcomes *can* happen, it is certain that one of those outcomes *will* happen. Roll a six-sided die and you know with certainty it will come up 1 or 2 or 3 or 4 or 5 or 6. Plant 100 tomato seeds and you know with certainty that none will sprout or 1 will sprout or 2 or 3 or 4 or … or all 100. From the information perspective, that is conservation of info. Nature doesn't just throw away probabilities. It's not possible *not* to have tails as a possible outcome when you flip a fair coin. You don't lose information about the state of the coin – it really does have a tails – when you toss the coin. And nature doesn't just add possible outcomes out of nowhere. There's no third possibility suddenly appears when you toss the coin. We'll see more of this when we get to quantum computation, shortly. And, as we'll see, the notion of unitarity gets kind of contentious when we start to talk about black holes.

Classical circuits

Information can be stored. There are libraries filled with information and, of course, computer hard drives. It can also be processed. We can flip bits or extend bit strings or shrink them. One of the great insights in computer science was the realization that in order to process information we use information itself as the processor. We use one bit of information to tell us what to do with another bit – flip it or send it to memory or just pass it along a circuit wire. As Alan Turing showed, information not only is the grist for computation but it also provides the instructions for the milling.

Here's a simple "full adder" circuit, for example. See Figure 2 for an explanation of the logic gates in this circuit (the vertical lines with open and solid circles at their ends).

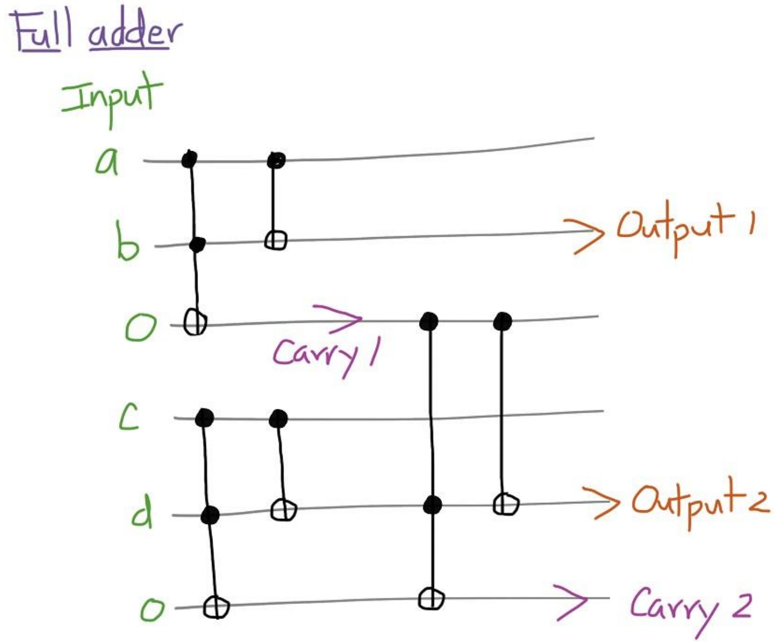


Figure 1. Full adder circuit. This circuit adds two-bit binary numbers. The result is a bit string of up to three bits. Wires 3 and 6 always start with an input bit of 0. ab and cd carry the input bit pairs to be added. To add one plus one, enter ab + cd = 01 + 01. The result is 010 (one plus one equals two, in binary notation). A couple other examples: 01 + 10 = 011 (one plus two equals three); 01 + 11 = 100 (one plus three equals four). The bit string output (far right of the diagram) reads from bottom to top. See example in the next Figure. Khan Academy binary arithmetic has more examples. Binary arithmetic is the basis of all computer operations and computer memory. The two gates in this circuit are the CNOT gate and the Toffoli gate (named after Tommaso Toffoli). Other gate combinations also could be used for the adder. See the gate figure on the next page for explanation how these two gates work.

Horizontal lines in the circuit diagram represent wires, which carry bits through time. They may be actual wires in a computer or radio waves carrying AM or FM signals or any of many other physical transmitters. Time runs left to right. We can parse the ticks of the clock as uniform intervals along the horizontal axis or in terms of the bit transformations, one after another, carried out in sequence by the logic gates along the circuit. Boxes represent single bit gates acting only on the bit in that particular wire. An example is the NOT gate; it flips the value of the input. If input is 0, output is 1; if input is 1, output is 0. Vertical connecting wires represent two- or three- bit gates. The action of these gates on the target bit depends on the value of the bit(s) in the input wire(s). For example, a CNOT gate (controlled NOT, vertical line from solid dot at upper wire to open dot on lower wire in the figure below) flips the target bit (wire through open dot) if the input (wire through solid dot) is a 1. If the input is 0 CNOT leaves the target at its original value. See Figure 2 and the truth values in the Appendix, Table 1.

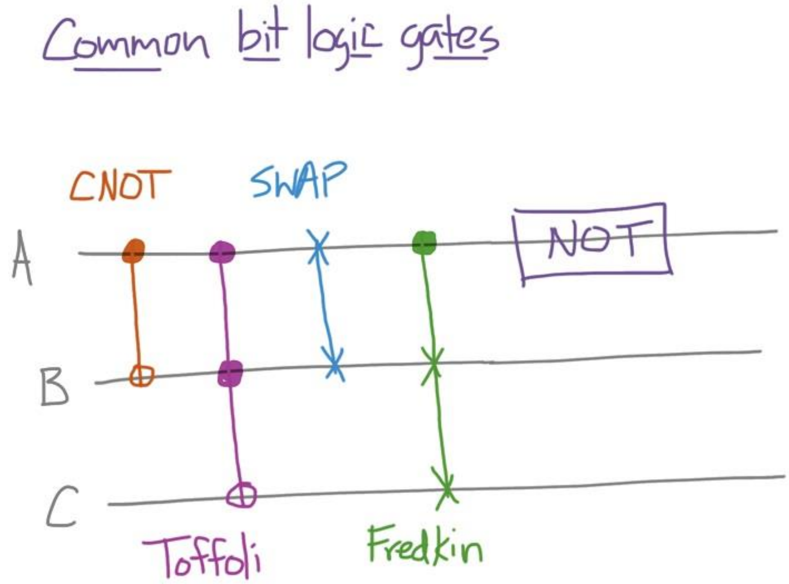


Figure 2. Some binary gates. Shown are gates useful to our purposes. Most classical circuits also include AND, NAND, and OR gates, not shown. NOT is a single bit gate. It flips the bit in its wire from 0 to 1 or 1 to 0. CNOT and SWAP are two-bit gates. SWAP exchanges bits between two wires; CNOT flips the target bit (open circle, B in this figure) if the input (A in the figure) equals 1, otherwise leaves the target bit unchanged if A is 0. Fredkin swaps B and C if A is 1, does nothing if A is 0. Toffoli flips C if both A and B are 1's, otherwise does nothing.

The full adder adds up to seven (111 in binary notation). If you link a series of full adders, with the carry value as input **a** to the next adder in the series, you can calculate any sum. And if you can add, then you can also subtract and multiply and divide. The full adder enables all basic whole-number arithmetic. That's most of digital computation right there.

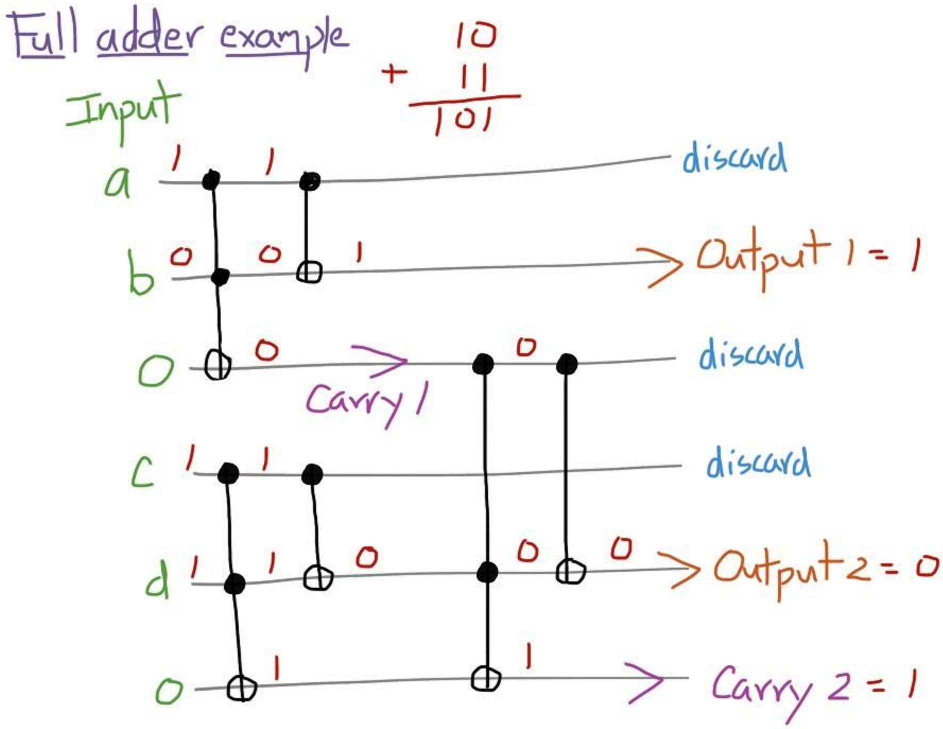


Figure 3. Example of a calculation with the full adder. $2 + 3 = 5 \rightarrow 010 + 011 = 101$. Two (10 in binary notation) plus three (binary 11) equals five. Red values track bits along the wires as the various gates operate. Output in standard pencil-and-paper notation reads from bottom to top on the far right of the circuit output, 101. Carry 2 (a 1 in this example) becomes input **a** for the next adder in a larger circuit. Try out other sums – it's kind of fun to track bits through these circuits and watch the gates do their magic.

It turns out you only need a handful of logic gates to build a circuit for any possible bit-wise computation. XOR (exclusive OR) and AND provide a universal set; the right combinations of just those two will build your computer. Just NAND by itself (not-AND) is universal. Apple could use circuits built from NAND to make the iPhone (but there are more efficient gate designs). Vice versa, a circuit with NAND gates can be decomposed into a circuit with combinations of other gates. See Table 1 in the Appendix for a more complete set of bit logic gates.

In circuit diagrams physicists saw new models for physics. Not only can you build circuits to run your calculations but maybe you can model the physics itself *in* the circuits. Two electrons exchange a photon that flips their spins – maybe that's a wire (electron moving in space and time) and a SWAP gate (photon exchanging spins). Further progress along those lines requires that we dive into quantum computation.

Onward, then, to quantum circuits!

<u>Quantum information and quantum states</u>

So far we've been thinking classically. Bits are discrete, either 0 or 1. Quantum mechanics says a "bit" of information can be a 0 or a 1 or a little bit of both at the same time, a "qubit." Moreover, qubits can become entangled with other qubits in vast networks. Information can be distributed throughout a collection of entangled qubits.

We need new symbols to contain these ideas. Up to now we've used 1 and 0 to represent bits. For qubits we'll use $|0\rangle$ and $|1\rangle$, the quantum symbols for state vectors. Quantum mechanics models the world with state vectors in a vector space, and linear algebra is the language to describe it. We'll use electron spin as an example, but the ideas (state vectors and state space) apply to any property. There's a state space for every parameter, be it spin or position or momentum or color charge or any other characteristic (observable) of particles and fields. The full state of the electron, including its spin, position, momentum, etc. sits in a larger (Hilbert) space comprising the particular observables as subspaces.

Let's spend a minute to clarify those concepts. They're not everyday vocabulary! (See Susskind and Hrabovski, 2013, for a more thorough discussion of fields and state space.) A field is something that takes on a value at every point in space. Temperature is a field, technically a *scalar* field. You get a number (a.k.a. *scalar* value) when you measure the temperature at any particular location in a room. The flow of water in a river is a *vector* field. The current has both a speed and a direction, 3 m/sec due west in the middle of the river, 1 m/sec east in an eddy along the bank. Electric and magnetic fields are good examples of vector fields in physics. The needle on a compass (a vector) points north.

As the name "quantum field theory" implies, physicists describe natural phenomena, especially particles and their interactions, in terms of fields. In this way of thinking, an electron is an excitation in the underlying electron field. Think of electrons like droplets of ocean spray excited by wind and currents out in the swells of the ocean "field." When we talk about particles in these terms, we're talking about field quanta, i.e. packets of field.

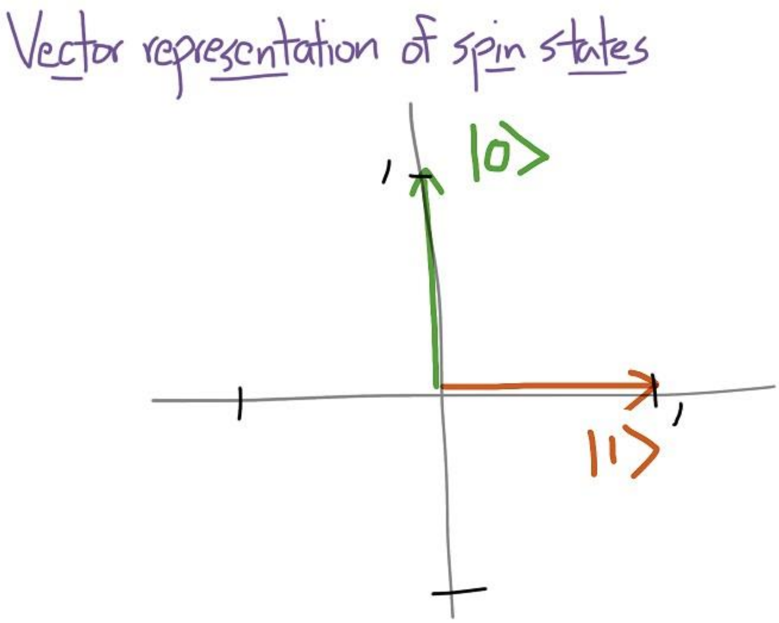
State space is like a catalog of all possible states of a system. The state space of spin for a single electron includes $|0\rangle$ and $|1\rangle$. When we measure its spin along the up-down direction, we find that the electron is either spin up or spin down. That's the result of the measurement. But there's much more. The state space of electron spin also includes $|left\rangle$ and $|right\rangle$ and also $|forward\rangle$ and $|backward\rangle$ and all other directions in between.

Using this notation, we can represent the state space for a system of two electrons is $|00\rangle$ (both up), $|01\rangle$ (first electron up, second electron down), $|10\rangle$ (first electron down, second up), and $|11\rangle$ (both down). That is, there are four possible outcomes when we measure the spin state of two independent electrons in the up-down direction. The entire state space for two electrons, of course, includes combinations of the two electrons pointing independently every which way. As you can see, things get real complicated, the state space expands rapidly, as you add more and more electrons to the system.

For convenience, we can record the state space as a matrix (a list of possible states in matrix form). For example, here's the spin state space for a system of three independent electrons. Rows are the possible outcomes for measurement e.g. in the up-down direction, eight altogether. Notice that we're interested in the state of the *system* composed of the three electrons.

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

This representation provides a convenient method to help visualize what's going on. Since we can record states as vectors (collections of numbers) we can also *draw* those vectors using the standard mathematical representation of vectors as arrows. Back to the spin state space for one electron, for illustration:



<u>Figure</u> 4. Vector representation of spins up $|0\rangle$ and down $|1\rangle$ on the coordinates representing the vector space of spin up vs. down states. Note a couple things. First, the coordinate axes are *not* the x, y axes we're used to in regular geometry. The axes here represent the direction of the electron's spin relative to some other outside Cartesian (x, y, z) axes. For example, the vectors in the figure represent spin direction relative to the spatial (Cartesian) z-axis. Second, note also that the vector labels are arbitrary. By convention, we've chosen $|0\rangle$ as the spin up vector and $|1\rangle$ as spin down. Finally, note that in this representation the vectors are *orthogonal* (i.e. at right angles) and not pointing opposite directions as we would expect in the regular world of ups and downs. Mathematical orthogonality means that a state is definitely one thing and not the other. In this case, when we measure the electron along the z-axis the results show it is definitely up or definitely down. This orthogonality in the vector representation follows the mathematical rules of linear algebra assuring that when we measure the spin of an electron it is either up or down – even though the state of the electron, before any measurement, may be a mix of both!

We mentioned that measurements only read out 1's or 0's for the state of electron spin. Unobserved, electrons' spin axes point every which direction. Here's the full quantum expression for the state vector representing electron spin.

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$$

The brackets tell you we're dealing with vectors. $|\psi\rangle$ (Greek letter psi) is the conventional symbol for a state vector. α (Greek letter alpha) is the amplitude (component) of the electron's spin along the $|0\rangle$ direction, i.e. how much the spin axis is tilted upward. β is the amplitude of spin in the $|1\rangle$ direction, i.e. how much the spin axis is tilted downward. α and β, the amplitudes, are complex numbers in the formalism of quantum mechanics. For our purposes, you can think of them just a numbers, e.g. $\alpha = 0.8$ and $\beta = 0.6$, the proportions of up-ness and down-ness in the electron's spin.

One of the requirements of the quantum conventions is that $\alpha^2 + \beta^2 = 1$. We've seen this before. It's unitarity. We are requiring that the spin of the electron is pointing in *some* direction and that the magnitude of the spin always equals one. It's an inherent, invariant property of the electron, part of what makes an electron an electron.

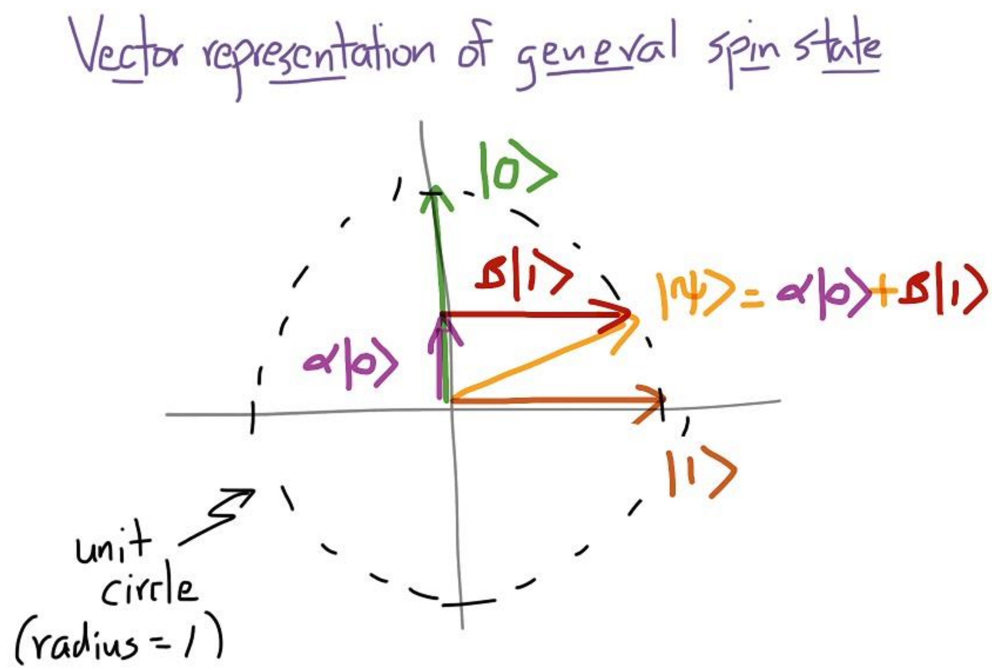


Figure 5. Vector representation of a general state vector, $|\psi\rangle$, showing its component vectors $\alpha|0\rangle$ and $\beta|1\rangle$. In this case, $|\psi\rangle$ is built from a proportion α of $|0\rangle$ and proportion β of $|1\rangle$. Note that $|\psi\rangle$, like $|0\rangle$ and $|1\rangle$, is one unit in length. This is the requirement of unitarity, assuring that calculations always give probability = 1 when you add all possible vector components for a particular state.

There are some subtleties here. Electrons aren't really like little spinning tops, but the real physical property of spin can be conveniently described in those terms. And there's nothing magical about the Greek letters. They're just handy when you run out of the good ol' Latin alphabet. a's and b's and p's and q's have already been taken for other purposes.

<u>Quantum circuits</u>

Back now to circuitry. Turns out we can understand a lot of quantum mechanics based on circuits, and with circuits we can calculate.

Wires in a quantum circuit are qubits instead of ones and zeros. We represent them in vector notation, e.g. $|\psi\rangle$. The gates in a quantum circuit are vector operators, unitary matrices in the mathematical formalism. We've already seen an example in our classical circuit.

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

The matrix $\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$ flips the spin of an electron from down to up. That's a perfectly good quantum operation, by the way, assuming the electron is in a pure down state to start with, spin axis pointed straight along the down axis. Unitary means what you suspect. Unitary operators (matrices) preserve the probabilities, so that circuits (and the world) never produce more states than they started with.

One of the joys (or headaches) of quantum circuits is that any unitary matrix qualifies as an operator and, therefore, a gate. Some are far more useful than others, though, and just as in classical circuits you only need a few kinds of gates to build any conceivable quantum circuit. See Table 2 in the Appendix for a list of qubit logic gates.

Among the most useful of the quantum operators is the Hadamard gate. Hadamard takes a pure spin state, up or down, and transforms it into a mixed state. For example

$$|0\rangle \rightarrow^{H} \rightarrow \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$$

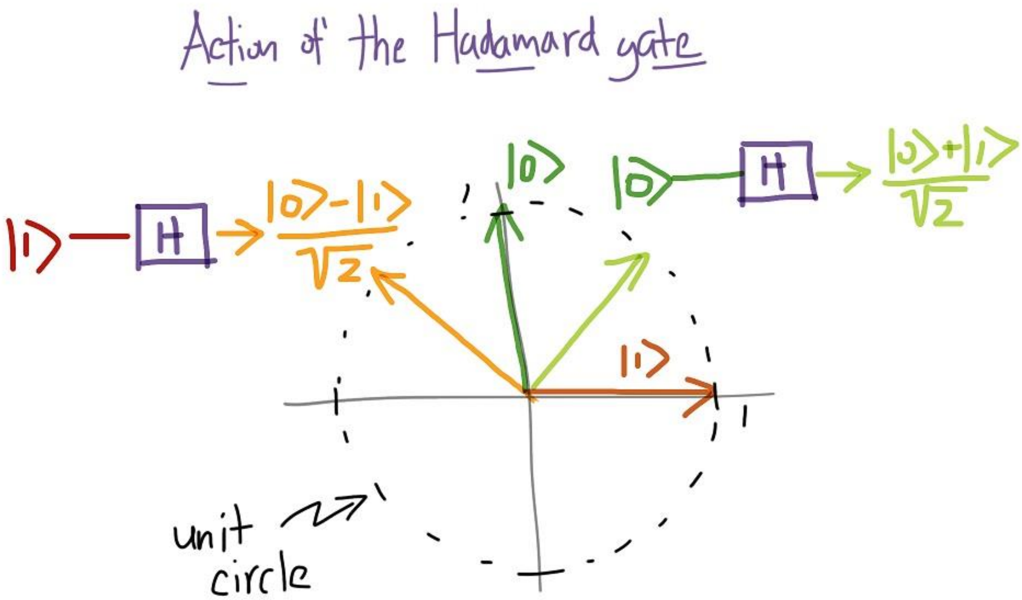


Figure 6. Action of the Hadamard gate on qubits $|0\rangle$ and $|1\rangle$. Note that Hadamard results in a mixed state; the original qubit is transformed into a mix of $|0\rangle$ and $|1\rangle$. In terms of the general state $|\psi\rangle$, Hadamard resets the values of α to $\frac{1}{\sqrt{2}}$ and $\beta \rightarrow \frac{1}{\sqrt{2}}$ or $\frac{-1}{\sqrt{2}}$. Note that, as required by unitarity, $\alpha^2 + \beta^2$ still $= 1$ after the transformation.

Now we can start to do some magic or, rather, replicate some of the magic that Nature performs. Hadamard takes the first step in the maths of entanglement. Whenever two particles bump into each other their state vectors become entangled. As a result, even after the two particles are separated in space and time, you can get information about the state of one particle by measuring the state of the other. Out in the real world particles are always bumping into each other. (More properly, the fields that carry particle properties are always interacting.) And 'way back in the beginning at the Big Bang origin of the universe, all the fields were jam-packed squished and all interacting, so the whole shebang is entangled. Since all that information is correlated through the entanglement, extracting information *here* can give us information about conditions out *there* across the universe and maybe even inside black holes.

We have all the circuit tools we need to generate entangled pairs of qubits. It's easy to represent entanglement with state vectors. For example, here's an entangled spin state for a pair of electrons.

$$|\psi\rangle = \frac{1}{\sqrt{2}}(|00\rangle + |11\rangle)$$

We don't know the spin of either electron until we measure one of them. All we know is the overall state $|\psi\rangle$ represented in the formula above; we entangled the electrons in that particular state. There's a 50% chance both electrons are spin up, 50% chance they're both spin down. But if we measure the first electron (labeled green) and find it's spin is up, then we also know, with 100% certainty, that the second electron (labeled by magenta) has spin up. And if we measure the first electron and find its spin is down, then we also find the second electron has spin down. That's entanglement. We can determine the state of one electron by measuring the other, even if they are separated from each other across the room or across the universe.

To build such an entangled state, all you need is two qubits for input then a Hadamard gate followed by a CNOT. Presto! You've got an entangled pair, a so-called Bell pair (named after John Bell, who studied the marvelous properties of such creatures and, with them, proved that quantum mechanics is not compatible with standard classical logic).

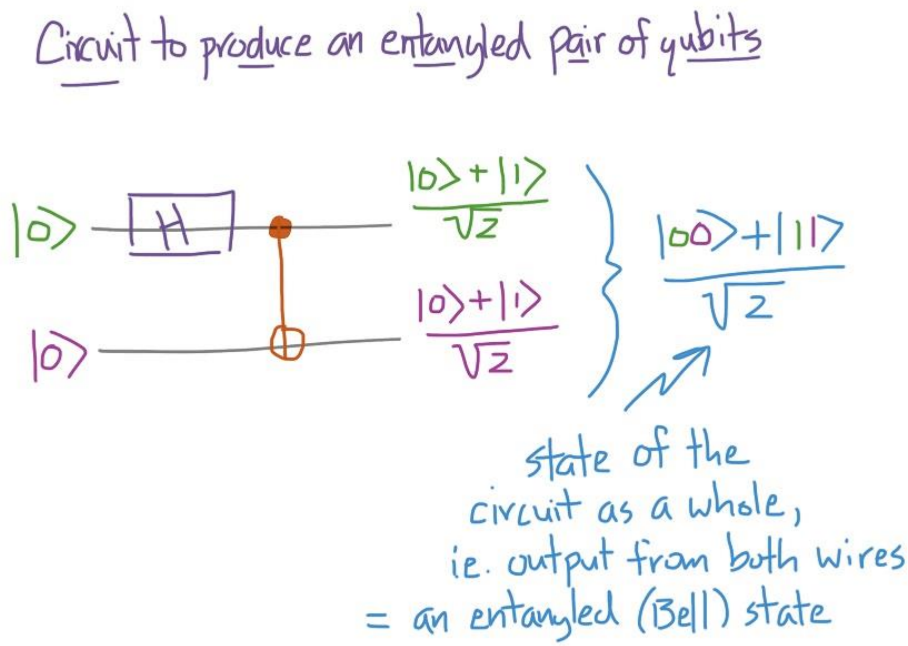
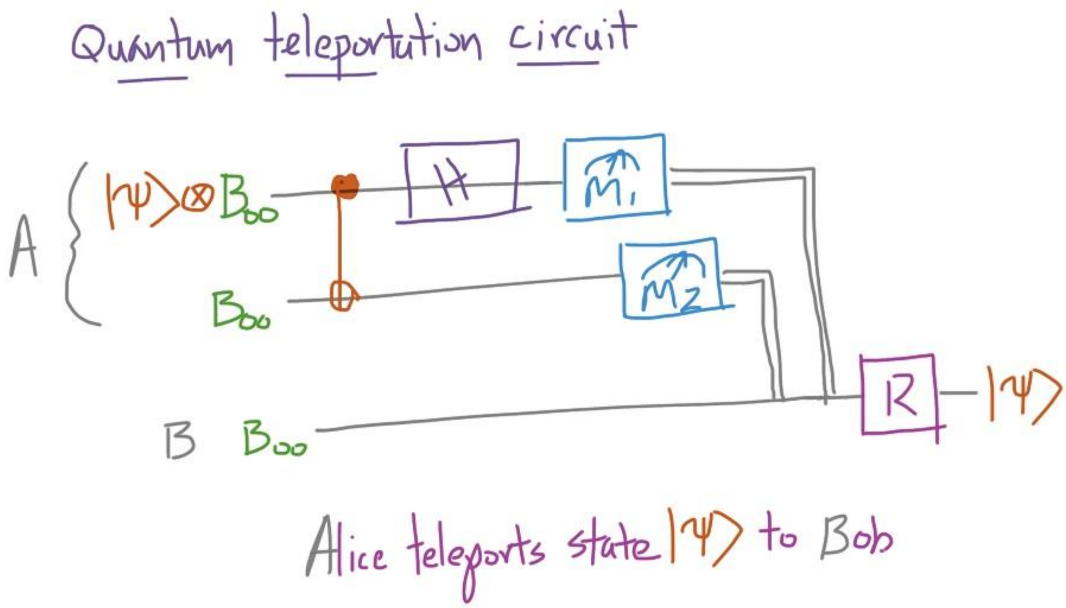


Figure 7. Circuit to prepare an entangled pair of qubits. Input qubits in this example are both $|0\rangle$. A Hadamard gate produces a mixed state in the top qubit, and a CNOT transforms the lower qubit based on that mixed state. Note that CNOT acts on the bottom $|0\rangle$ twice, first with the $\frac{|0\rangle}{\sqrt{2}}$ control and then with $\frac{|1\rangle}{\sqrt{2}}$ to produce the mixed state in the bottom wire. The output

superposition of both wires is an entangled state referred to as B_{00} , the Bell state produced when both inputs are $|0\rangle$. See if you can figure out the other Bell states, B_{01}, B_{10}, and B_{11}.

Quantum mechanics won't allow us to measure all the details of a full state, $|\psi\rangle$. When we measure a particle's spin, for example, we choose the orientation of our measuring device, say along the z-axis. Particles entering that device may have spin oriented any which way, but all we can detect is their component of spin along z. For any one particle, all that the detector can tell us is "spin up" or "spin down." If we measure lots of particles that were all prepared in the same state, then we can count how many spin up's we measure and how many spin down. ("All prepared in the same state" is key here.) Those counts give us α^2 and β^2 for the state ψ in which the particles were prepared.

With entanglement we can do wonders. Entanglement enables circuits to send two classical bits of information using just a single qubit. This "superdense coding" allows a sender, Alice, to send twice as much information to a receiver, Bob, at the same cost of computation.
Even more, entanglement allows teleportation. It's not yet (and probably never will be, because of practical limitations) the "beam me up, Scotty" teleportation of Star Trek. But it has already been accomplished in a variety of physical systems using quantum circuitry. Alice can teleport a Bell state to Bob.



<u>Figure</u> 8. Teleportation circuit. Alice, A, processes two qubits to teleport the state $|\psi\rangle$ to Bob, B. One of Alice's inputs is the tensor product of $|\psi\rangle$ with the Bell state, B_{00}. We'll skip the details of tensor products; think of this product as in standard encryption – the product resulting from multiplying the coded message times a prime number, the key. Later, if Bob knows that prime key, he can divide the product to extract the message. This teleportation circuit is a bit more complicated, but that's the essence. Alice entangles the tensor product with a second Bell qubit. Then she measures the qubits in both wires. As we've seen, measurement reads out a classical bit; that's what the double lines represent – bits rather than qubits. Alice sends those measurements to Bob over a standard circuit (hence there can be no faster-than-light teleportation). The bit pair, $00, 01, 10,$ or 11 that Alice sends is the key that Bob uses to extract $|\psi\rangle$ from his own Bell state. The replicator is one of the rotation gates, R. Bob rotates B_{00} around the *X* axis if he receives 01, around *Z* if he receives 10, and around *X* then *Z* if he receives 11. If Alice sends 00, then B_{00} itself is $|\psi\rangle$.

We won't go into the details how these circuits work, but if you'd like to look behind the curtains in this magic show, check out Nielsen and Chuang, 2008, or see [Nielsen's YouTube lectures] on the subject (Nielsen, YouTube 2014).

A final comment on these circuits and how they help to illustrate the mechanisms of quantum mechanics: We've used vectors in two-dimensional space as examples. As we've already mentioned, though, nature is a jumble of fields – electron fields and photon fields and the weak and strong fields and the Higgs field and more. To fully describe the state of any particle, we have to include all the appropriate fields. But then, maybe that's not so bad. We might not have to imagine vectors in six dimensions to represent six fields. Maybe the circuits, with appropriate gates, can simplify things. Maybe circuits can help us understand complicated field interactions in terms of information processing. Feynman diagrams, the standard representation of field interactions, sure look like quantum circuits, and their components behave like wires and gates.

Complexity

As you can see from the figures, circuit design can quickly become more complicated. Add more wires, more gates, pretty soon you have an indecipherable spider web. There's an entire discipline devoted to the study of circuit complexity and, more generally, computational complexity. As we'll see, complexity theory is providing insights not only into computer design but also into the insides of black holes.

A basic question for computer programmers (and engineers and mathematicians generally): is this problem solvable? If so, how much do I have to invest in time and computational resources in order to solve the problem? Information theorists have gained profound insights not only into computation but into the very foundations of mathematical logic trying to answer such questions. Most famous is Godel's incompleteness theorem. There exist mathematical truths that cannot be proven using the formal structure of mathematics. In the computer world a comparable puzzle is the halting problem. Can you prove whether or not a given algorithm, running on any universal computer, will ever reach a solution and stop?

Is a problem solvable using the computational tools at hand? Complexity theorists categorize problems according to the time and computer resources (i.e. memory, processor speed, etc.) required for a solution. Polynomial-time problems are those that can be solved within a time related by a polynomial function of the size of the input. For example, calculating the cost of the groceries in your shopping cart is a polynomial problem (linear in this case). Just multiply how many of each item by the cost per item, and add them all up. Done. On the other hand if your task is to figure out from first principles the energy released in nuclear fission and the resulting shock wave, the calculations rapidly blow up. A single neutron splits a uranium nucleus which releases two more neutrons which split two more nuclei resulting in four neutrons then 8 and 16 and . . . Try to track the energy release and resulting shock wave in a bomb simulation and you're dealing with exponential increase in calculations. Your computer may bog down.

In broad terms, we want to know which problems are solvable and which are not. Typically polynomial time problems are solvable and exponential problems are very difficult or impossible given present computer resources. See Scott Aaronson's book *Quantum Computing Since Democritus* for a more thorough discussion. What we're most interested in here is, given the nature of the problem, what is the most efficient design for the circuitry to solve the problem? What is the optimal configuration of wires and gates?

A wonderful surprise emerged from research into this question. It is not unheard of in the sciences to find Einstein's field equations hiding in unusual nooks and crannies. The equations for gravity-as-geometry pop up in string theory and thermodynamics and all kinds of unexpected places. But no one expected them in quantum circuits.

Michael Nielsen, an information theorist then at the University of Queensland, and his collaborators found that quantum circuits could be optimized using the rules of Riemann geometry, the rules of general relativity. Optimal circuits are, effectively, solutions to the geodesic equation – the shortest path between two events in spacetime – translated into the architecture of a quantum circuit. (Nielsen et al, 2006) For general purposes, the optimal quantum circuit is the one with the fewest gates required to output a target state given a particular input state. Given $|00101101\rangle$ what is the minimal gate set to produce $|11001011\rangle$? Nielsen et al showed it is a geodesic through the circuit.

Just think of it! It's like gravity in the machine. General relativity in the circuits. Perhaps you can mimic gravity in the quantum circuits. Turns out there are other lines of evidence that indicate you sure can.

<u>The Quantum Church-Turing Thesis</u>

Alan Turing arguably saved England in the Second World War with the computer that cracked the German Enigma code. Before that, he laid the mathematical foundations for universal computational devices, what we regard now as the modern computer. Before his discoveries computers were hard-wired to solve a specific problem, i.e. what is the trajectory of such-and-such an artillery shell fired at such-and-such an angle. Try to solve a different problem and you have to re-wire the machine. Turing showed how to design a general device that could solve any problem. All the flexibility comes in the software code, the set of instructions given to the device. Need to solve a different problem? Just change the code.

Along the way thinking about such things, Turing along with Alonso Church showed that the processes in any one universal computer could be replicated in any other universal computer. Write a program for your Mac and you could reproduce the same computations (with appropriate code) on your PC. Turing and Church were considering classical, digital computers. Dreams of quantum computers were for the future, but further mathematical logic seems to imply that any universal quantum computer should be able to reproduce the processes of any other.

We're still a long ways from universal quantum computers. In fact, a universal quantum device might be beyond reach. We may be stuck with dedicated quantum computers, like the early digital devices, designed to crack specific problems. Even so, the implications are marvelous.

Consider: nature itself is a quantum processor. Nature takes quantum information, processes it, outputs some result. Leaves capture quanta of light, photons. Photosynthesis transforms those quanta into new chemical bonds. Output is glucose. It's a computation. A really complicated one in its details, but a computation nonetheless. If Church and Turing are correct, if any quantum computation can be simulated on any other quantum device, then we should be able to model photosynthesis in a quantum computer. That's the quantum Church-Turing hypothesis. You can model any quantum system, including natural systems, on any other properly designed quantum computer. Nature in the machine.

Nature's own quantum computers are all around us. The idea that we humans could build quantum computers originated as reverse engineering. Richard Feynman, the great American physicist and educator, started it all. Standard digital computers were tackling problems at the forefront of physics, e.g. analyzing scattering amplitudes in the great particle accelerators. Feynman realized, however, that a digital machine would quickly be overwhelmed by calculations in quantum mechanics, trying to keep track of continuous variables of the state vector. Feynman's solution: simulate quantum systems on a quantum computer. Use quantum mechanics to solve quantum problems. (Feynman, 1981; and see Preskill, 2021.)

<u>Into the lab</u>

What's the payoff? Information theory, digital circuits and quantum circuits, complexity, Church-Turing. They're giving us some swell ideas, but how do we know we are really onto something? Science demands evidence.

Well, we're starting to see that evidence. Among other marvels, these notions from quantum information theory provide blueprints to experiment with black holes in the lab. Black holes in a laboratory. On quantum computers at your neighborhood university. Imagine that.

Here are a couple examples. Many more are in the works.

In 2019 Christopher Monroe and his colleagues at the University of Maryland built an entangled system of ytterbium ions that mimics the physics of information scrambling inside a black hole. (Landsman et al, 2019.) Any new qubit of information, theory predicts, diffuses rapidly throughout an entangled quantum system. You have to probe the whole system afterward in order to recover that information. Under the event horizon of a black hole, any added information should "scramble" as fast a nature allows. Turns out that's really fast, about five millionths of a second for a solar mass black hole. It's like a drop of ink dispersing in a glass of water instantaneously.

Monroe and colleagues trapped a linear array of seven ytterbium ions in an electric field. (The biggest quantum computers even as of 2023 still comprise only a few dozen qubits.) They controlled the spin states of the ions with lasers, with which they were able to entangle all the spins. Then they tweaked (e.g. flipped the spin of) the ion at one end of the array and measured the time it took for the tweak to reach the other end. Scrambling by the entanglement in such an array effectively reduces the viscosity of information transfer and reduces the arrival time. In their system, information transfer through the entangled spins was essentially instantaneous, as predicted.

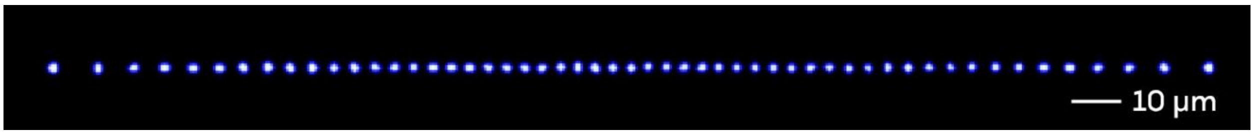


Figure 9. Array of 50 ytterbium ions held by electric field and fluorescing in laser light. For the scrambling experiment the research team used just seven ions, the maximum that could be entangled in the system. Image courtesy of the Monroe Lab, University of Maryland.

That's part of the black-holes-on-a-quantum-computer story. These days things are getting curiouser and curiouser. Now it's wormholes.

The stuff of science fiction. But in fact, wormholes were described 'way back in 1935, one of the predictions that popped out from the (relatively) new theory of general relativity. Albert Einstein and Nathan Rosen discovered wormholes in mathematical solutions to the Einstein field equations. There they were, wriggling out of the maths of general relativity. (Einstein and Rosen, 1935.)

Crazy stuff. The maths show that a black hole here can be connected, through a wormhole, to another black hole clear across the universe. And you can traverse the wormhole from here to there in a jiffy, through the wormhole doorway. Cross the universe without having to bother with silly restrictions of you-can't-go-faster-than-light. Through the wormhole and presto, you're there!

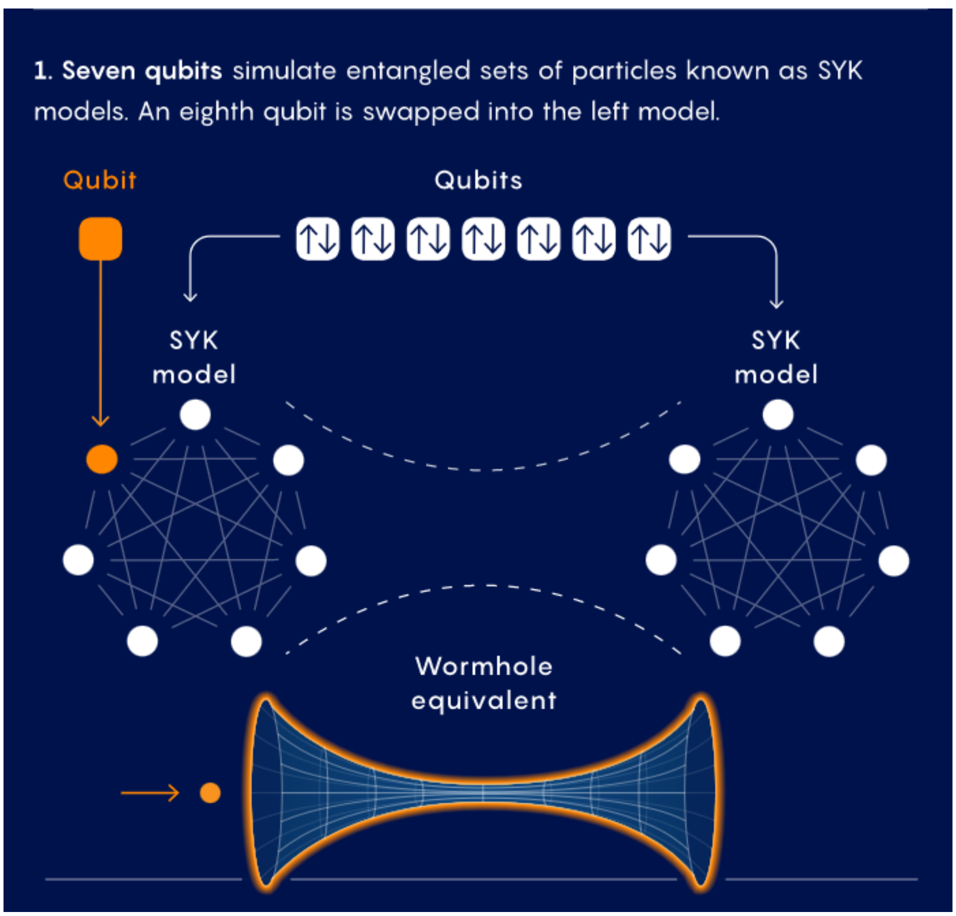
Problem is the classical "Einstein bridge" wormhole collapses soon as it's created. Before you can step in, it's gone.

Quantum maths, on the other hand, allows wormholes. Given the right conditions (a negative energy state), quantum mechanics can prop open a wormhole. It won't be wide enough for you to traverse, but you can send quantum stuff from one end to the other. Quantum stuff like qubits.

About the time of the Monroe lab's scrambling, research groups at Harvard and Caltech got to wondering if they could create a wormhole on a quantum computer. They had the design specs. The Google quantum computer lab at Santa Barbara had the machine. It was crazy enough they gave it a try. (Quanta Magazine, 2022; Wolchover, 2022.)

Instead of trapped ions, Google's Sycamore quantum computer uses superconducting circuits as qubits. A current can flow clockwise around the microscopic circuit element or anti-clockwise or in a superposition $\frac{1}{\sqrt{2}}(|\circlearrowright\rangle + |\circlearrowleft\rangle)$. Ones and zeros and superpositions. Qubits. The team created a wormhole comprising entangled qubits, stabilized it with magnetic fields, and teleported a message qubit from one side to the other. Like dropping Alice through the event horizon of one black hole and teleporting her through a wormhole and out the event horizon of Bob's black hole far far away. (Jafferis, 2022.)

Well, not quite so dramatic. The experimenters will be the first to point out there's not a real physical wormhole in the lab. But the maths are the same. The mathematical model on which the Sycamore wormhole is built is the same model as a perfectly acceptable black hole / wormhole state. By Church-Turing, they're the same. Tinker with one, poke and probe and try to figure it out, and it's just as if you're studying the other.



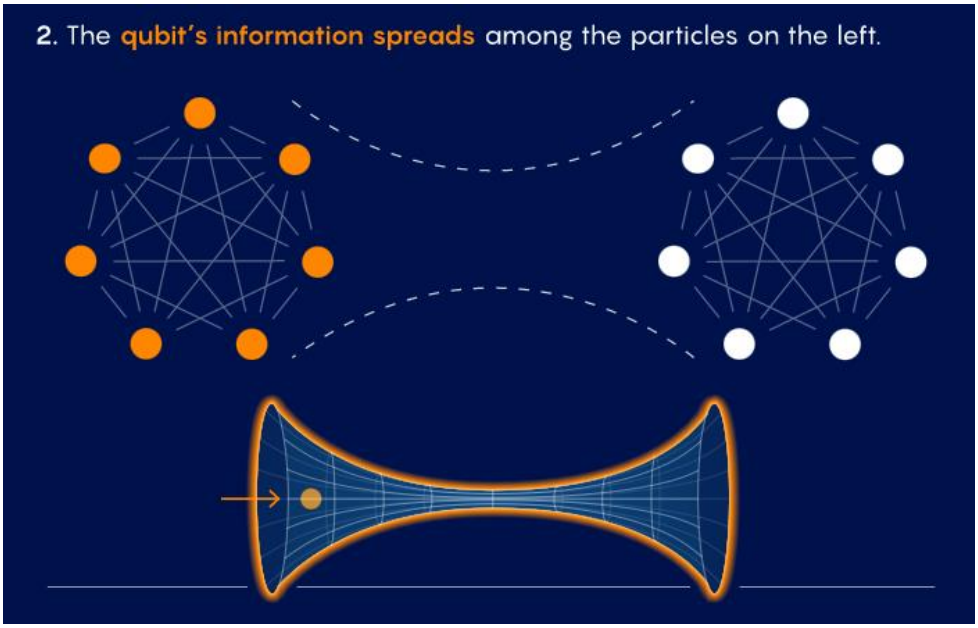


Figure 10. Wormhole in the lab. Maria Spiropulu's group at Caltech prepared a wormhole simulation on Google's Sycamore quantum computer in 2022. Shown are the first two steps in the preparation. The state of in input qubit is entangled with seven qubits (superconducting circuits in Sycamore). Then in Step 3 a magnetic pulse transfers the entangled state to the particles on the right. The state localizes to a single particle and is extracted as readout. Steps 3 and 4 (not shown) essentially reverse the process in steps 1 and 2. Image credit: Merrill Sherman, *Quanta* Magazine.

Outlook

These are just a couple examples of progress in a burgeoning field. Condensed matter physicists use these ideas to develop and explore new materials. The mathematical (SYK) model which provided the theoretical basis for the Sycamore wormhole was formulated to understand the behavior of electrons in a "strange metal" discovered by Subir Sachdev and Jinwu Ye. The model connects black holes and condensed matter and quantum computers and who knows what else. Cryptographers build entangled systems to ensure unbreakable network security. Chemists model reaction mechanisms to improve yield. And much more. For some examples of physics at the frontiers of quantum computation see the Monroe Lab web site (Monroe Lab) and presentations by Monika Schleier-Smith (Schleier-Smith, 2021). There are many other active research groups around the world. It's heady times, with lots of excitement and rapid progress.

This is a marvelous realm to explore. Ideas from information theory, quantum computation, general relativity, condensed matter physics, thermodynamics, and other disciplines are finding common ground, pushing progress toward understanding the fabric of the universe. More on that next.

Literature cited

Aaronson, Scott. 2013. Quantum computing since Democritus. Cambridge University Press.

Adam R. Brown , Daniel A. Roberts , Leonard Susskind , Brian Swingle , and Ying Zhao. 2015. Complexity equals action. https://arxiv.org/pdf/1509.07876.pdf

Brown, Adam R. and Leonard Susskind. 2018. Second law of quantum complexity. Physical Review D 97, 086015

Brown, Adam R., Leonard Susskind, and Ying Zhao. 2017. Quantum complexity and negative curvature. Physical Review D 95, 045010

Chuang, Isaac. MIT quantum computer architecture resources. https://www.media.mit.edu/quanta/qasm2circ/

Einstein, Albert and Nathaniel Rosen. 1935. The particle problem in the general theory of relativity. Physical Review 48:73.

Einstein, Albert, Boris Podolsky, and Nathaniel Rosen. 1935. Can quantum-mechanical description of physical reality be considered complete? Physical Review 47:777.

Feynman, Richard P. 1981. Simulating physics with computers. International Journal of Theoretical Physics, 21:467–488.

Feynman, Richard P. Tony Hey and Robin W. Allen, eds. 1999. Feynman lectures on computation. Westview Press.

Jafferis, Daniel et al. 2022. Traversable wormhole dynamics on a quantum processor. Nature 612: 51-55. https://www.nature.com/articles/s41586-022-05424-3

Jefferson, Ro and Robert C. Myers. 2018. Circuit complexity in quantum field theory. https://arxiv.org/pdf/1707.08570.pdf

Khan, Sal. Linear algebra. https://www.khanacademy.org/math/linear-algebra

Landauer, Rolf. 1991. Information is physical. Physics Today **44**, 5, 23

Landsman, K.A, et al. 2019. Verified quantum information scrambling. Nature 567: 61-65.

Monroe Lab at Duke University. https://iontrap.duke.edu/

Nielsen, Michael. 2014. Quantum computing for the determined. https://www.youtube.com/playlist?list=PL1826E60FD05B44E4

Nielsen, Michael A. and Isaac L. Chuang. 2017. Quantum computation and quantum information. Cambridge University Press.

Nielsen, Michael A., Mark R. Dowling, Mile Gu, and Andrew C. Doherty. 2006. Quantum computation as geometry. Science 311: 1133-1135

Preskill, John. 2017. http://www.theory.caltech.edu/people/preskill/ph229/notes/chap6.pdf

Preskill, John. 2021. Quantum computing 40 years later. https://arxiv.org/abs/2106.10522

Quanta Magazine. 2022. How physicists created a holographic wormhole in quantum computer. https://www.youtube.com/watch?v=uOJCS1W1uzg&t=2s

Schleier-Smith, Monika. 2021. Atoms linked by light. Physics at Berkeley.

Susskind, Leonard and George Hrabovski. 2013. The theoretical minimum. Basic Books.

Wolchover, Natalie. 2022. Physicist create a holographic wormhole using a quantum computer. https://www.quantamagazine.org/physicists-create-a-wormhole-using-a-quantum-computer-20221130/

Appendix

Table 1. Some of the common gates in binary circuits. A few are universal gates: NAND, Toffoli, and Fredkin among them. Complete circuits can be built using just those gates by themselves. Usually, though, and because of limitations in the physical hardware of the computer, various combinations of gates are more efficient. The truth table shows the output for any given binary input.

<table>
<tr><th>Gate</th><th>Truth table</th></tr>
<tr><td>NOT
flips the bit in its wire</td><td><table><tr><td>Input</td><td>Output</td></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table></td></tr>
<tr><td>AND
output 1 if both A and B = 1, output zero otherwise</td><td><table><tr><td>Input AB</td><td>Output C</td></tr><tr><td>00</td><td>0</td></tr><tr><td>01</td><td>0</td></tr><tr><td>10</td><td>0</td></tr><tr><td>11</td><td>1</td></tr></table></td></tr>
<tr><td>NAND (not AND)
output 0 if both A and B = 1, output 1 otherwise</td><td><table><tr><td>Input AB</td><td>Output C</td></tr><tr><td>00</td><td>1</td></tr><tr><td>01</td><td>1</td></tr><tr><td>10</td><td>1</td></tr><tr><td>11</td><td>0</td></tr></table></td></tr>
<tr><td>XOR (exclusive OR)
output 1 if either A or B = 1, output zero otherwise</td><td><table><tr><td>Input AB</td><td>Output C</td></tr><tr><td>00</td><td>0</td></tr><tr><td>01</td><td>1</td></tr><tr><td>10</td><td>1</td></tr><tr><td>11</td><td>0</td></tr></table></td></tr>
</table>

<table>
<tr><td>SWAP
exchanges bits between two wires</td><td><table><tr><th>Input AB</th><th>Output AB</th></tr><tr><td>00</td><td>00</td></tr><tr><td>01</td><td>10</td></tr><tr><td>10</td><td>01</td></tr><tr><td>11</td><td>11</td></tr></table></td></tr>
<tr><td>CNOT
flips target bit, in second wire, if bit in the input wire is 1</td><td><table><tr><th>Input AB</th><th>Output AB</th></tr><tr><td>00</td><td>00</td></tr><tr><td>01</td><td>01</td></tr><tr><td>10</td><td>11</td></tr><tr><td>11</td><td>10</td></tr></table></td></tr>
<tr><td>Toffoli gate
flips bit in target wire C if both input wires, A and B, are 1’s</td><td><table><tr><th>Input ABC</th><th>Output ABC</th></tr><tr><td>000</td><td>000</td></tr><tr><td>001</td><td>001</td></tr><tr><td>010</td><td>010</td></tr><tr><td>011</td><td>011</td></tr><tr><td>100</td><td>100</td></tr><tr><td>101</td><td>101</td></tr><tr><td>110</td><td>111</td></tr><tr><td>111</td><td>110</td></tr></table></td></tr>
<tr><td>Fredkin gate
swaps bits in target wires, B and C, if A is 1</td><td><table><tr><th>Input ABC</th><th>Output ABC</th></tr><tr><td>000</td><td>000</td></tr><tr><td>001</td><td>001</td></tr><tr><td>010</td><td>010</td></tr><tr><td>011</td><td>011</td></tr><tr><td>100</td><td>100</td></tr><tr><td>101</td><td>110</td></tr><tr><td>110</td><td>101</td></tr><tr><td>111</td><td>111</td></tr></table></td></tr>
</table>

Table 2. Standard universal set of quantum gates. Note that these gates are equivalent to vector operators – matrices – that rotate state vectors in three-dimensional vector space. For example, the action of the X gate is to rotate a vector around the X axis. We can choose X to represent direction of a physical parameter such as spin or “direction” in some other state space, such as color charge. For example, an X gate, in matrix form, operating on spin down is represented as

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

where $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$ is the vector representation of spin down and $\begin{bmatrix} 0 \\ 1 \end{bmatrix}$ is the vector spin up. With this set of gates, we can rotate state vectors to any orientation in space, i.e. we can represent any of the infinitude of states on 2-D or 3-D coordinate systems. See Figure 6.

<table>
<tr><th>Gate</th><th>Truth table or matrix form</th></tr>
<tr><td>CNOT
flips target qubit in the second wire if qubit in the input wire is 1</td><td><table>
<tr><td>Input AB</td><td>Output AB</td></tr>
<tr><td>$|00\rangle$</td><td>$|00\rangle$</td></tr>
<tr><td>$|01\rangle$</td><td>$|01\rangle$</td></tr>
<tr><td>$|10\rangle$</td><td>$|11\rangle$</td></tr>
<tr><td>$|11\rangle$</td><td>$|10\rangle$</td></tr>
</table></td></tr>
<tr><td>Hadamard
creates mixed states</td><td><table>
<tr><td>Input</td><td>Output</td></tr>
<tr><td>$|0\rangle$</td><td>$\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$</td></tr>
<tr><td>$|1\rangle$</td><td>$\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$</td></tr>
</table></td></tr>
<tr><td>X
rotates state vector around the x-axis</td><td>$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$</td></tr>
<tr><td>Y
rotates state vector around the y-axis</td><td>$\begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}$</td></tr>
</table>

Z rotates state vector around the z-axis	$\begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}$
Phase shift	$\begin{bmatrix} 1 & 0 \\ 0 & i \end{bmatrix}$
$\frac{\pi}{8}$ Phase	$\begin{bmatrix} 1 & 0 \\ 0 & e^{i\frac{\pi}{4}} \end{bmatrix}$