PROFESSOR: So today I want to finish up a couple of loose ends in the class. The first is by the end of the day, we'll talk about EPR that we've picked up on the last couple of lectures. We're going to close up Bell's theorem from the very beginning. So we talked about Bell's Inequality, and we'll re-prove Bell's Inequality, and then we'll see what quantum mechanics has to say about it.

But first, I want to show you something neat. So a couple of times, we've talked about entanglement, and what it can do. And I want to spend the first good chunk of the class talking about what entanglement can do, and also what it can't.

So before we get started, I want to remind you of a couple things from the last lecture. So the first is particles can have spin. Half-integer intrisic angular momentum that has nothing to do with being a rigid object that rotates, but is just an intrinsic property of angular momentum. And this spin is observable-- it's represented by an operator. In fact, a vector of operators, Sx, Sy, and Sz, that satisfy the same [? computational ?] relations as the orbital angular momentum.

Now last time, one thing we showed is that a state which is up-- or, for example, down-- at an angle theta, so if I measure the spin along this axis, I will always find it's other up or down, spin-1/2, either plus h-bar upon 2 or minus h-bar upon 2. But I can express this state in terms of the states with spin, which are up or down along the z-axis. And we gave an expression-- and I challenge you to follow the logic given to derive this expression. An expression for the state, which is, in this case, down along the angle theta-- it went down when measuring spin along the angle theta-- in terms of up and down of spin along the z-axis, or long angle 0, with theta here being the declination from the vertical. OK. Yep.

AUDIENCE: [INAUDIBLE] angle.

PROFESSOR: Yeah. There's a general thing for this, that says for a general angle. And it's actually in the notes, which I guess weren't posted from last time. But it's in the notes. And it's also easy to derive right? So what's the point here? The point is you have a spin vector, and it's a vector, Sx, Sy, Sz. And you can compute the operator representing spin along a particular direction by taking a unit vector in that direction and taking the dot product. So that gives you a particular linear combination of Sx, Sy, and Sz. But, from last time, you know that's just some stupid matrix. And you could express a matrix in any basis you want, for example, the basis of up and down in the z-direction, which is what we used last time. Then you can find the eigenvectors of that, and that's how you find these states.

So there's a very simple generalization. It just involves an extra angle theta, which is the angle around the equator. OK?

AUDIENCE: [INAUDIBLE].

PROFESSOR: Sorry, phi, extra angle phi. So we then did the following thing. I talked about the Stern-Gerlach experiment and, here in an abstract way from the Stern Gerlach experiment, the core of the nugget. And the core of it is this. Suppose I have some spin, and I put it in a particular state. Let's say, for simplicity, up in the z-direction plus down in the z-direction with equal amplitudes. So they're equally likely to be measured.

And then, these guys, these two states, should be degenerate in energy, because the system is rotationally invariant. It's just a spin, sitting there. There's rotational invariance, so we know the energy can't depend on the z-component of the angular momentum. But we can break that symmetry and break the degeneracy by turning on a magnetic field.

So for a Stern-Gerlach experiment, we turn on a magnetic field that had a gradient. But I just want to look at a constant magnetic field for the moment and see what it does. So we turn a constant magnetic field. That's a contribution of the energy,

which is minus the magnetic field dot the magnetic moment, which is a constant times the spin. But if the magnetic field is in the z-direction, then this is just Bz Sz.

So we then found yesterday that this led to a splitting-- that the up state had energy plus h-bar omega, and the down state had energy minus h-bar omega, where omega is given by mu naught Bz upon two. Yesterday, we had an additional term which involved the z dependence, the beta z term, but here I'm just looking at a constant magnetic field.

But if we know the energies are, these are still the eigenstates of the energy operator. If we know the energies are, we know how the system evolves in time. Here is the initial state. These are the eigenstates. The energies are these. So we can let the system evolve in the time, and we find that all we do, is we evolve the system with phases. This is as usual in quantum evolution. If you have an expansion of the state in energy eigenstates, time evolution is just phases for each term.

But note that there's a simple thing. This omega, the thing that determines the rate of evolution of the phase, is controlled by magnetic moment, which is something intrinsic to spin, and the magnetic field, which is something I can dial in an experiment. I can really tune it. Meanwhile, we decide how long to leave the magnetic field on. I can turn the magnetic field on for a while, and I could turn it off while later.

So suppose I leave the field on for time t, such that omega t is equal to pi upon 2. OK. So you turn on the magnetic field for a time t, such that omega t is pi upon 2, or t is equal to pi upon 2 omega. What is the state then, afterwards? The state subsequently psi after, is equal to this guy-- omega t is equal to pi upon 2.

But what's e to the i pi upon 2? i. Right? And e to the minus i pi over 2, is minus i. So this is an i and a minus i. I can pull out the i so that the phi after is i, a phase, upon root 2 times up z, minus down z. Yeah. That's cool. And this is a state that we saw last time. It's actually up to normalization, which is an i, up or down x in the x-direction.

So what this lets us do is something really quite cool. What this tells us is that all these calculations are just the Stern-Gerlach calculation, but even easier, because we don't have a gradient. If we have the spin in some superposition, which, incidentally, was initially up and down in the z-direction, this is equal to up in the x-direction. So in x, we have it up. We turn on a magnetic field in z, and what happens is the spin precesses around the z-axis. And we get down. We get precession.

Here's the important point. The important point for everything that's going to follow is this. If I have some spin state, and I want to take it to some other spin state, how do I do so? Well, I can build a machine that does it, by turning on the appropriate magnetic field and having the magnetic field precess the spin. OK. Here I found exactly what magnetic field I need to turn on, with what amplitude, and for how long, such that I think the state up x to down x, with a known phase i. Everyone cool with that?

So any unit-- here's the thing I want to convince you of-- any unitary operation that takes you from from one spinner to another spinner-- up z, down y, linear combination of up and down z, some particular linear combination of up and down x. Any such pair of vectors can be related by unitary transformation. And any such unitary transformation can be built by some series of magnetic fields. That cool?

You can just can prove this yourself quite quickly, but let me just state it. So now this gives us the ability to do the following. One, we have spins, one, spins that can be put in a system that can be put in states like up z and down z. And two, the ability to rotate states. The ability to evolve states from state to state. From spin state 1 to psi 2 by turning on some magnetic fields, so by suitably choosing our magnetic fields. Everyone cool with that?

So here's a question I want to ask, what do we do with this? Imagine we really had this equipment in front of us. What power, what awesomeness could we realize? And this is the entry point to quantum computing.

The answer to what can you do with this cool machinery is you build a quantum

computer. So let's talk about what that means. So quick question, do quantum computers exist? Yeah. Are they very big? No, OK. The biggest quantum computer that's been built my knowledge is one that has factored the number fifteen Or they might have done 21 at this point, I'm not exactly sure. 21? Done? Yeah, OK. 21's done. So that's the upper end.

Well, they're large, physically. It's true. They take up a room. Or at least a large table top, an optics table. But they're not very big in a useful sense.

OK. And there's actually reasonable people, not just reasonable, intelligent people-- and reasonable. That's a nontrivial-- Ok. The people who live at the intersection of intelligent and reasonable, which, admittedly as my colleague is pointing out, is not a large overlap. There are people in that overlap who suspect that, for some fundamental reason, it's impossible to build a useful quantum computer, in some deep, deep sense. I don't know whether that's true or not.

But we're about to engage in a discussion of what happens when you build quantum computers for n bits were n gets large, like millions and billions as you need for codes or for studying images. And, of course, this is a little strange, because such a compter doesn't exist, which leads to my colleague and friend Scott Aaronson's lovely comment that his job, as someone who studies quantum computers, is to prove things that can't be done with computers that don't exist. Which is what we're about to do, so.

So let's talk about quantum computing. So what do I mean by quantum computing? Here's the basic idea. Let's talk about classical computing. Classical computing says, I have these things called bits, which are binary objects. They're either 0, or they're 1.

And we realize them in a very particular way. We realize them with a little magnet, which points north or south. Now, it's not always exactly a magnet. It isn't like an old-style hard disk in your computer. It's something a little more fancy than just a magnet. But it's still basically a magnet, and you have some north, south. And whether it's pointing north, or whether it's pointing south down here, which I'll call 0.

Or whether it's pointing north down here, which I'll call a 1, determines whether you call this 0 or 1.

And classical computation is your data are a set of numbers-- what's pointing down. And the process of doing a classical computation is, build a machine which is governed by classical mechanics, OK, that takes this initial configuration and replaces it with a new one, f of 0, 1. Which is also some number, so maybe it's 0, 0. I don't know-- it's whatever f is.

And what you want to do is you want to perform some calculation, which is some function, a known function, of your input variables-- a function, in this case, of two bits, which produces the output that you want, and you build a machine that does that. OK. So let me give you an example. I want to build a classical computer that takes a string of binary integers-- 0, 0, 1, 0, 0. And performs a logical [? NOT ?] on each bit independently. So I need to build that computer out of objects available to me. OK. And I need to use nothing other than the laws of classical mechanics. That has to suffice to describe the system. So let me give an example of such a computer.

Why, Allen, why don't you do this calculation? OK. So, that's the input, and I am now the classical computer 0, 1, 1. Right? Now that's not actually what you do in your cell phone or on your laptop. That uses transistors, but it's the same basic idea. You build a set of machines. You build a set of objects, you know, magnetic fields, electric currents that effect this calculation. And so there's some relationship between what electric fields you turn on and what currents you induce to flow and the calculation you want to perform-- f of the input. OK, this is a basic idea of a classical computer.

Oh, and one last thing. Suppose we have n bits. Suppose we have n classical bits. Then we have 0, 1, the end. And how many numbers do you have to specify to specify the configuration of n bits? This is sort of [INAUDIBLE]. No, you just need to specify the number of each bit, right? So we need n bits. So n binary numbers. Everyone cool with that? I specify what each register is, and I'm done.

Imagine this is quantum mechanical. Instead of having bits, let's take quantum mechanical bits. By which I'm going to mean a system that has two possible observable states, 0 and 1. OK. So these are the possible observable states. And to specify a general configuration, I see that psi is equal to some linear combination, alpha 0 plus beta 1. Now, if I measure, we know that we'll find either 0 or 1. If we measure the spin in the z-direction, we will measure either up or down. However, at a given moment in time, the system need not be in a state corresponding to a definite value. It could be in a general superposition. Agreed?

So now here's the question. How do you how many numbers does it take to specify the state of a single quantum bit? Two complex numbers. Right? It takes two complex numbers to specify the state of a bit. And if we have n qubits, and there are n of these guys, well then, how many numbers does it take?

Well, I have to specify the state of every possible superposition for every possible configuration the system. So, for example, it could be all 0, 0, 0, dot dot dot 0, plus beta. The first one is 1, and the rest are 0, dot dot dot plus. And how it terms are there? There are 2 to the n terms. Right. So how many numbers do I need to specify this state of n qubits? I need 2 to the n complex numbers.

Yeah? Everyone see that? So this immediately leads to a really frustrating reality we have to face. Suppose you want to simulate, using a classical computer, such as sits on my desktop, I wanted to simulate a quantum mechanical system of n bits, or n spin-1/2 states, evolving according to some energy function-- evolving according to some Hamiltonian [INAUDIBLE].

How many variables, and how much memory will it take? 2 to the n, right? If I've got n bits that I want to describe, it's going to take 2 to the n complex numbers. So that's how many variables I have. So if I have 10 bits, literally, 10 little spin-1/2 objects, to accurately specify the quantum configuration system, I need 2 to the 10 complex numbers. And that's for 10 spins. How many things make this piece of chalk? Right?

So could you ever, in an honest way, simulate on a classical computer, like sits on

my desktop, a quantum mechanical system the size of this chalk. Absolutely not. It's completely and utterly intractable. You have to come up with some sort of approximation scheme.

So simulating quantum systems directly, by just taking the spins and representing them with a wave function, is wildly inefficient. Incredibly difficult. Interestingly, the converse is almost certainly not so difficult. It's almost certainly possible for a quantum system to simulate classical evolution quite easily. And how do you know that? Yeah, here we are. Exactly, right? Nature apparently has no trouble simulating classical mechanics with underlying quantum mechanics. Quantum mechanics is the operating system, and classical mechanics is the simulation it's running. Yeah, in a very useful sense, a very real sense.

So, at this point, Feynman, I think, was the first person who really pointed this out. I don't know the details of the history, but the lore is that this was really from his observation. Look, if that's true, this opens up a real possibility for computation. If things like spins-- hard problems like calculating how spins evolve or even the motion of chalk in the room-- can be done pretty efficiently by nature, a comput if we can build a computer that used quantum mechanical bits, whose variables were quantum mechanical, and involved all the superpositions and interference effects of quantum mechanics, perhaps we could build computers that run exponentially faster and with exponentially less memory and less resources than classical computer would. Because apparently it works, right? We're here, as was previously said.

So this is the question of quantum computing. Can you use quantum mechanics? Can use, in particular, the quantum evolution of the system to perform calculations that you care about, rather than classical computation? And if you do so, do you gain anything? Do you get any sort of speed ups? Is there an enhancement? Is anything better?

So, here's the basic gig. The basic gig is that we're going to take our system, considering the following kinds of systems. Our systems are going to be n qubits or

n spins. But because I want to emphasize that the substrate doesn't matter--- the underlying material doesn't matter-- I'm going to refer to them purely, and this is totally typical in the field, as qubits, rather than spin systems.

And the reason is it might work use little isolated spinning particles with intrinsic spin. Or that might turn out to be technologically infeasible. It shouldn't matter at the end of the day, in the same way that if I ask you, how does you computer work? Or if I ask you to write a code, you write some code in C or Python or whatever-- what the hip kids are using these days. So you write some little Scheme code-- I still love Scheme. You write some little Scheme code, and it performs some calculation as to defined in just logic, in lambda calculus, in abstract logic. Do you need to know the substrate? So you need to know whether your transistors are built out of silicon or germanium? Or indeed, whether it's running on vacuum tubes? No, that's the whole point of abstracting away computation. The substrate doesn't matter.

So we can use spin-1/2. And that's going to be useful for us at various moments. But I want to emphasize the important thing is the actual logic of the process, not the underlying substrate.

Here's what I need. My systems are going to be n copies, or n qubits, where each bit is specified by either 1, represented by up, or 0, represented by down. I'll use these interchangeably. So that the full system, psi, is a superposition of sum over all possible-- sorry, I didn't mean to write that.

So this is my system. It's going to evolve according to some energy operation. And so my input is going to be some wave function, psi n, for these n qubits. Computation is going to be, evolve with some energy operator-- which I've chosen-- to implement an algorithm, in the same way that the precise series of magnetic fields that we turn on in a classical computer, or currents that we allow to flow in a classical computer, implement the algorithm that we want to effect.

We evolve with some energy operator, implementing our linear algorithm, and we get some output wave function, psi-- again-- for our n bits, n quantum bits, out.

So this is the basic gig with the quantum computation. We're just replacing strings of binary numbers and functions-- evolution according to classical mechanics in Maxwell-- to other strings of numbers with superpositions of states. Superpositions of strings, as it were, evolving according to the Schrodinger equation into again general superposition.

So how does this differ? What are the key differences from a classical computer? So the key things are first off, the input and output are not definite values. They could, in general, be superpositions. They do not have to correspond to a definite state in the 1, 0 basis. These, in general, are superpositions of 0, 0, 0, 1, 0, 0, dot dot dot. So the input and output are superpositions of the values we'll measure.

OK. Second, when we do measure, at the end of the day, the output, we get interference from the various different terms in the superposition. The different terms in the superposition will lead to interference effects. And so our output will be sensitive to that interference between the different terms in the superposition. It may be that we're in some pure state. But it may be, more generally, we'll be in some superposition of the states we want to measure. So we're going to get interference.

So naively, this is a disaster because this means we get probabilistic outcomes. That sounds bad. And so, this leads to the key move in quantum computation. Given that we're going to have interference, and given that those interference effects are going to affect the probabilities of various outcomes, what we're going to want is to tune, or orchestrate, the interference to get a definite outcome. We want to get the correct answer out, rather than just probably the correct answer out.

Now there's a slight variation of this. It's not obvious at the beginning, how to do that. It's not even clear that you can. So I'm going to show you that you can. I'm going to show you an explicit algorithm that realizes this.

But, OK, that's obviously going to be tricky. Here's something else you can do. You can also focus on special problems. Focus on checkable problems. And what I mean by this is imagine we have some algorithm that gives us an output, and that output is probably the right answer. But we're not sure. There's some quantum

mechanical probability that this is the correct answer to our computation-- there's some probability that it's not.

So, if the calculation was difficult, but it's easy to check whether you have the right answer then that's great. Imagine it's 10% that you get the right answer, but it's trivial to check. So, for example, factoring numbers, right? If you multiply the numbers, that's easy. You check to see whether you got the right thing. So, for example, if you imagine that I build a computer that is supposed to factor numbers-- I almost said factor prime numbers. That would be a boring computer. So imagine that we build a machine that factors numbers. Right? And so imagine its output is probabilistic. So, you say 15, and I say 3 times 5. You say 15, I say 5 times 3. You say 15, I say 7 tiimes 2. At that point, you're like, well, which one is right? Well, you can check by multiplying.

So if you have a problem, which is easy to check, but hard to do, then probablistic is already OK. That's just as true classically as it is quantum mechanically. Those are our basic moves.

And so, the key thing for us is that when we have n quantum bits, we have these interference effects. And in particular, as we started talking about last time, we get entanglement effects. What we're going to discover is that the key to making a good quantum algorithm is going to go to attune the entanglement appropriately. So going to have to define that for you in just a minute. Yeah.

AUDIENCE: [INAUDIBLE].

PROFESSOR: Sorry?

AUDIENCE: [INAUDIBLE] process.

PROFESSOR: Well, it may or may not be. So the check, for example, , imagine the process I just gave you. So the question is, what do you do if the checking process is probabilistic? But you can use a classical computer to check, if you have an easy checking algorithm-- for example, multiplying numbers together.

AUDIENCE: [INAUDIBLE].

PROFESSOR: No. Good. And so the question then becomes, doesn't that defeat the point of using a quantum computer, if you're using a classical computer to check? And so here's the thing. If I ask you to factor a gigantic number, what do you have to do? Well, you have to take all numbers up to its square root, and you have to multiply them together, and that takes forever. Right? But if I tell you, do these two numbers multiply to give you the third number? That's easy, right? So I use a quantum computer for the hard part, and the classical computer for the forward part. For the checking. And that's a special class of problems which are checkable. These all have fancy names in information theory. I'm not going to use the fancy names.

AUDIENCE: [INAUDIBLE] the wave function?

PROFESSOR: Yeah, exactly. I mean, suppose I give you some output, and let's say the output is 0, 0 plus 1, 1. Right? What are you going to get, if you measure the first bit? Well, you're either going to get 0 or 1, with one probability or another, right?

When we compute the probabilities in general, when we have many terms in our superposition, we're going to get interference effects from different terms in the superposition. And those interference terms will tune the probability, so they're not just the naive probability of that one thing. But you get these interference terms. Norm squared of one, norm squared of two, and then the real part twice real part of the cross term. And that twice the real part of the cross term didn't exist classically. Quantum mechanically, it's important, and it can change the final probability. That's what I mean by the interference effects.

OK, so let's make all this much more explicit. So far I've just given you philosophy, and you should be deeply suspicious of philosophy in a physics classroom. So, again, to be explicit, a quantum bit, or qubit, is equal to a two state system, and, again, the substrate doesn't matter. I could be talking about spin-1/2 systems, or I could be talking about ropes on a cylinder with winding mod 2. I could be talking about all sorts of things. But it's some system with two quantum states, 0 and 1.

I want to emphasize this is not going to be on the final. So this is just for your moral well being.

So a quantum bit is a two state system. We have these two states, which I'll call 0 and 1, and a general state, the general wave function, psi, is equal to alpha 0 plus beta 1. OK. And what this means is the probability that I measure 0 is equal to norm alpha squared, et cetera. OK.

Now, again many systems can be used-- many different substrates. This is what I'm going to mean by a qubit. So that's one qubit.

The much more interesting system is two qubits. So let's study two qubits. So in the case of two qubits, what's a general state? Well a general state is going to be, well, the first particle could be 0 and the second particle could also be 0. Or we could have the first particle 0 and the second particle 1, with some coefficient, plus beta, plus gamma 1, 0, plus delta 0, 1. OK.

So this is just a general superposition. Now, you might worry at this point look, are these identical, or these non-identical spins?

But here's the thing. I've got a spin here-- it's in a box-- and I've got another spin here-- it's in a box-- I've got another qubit over here. It's in a box. So they're distinguishable because they're in different places in my laboratory. So these are distinguishable particles. The particle in this box is either up or down. OK.

So these are distinguishable, and I don't have to worry about symmetrizatoin, Bosonic or Fermionc statistics, or any of that. They're distinguishable.

And we need normalization, so norm alpha squared plus dot dot dot is equal to 1. And what does this mean? What this means is that, for example, the probability of the first part, which I'll call a is equal to 0, is equal to-- well, we sum up all the possible ways we could do that. We have norm alpha squared plus norm beta squared. Whoops-- this was 1, 1. Thank you.

And if we're more specific, the probability of the first qubit is 0, and the second qubit

is 1-- is equal to 1. I need 0, 1. That's this guy, norm beta squared. OK. What I mean by two qubits.

But this immediately leads to a funny thing. There are two kinds of states that a pair of qubits could be in, a very special set of states, which are called separable states. And these are special. And these states correspond to the full system being in the state where the first particles is in one state, and the second particle is in the second state. OK.

So, for example, this could be eg the state 1 over root 2-- sorry-- 0 plus 1 for the first particle, and the second particle to be in the state 1 over root 2, 0, minus 1. [INAUDIBLE] Let's just call these general coefficients. a plus b and times c plus d.

So what does this equal to? Well, this is of the form-- there's going to be a term that's 0, 0, and that's ac. Plus a term that's 0, 1, and that's ad with a minus. Minus ad. Plus a term that's 1, 0. That's bc. And minus a term that's 1, 1, and that's bd. OK. This is clearly not generic, because it implies relationships amongst the alpha, beta, delta, gamma, apart from just normalizability. Everyone see that?

So it's a pretty non-trivial thing that you can write a thing as, the first particle is in one state, and the second particles in the second state. And here's physically what that means. Physically, what that means is that, if you're in a state like this, and I ask you, what's the probability that I measure the first particle to be 0, do I need to know anything about the state of b? No.

If I want to know what probability of the first particle is 0, I just take norm a squared. Because I'm going to get plus norm c squared plus norm d squared. So that's fine. So, imagine the probability of the first particle being up or down is independent of any information about the second particle, right?

There is another thing that's important. Suppose I tell you, ah ha I've measured, and the first particle is in the state 0. OK. Cool? What is the state subsequent to that measurement?

So if we measure a is equal to 0, what is the state subsequent? Psi is equal to--

well, that's 0, and we know we've lost this, so this particular subsystem, this particular qubit has been collapsed to the state 0, so we have 0 times c, 0, minus d, 1 for the second particle.

Have we learned anything about the state of the second particle? Have we learned anything about the state of the second particle? Absolutely not. Right? Beforehand, what was the probability that the second article is 0? Norm c squared. And the second particle 1? Norm d squared. Now what's the problem? Same. Norm c squared, norm d squared. So when you have a seperable system, measuring one qubit tells you nothing about the other qubit. Cool?

On the other hand, consider a state which is not separable. So the generic states are not separable. And let me give you an example of the state which is not seperable. Which one do I want to pick? Yeah, what the hell.

Psi is 1 over root 2, 0, 0, plus 1, 1. Can this be written as the product of the first particle in one state and the second particle in the other? No, because they were have to be cross terms, which don't exist here. Right, just compare this to that form. So, we have that the coefficient of 0, 0 is ac. So a and c must both be non-zero, and the coefficient of 1, 1 is bd. So the coefficient bd must be-- both of those must be non-zero. So ac and bd are all non-zero. That means these terms have to exist in order for it to be separable. Yeah? Because a and b [? can't ?] vanish, and b and c [? can't ?] vanish And these orthogonal states are independent.

So this is not a separable state. We call these states entangled. And it's funny, because [INAUDIBLE] I think there's an e in there.

AUDIENCE: [INAUDIBLE]

PROFESSOR: OK. That's better. Look, I'm not a professor of spelling.

It's a little bit funny to give these a special name and call them entangled, because the generic state is entangled. It's sort of like calling mice, mice, and calling all the other mammals non-mice. Oh look, well, that was bad example. Oh look at mice right across from [INAUDIBLE].

AUDIENCE: [LAUGHS].

PROFESSOR: Hearkening back to an earlier lecture. So, in any case, we give these a special name, and the reason we give them a special name is that these seperable states do you more or less what you'd expect classically. There are no strange correlations between the state of one and the state of the other. They're independent quantities.

But when you have a generic state, something funny happens. They're entangled. And here's the funny thing that happens.

Suppose given the state, what's the probability that I measure the first particle be up, or to be 0? 1/2. What's the probability that I measure the first particle to be 1? 1/2. So, before doing any measurements, the first particle is equally likely to be 0 or 1.

So suppose I measure the second particle to be up. OK. Then the probability that I measure the first particle is 0 is equal to 0, and the probability that the first particle is 1 is equal to 1, because I've this collapse onto this term in the wave pack, the wave function.

So measuring the second qubit alters the probability distribution for the first qubit. These guys aren't independent. They are correllated, in a way that you studied on the problem set. They are correlated. They're in a correlated state. We call this correlation entanglement.

And here's the thing that's most spooky about this, and we'll come back to this in a few minutes, but I didn't tell you anything about the geometry of the set up. But, in fact, when I was thinking of that measurement, I built the two little qubits in my lab, and I took one, and I sent it to France-- because France, you know. And I took the other one, and I said it to the planet Zorg. And so off on the planet is some poor experimentalist huddling in the cold. And our French experimentalist makes a measurement, altering the probability distribution instantaneously on Zorg.

That should make you a little worried. That sounds a little crazy. We'll come back to

why that is or isn't crazy, and the EPR analysis, which puts flesh on it, sounds crazy, in a little bit.

But for the moment, let me just emphasize that, while the generic state is entangled, the generic state is also different than what your classical intuition would expect. There are correlations between the particles.

So this is something that happens with quantum mechanical particles that doesn't happen with classical particles. And that means it's something that can be used in a quantum computer that can't be used in a classical computer. So let's see what using entanglement gives us for computation.

So let's go come back to the quantum computing problem. And how do we compute with qubits? So how to compute. OK. So again, as usual, the way we take our input to our output is we build a machine that implements an algorithm of our choice by arranging the physical evolution of the system under time. So that means picking an electric field. Sorry, an energy operator-- an electric field, good lord.

So it means picking an energy operator. OK. So computing is Schroedinger evolution with our chosen, our attuned, energy operator. So, for example, I want to build for you now a couple of basic circuit elements that you might want to use in a quantum computer.

So example one. The first example is, I want to build a NOT gate And what NOT means is that it takes the state 0 and gives me 1, and it takes the state 1, and it gives me 0. OK. This is what NOT does.

So how do I build a NOT gate? Well, I can realize this in a nice way. Doot do do. Do I want to say it that way? Yeah, good.

So if I realize 0 as 1, 0, and 1 as 0, 1 from last lecture and 0, 1 to 1, 0, so we need an operator. We need time evolution to effect multiplication by an operator. That takes this vector to this vector, and this vector to this vector. We know what that unitary operation is. That unitary operation, which I'll call NOT, must equal to 0, 1, 1, 0. And this operation takes this guy to this guy, and it takes this guy to this guy.

Yeah?

But I can write this as-- I like this-- shoot, there's a phase-- minus i, e to the i pi over 2, 0, 1, 1, 0. I mean, you can't stop me, right?

So this expanded out, as we've done before, expanding this out gives me, with the exponential, 1 plus the thing and all the other terms. This becomes minus i times cosine of pi over 2 times the identity, the 2 by 2 identity, up plus i sine of pi over 2 times 0, 1, 1, 0. But what's cosine of pi upon 2? Yeah, 0. Come on. Is everyone that tired? So cosine pi over 2 is 0. Up. Sine of pi upon 2 is 1, because i times minus i, that's 1 times 0, 1, 1, 0. Pretty solid.

But what is this? Well this is the form Schroedinger evolution with a magnetic field, and this is just the unitary transformation unitary transformation for a magnetic field. Well, OK. We can say this. This is the x polymatrix from last time. So this is like Sx, unitary transformation for a magnetic field in the x-direction for some time t. For time t times the frequency, omega, which is given by mu B upon 2 is equal to pi upon 2. OK. Just like before, but for a slightly different one. A slightly different magnetic field.

So my point here is that we can pick a magnetic field that does this. We turn a magnetic field with a known amplitude with a known amount of time, details here don't matter so much. The point is we can do it. We turn a magnetic field in the x-direction, and it takes 0 to 1 and 1 to 0. Everyone cool with that?

So here is a substrate, an actual physical system that effects this particular evolution. I can build a NOT. The crucial thing is that I can build a NOT gate. And I'll represent that not with some unitary transformation U sub NOT.

So that's a useful one, but that's not the most useful gate because, if you only ever impose logical NOTs, you just get everyone angry. But you don't actually get anything done. Second example-- let that be a lesson to you, Congress.

So, the second example, if we turn on the magnetic field in the y-direction for a

particular time t, what we find is that 0 goes to 1 over root 2, 0 plus 1, and 1 goes to 1 over upon root 2 0 minus 1.

And this should be familiar. This is the up x state, and this is the down x state. Just as we talked before. So we turn on some B field, and we get this.

So this operation has a name because it's going to turn out to be very useful for us. It's taking a system that's definitely in the state 0, for sure, right? And it put us in a superposition of 0, 1. It's a definite superposition, so it's not like we don't know what happened. But it's a superposition, and you've lost certainty that you'll measure up in the z-direction. You've gained certainty that you measure up in the x-direction. But if we do all our measurements in z, we just taking ourselves from definite to superposition. Cool?

So that's useful because we know that's something a quantum computer can do, that a classical computer can't do. Something a quantum computer can take advantage of that classical computer can't take advantage of is this process of putting things into superpositions.

So here we've got an operation that puts things in superpositions. And I'll call this Hadamard. I don't know the history of why that's called Hadamard, presumably there's some guy with a last name of Hadamard. Anyway, the U Hadamard does this. And as a matrix, it's represented as 1 over root 2, times 1, 1, 1, minus 1.

And there's a the last one, which is going to be useful for me, another one is called C C-NOT. Controlled-NOT. Controlled-NOT does a really cool thing. It takes 0, 0, and 0, 1 and 1, 0, and 1, 1. What does is it says, I'm going to apply a NOT to the second qubit if, and only if, the first qubit is 1. So this takes me to-- 0, 0 goes to-- well, do I perform an NOT on this bit? No, so 0, 0. Do we perform a NOT on this bit? No, 0, 1.

Now I do perform a NOT on 0, so I get 1, 1. And 1, 1-- I perform a NOT on this bit, which gives me 1, 0.

So this is called controlled-NOT. It's a very useful thing. I'm going to represent this

in the following way. I should represent all of these.

So this NOT gate, first, I take some initial bit, and it's in some state. And then I impose U-NOT, and it gives me an out state. Similarly, with the Hadamard, I take an initial state n, and I've apply U Hadamard. And I get u out.

And for controlled-NOT, I now have two qubits. And I take the two qubits, and I do a controlled-NOT, which is represented in this form. Which says, do a NOT on this guy, controlled by this first bit. And so this acts as U C-NOT. OK.

And the key thing here, is that while it's always possible to find a physical real representation of some particular unitary transformation, at the end of the day, all we're going need is some truth table. At the end of the day, all we need is the logic that's being effected. And so, the details of the substrate can be abstracted away.

So what we do with this? First, so what can we do? Before we actually talk about what we can do with it, let's briefly talk about what you can't do with it. So what are things you can't do with these sorts of operations? What can't you do?

And to me this is among the more surprising things. Remember that what we're doing here is going to be evolving a system for a Schroedinger evolution, and a Schroedinger evolution is linear, it respects superpositions, it's unitary, it preserves probability, and let's just focus on that.

It's linear, unitary, and it lots of other properties, [? temporal ?] invariance, unless you turn on a magnetic field, which you do. But in particular, it's linear and unitary. And those facts are going to constrain, powerfully, the kinds of operations we can effect on a quantum system.

So, in particular, when we look at just two qubits, there's a beautiful, beautiful theorem says there's no cloning. And here's what that means. The no cloning theorem, pretty high flautin' for what it really is, which is the following.

Suppose we have a system, which has input xy. And I want to build a machine that says, look, I've got this first qubit it in the state x, and what I want to do is I want to

make a copy. I want to make another quantum system that's in exactly the same state as whatever x is. So you hand me a system where first bit's in state tax, and the second qubit's in state y. And I want to make a copy of x. And y is just, who cares what's in it? So I want this to go to x, x. OK. For all y. So regardless of what data was in here, I want to overwrite that data, and rewrite-- or that datem-- rewrite it with x. Can you do this? No, right. Why?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Excellent. It would violate linearity and also unitarity, indeed. So to see that quickly, it's easiest to see the unitarity, I think. Well, it violates them both, but for unitarity, you manage to take a linear combination of these guys, where the two states y are orthogonal, and you take the norm squared. So you've normalized it to 1. The linear combination of each one goes to Sx, where the coefficient is the sum of those two terms.

So we have, for example, x z, alpha x z plus beta x y goes to alpha plus beta x x. And that's really bad, because if x, z, and y are orthogonal, then normalization is alpha squared plus beta squared is 1. But x x, the normalization is alpha plus beta quantity squared is 1. And in general that's not true. In fact, this could be 0.

So this violates linearity and a unitarity rather badly. So you can't clone. This is really disturbing. That means if you have a quantum, and you want make a copy of it, you can't. You can't ever make a copy of your quantum system. One copy. One chance. That's it. No cut and paste.

So, as you can imagine that pretty powerfully constrains things that you can do.

So, a related thing here is that there's no forgetting. Quantum evolution is, unlike an elephant, it is highly-- well, it's like an elephant, I guess. It remembers very well. It never forgets anything. And you can see that from this. This would be an example of forgetting. You forgot what was in the state y. You can't ever do that. OK. So I leave this as a challenge you to prove this show. It's a simple extension of the same logic.

So what can you do? What you can do, is you can entangle two qubits. And that's really the juice of everything. You can entangle. So let me show you entanglement.

Good, no e. Sorry, question?

So you can entangle, and here's how you do it. Let's take this state 0, 0. So we have two qubits. The first one's in the state 0, and the second one is in the state 0. And now, I'm going to do the following set of operations to it. I'm first going to impose a Hadamard operation on the first qubit, and nothing on the second. And then I'm going to apply controlled-NOT, and we're going to see what I get out.

So the initial state is 0, 0. After I Hadamard, well, the first bit is no longer in 0. Hadamard on 0 gives me 0 plus 1. So this is now the state 1 upon root 2. 0 plus 1 times 0, also known as 1 over root 2, 0, 0 plus 1, 0. Now is this the separable state? Yes, there is separated.

And now I'm going to perform a controlled-NOT and what the controlled-NOT does, is that it switches the second bit, if the first bit is a 1. So what is the state after we've done this? The state after we've done this is, well, from the first term, 1 upon root 2, from the first term 0, 0-- what happens to that when we controlled-NOT? Well, we NOT this if this is 1. This is not 1, so we don't NOT it. We leave it alone. And the second term-- 1, 0 plus well, we flip this, if this is 1, and not if it's not, so this is 1. We flip it, and we get 1, 1. And now this is the prototypical entangled state-- that I think I just erased. But this is our entitled state.

It's not separable. But if I measure the first one, I know what the state of the second one is, which is to say it's entangled. Cool?

So by performing this series of operations, which is nothing other than a series of magnetic fields which I'm going to impose to the system, I've taken a state with initial conditions 0, 0, and put it into an entangled state, 0, 0 plus 1, 1. And that's all we need for the first basic algorithm of quantum computation.

So this idea the quantum computers might be able to do things faster than classical computers floated around for a while. It took a while for people to make that sharp.

And David Deutsch, who is a very entertaining and bombastic speaker, and he wrote-- I guess it's several now-- pretty entertaining books on the topic. And he sounds crazy. You listen to the guy talk, and he sounds nuts. He sounds like he's just way out there. The thing he's just-- gah! As a theorist, you listen to him like, just slow down there, buddy. Right? And so for a long time, I thought the guy-- I only knew his sort of public persona-- I thought, yeah, he's a little bit crazy; I'm not exactly sure-- and this is why everyone thinks he's such a damn genius. Because this is beautiful. So here is-- I don't know how he came up with this, but he's clever.

So here is what's now called the Deutsch-- and it's really the one bit version of the Deutsch-Josza algorithm. So there is a first algorithm by Deutsch that didn't quite what it was supposed to do, then it was improved together with Jozsa, and they made an n particle version and everything was awesome. But here's the Deutsch-Jozsa algorithm.

And what it is, it's a series of rules for how to make a quantum mechanical system evolve so as to effect the calculation you wanted to calculate. So you have to grant, to begin, you have to let me pose a problem to solve that can be solved in this fashion. And this problem is going to sound a little contrived. And, in fact, it's wildly contrived. It was contrived so that it could be solved in this fashion. But it's actually one that preexists the algorithm itself, so it's not quite as ridiculous

So here's the problem. So the statement of the problem is that someone has a function f of x. So, let's say Matt knows a function f of x. Now the thing is, it's extremely expensive to evaluate this function f of x. So the way you evaluate involves putting 20 kilomeres in superposition states with each other. You have to run a whole experiment. And it costs a lot of money to run, so he charges-- $1 million dollars order to--

AUDIENCE: [LAUGHTER]

PROFESSOR: Thank you, you guys are not quite old enough to-- so he knows the function f of x and he charges a million dollars in order to evaluate the function. You say, hey, Matt, look, I know this is a function-- which I should tell you f is a function that takes

a single bit, 0 or 1, to another single bit, 0 or 1. So it sounds like, how hard could this possibly be? But in fact, it's a very hard function to evaluate.

So you say, hey Matt, what's f of 0? And he's like, give me a million bucks. So you give him a million bucks. And he's like, 1. And you're like damn, that cost a lot of money.

So now here's the question. So this is not yet the problem. The problem is this. Is f of 0 equal to f of 1 or not? OK. So f of 0 is either 0 or 1. f of 1 is either 0 or 1. Are they equal to each other?

So this is easy, right? Classically, this is stupid. You calculate the function f twice. You evaluate f of 0, you get a number. You evaluate f of 1, and you get number. You look at your piece of paper, and you say it's either the same or different. How much does that cost you? Two million bucks. Better have good funding. So this is an expensive--

And here's what Deutsch and Josza have to say. This is really Deutsche at the beginning. It's really quite spectacular. Deutsch says actually, I tell you what, give me a million and a half, and I'll do the computation give you the answer. At which point you think, like I did previously, the guy's clearly raving. But, in fact, he's going to make a profit,

And here's how he's going to do it. He's going to build, not a classical computer, but a quantum computer using quantum interference and entanglement to do this calculation. One evaluation. And here's how it's going to work.

And the first thing we have to do is a preview, or set up, in order to do this calculation, you need two things. First off, you need Matt to be able to evaluate his function in a way that respects quantum mechanics. So, in particular, Matt had better be able to do his experiment, if I give him an superpositon. So we better be able to effect the calculation in a quantum mechanical way. The same way that we implemented a NOT quantum mechanically, or the controlled-NOT quantum mechanically, or the Hadamard, with some set of magnetic fields. He must be able

to implement it quantum mechanically. Otherwise, it's not an interesting function.

And let me just point out that any function you can think of can be implemented quantum mechanically, because you are quantum mechanics. OK? You're just not an elegant implementation-- and no offense-- of the quantum mechanical computation.

So the set up is that Matt needs to be able to give me-- Matt provides-- a unitary transformation, a unitary operation, use of f that takes two qubits, x and y to x and f of x plus y. Where what this means, f of x plus y, is addition mod two.

So what this says is, if y is 0, then this gives me f of x plus 0. If f of x is 0, that's 0 plus 0, so that gives me 0. If f of x is 1, is this gives me 1 plus 0, that's 1. On the other hand, if y is 1, then this gives me-- if f of x is 0, it gives me 1 plus 1, which is 0. And if f of x is 0, it's going to be 0 plus 1, which is 1. Everyone cool with that? Yeah.

AUDIENCE: [INAUDIBLE]. --question, but like actually, how do you know that the matrix actually [INAUDIBLE] I mean, how can we know [INAUDIBLE] if matrices actually prove that quantum mechanics [INAUDIBLE] What if the matrix is is just an approximation--

PROFESSOR: You mean what if quantum mechanics is only an approximate description--? Of the--

AUDIENCE: No I'm sorry. [INAUDIBLE] To what if quantum mechanics-- the inelegant reprensentation of [INAUDIBLE] implementation of quantum mechanics

PROFESSOR: Is inescapable--?

AUDIENCE: [INAUDIBLE] is just an approximation of the problem, or is a really, really good approximation--

PROFESSOR: This is an interesting question. So it's tempting to think that this is a philosophical question, but it turns out not to be in a way that will be made sharp in about 10 minutes with Bell's Inequality. But a complete answer to that question remains open, and, probably, always will. But let me rephrase the question slightly, and tell me if this an accurate statement.

Look, at the end of the day, what we're doing is we're going to develop a model where quantum mechanical calculation does something in particular. And that may or may not be a good model the real world. And in particular, whatever the actual thing, the actual system, that we're studying does, may or may not be well described by that quantum mechanical model. So can we check whether or not it is? Is that more or less the question?

Yeah, and so the problem is all we can ever do is say that our model is a good or bad model. On the other hand, we can do the following. And this is the really neat thing.

You might say, look, underlying quantum mechanics is going to be something more fundamental that's going to lead to slightly different results in exactly the sort of situations where we're going to care about quantum computation of large numbers and bits. And if you tell me just a little tiny bit about what properties that underlying description will have, that becomes an empirical question.

So, for example, if you say, look, I suspect that underlying the quantum mechanical probabilities is some classical probability distribution over a hidden variable that you have not actually measured. And what we're going to find out is that we can rule that out experimentally. Just that extra little assumption that there's an underlying hidden variable-- a secret probability distribution on some variable we just haven't observed yet-- that is enough information about the system to rule out that model, amazingly.

So I think we'll never have a full answer your question. But all we can do is work and see how well our models fits. And so far, nothing's ever disagreed with the quantum mechanical description.

Let me hold off on questions just now. But it's a good and interesting question that's a hard one deal with, by which I mean it's an open question.

So Matt provides for us an operator that allows us to calculate f of x. Now you might

have said, well look, why not just take x, and why not have Matt build a machine that takes x and gives you f of x. Could you have done that?

AUDIENCE: [INAUDIBLE]

PROFESSOR: Well, it's not exactly no cloning. But let me leave this to you as a fun thing to think about. Why do we need this carrier bit, as well?

OK. So there's our set up. Matt provides this function for us, U. And here's the algorithm. So the algorithm. And it's a series of steps, one by one we do them. We perform these operations are on our qubit. So here's what Deutsch says. Deutsch says start input, a state psi, is equal to 0, 1. First qubit is 0, in the state 0. The second qubit is in the state 1, for sure. We implement that with our boxes, or however we want to implement it.

So we find ourselves in a definite state, you know, hard-soft. So we take a hard box, and we take a soft box, and we pull out the hard one and the soft one.

One, Hadamard each. Hadamard on both bits. Both qubits. OK. So what does this take us to? It takes us to psi is equal to-- well, the 0 goes to 1 over root 2, times 0 plus 1.

Did I erase Hadamard? No, good. There's Hadamard. So it does this-- does this.

So it takes the first one to 0 plus 1, and it takes the second one to 1 over root 2, 0 minus 1. Cool?

So at this point, this isn't very interesting. What we've done is we take it from one superposition to a different superposition. And doesn't seem to have anything to do with f. In fact, we haven't measured f.

Two. Apply f. So we apply our operation U sub f. And well, this is a sort of an entertaining one. If we take this 1 of root 2-- so I'm going to rewrite this in a slightly simpler form-- this is 1/2, 0, times-- 0 times 0 minus 1. That's 1. Plus 1 times 0 minus 1.

And the reason I'm doing that is we're going to apply Uf. And Uf, our function f, uses that first bit as a control bit for the second. So here's the control bit for the second. So we apply Uf, and this gives us 1/2.

I'm going to actually do this on the next board, because it's going to be gigantic.

So this gives us 1/2. 0-- so that first one, this is going to take this and give it 0 plus f of 0, and 1 plus f of 0. So times f of 0, plus 0. Minus f of 0 plus 1. Plus for the 1, this going to be times f of 1, now, plus 0. Minus f of 1, plus 1. OK? Now, here's a crucial step. This is equal to, and note the following, look at this particular guy. So for that particular guy, suppose f of 0 is 0. If f of 0, is 0, then this is 0 plus 0, which is 0. So f of 0 is equal to 0. And this gives me 0, and this gives me 0 plus 1, which is 1. 0 minus 1.

On the other hand, if f of 0 is equal to 1, then we get 1 plus 0, which is 1. And here we get 1 plus 1, which is 0 minus 0, which is equal to minus 0 minus 1. Yeah? OK.

So I can write this as minus 1 to the f of 0 times 0 minus 1. Everyone cool with that? This is just a little exercise in binary arithmetic. So we can write this first term. This is 1/2 minus 1 to the f of 0, 0, times 0 minus 1.

So that's for the first one, and exactly the same logic is going to apply to the second. But now f of 1, instead of f of 0.

Plus minus 1 to the f of 1, times 1, times 0, minus 1. Now, I want to point something out to you. If f of 0 is equal to f of 1, than what's true of f of 0 plus f of 1? Well, if they're the same exactly, then either it's 0 plus 0, in which case we get 0, or it's 1 plus 1, in which case we get 0. So this is 0, if it's the same, and 1, if it's not. OK. So we could either know them both, or we can measure f of 0 plus f of 1.

So notice what happens here. This is equal to 1/2, and now I'm just going to pull out a factor of f of 0, minus 1 to the f of 0, times-- well, both of these terms have a 0 minus 1 on the second bit, so the second qubit is in the state 0 minus 1. Right, everyone cool with that?

So this is equal to, for the first qubit, 0 plus 1, times minus 1 to the f of 0, that I pulled out to square it, plus f of 1, times 0 minus 1.

And here's the quantity we wanted to measure. If this is 0, then they're even. Then they're the same. If it's 1, then they're not the same. So at this point we just forget three, forget about the second qubit. Oh, lord. Forget about the second qubit, and so forget about the second qubit just does this, just focus on this guy.

And now, if f of 0 plus f of 1 is 0, so that they're the same, this is 0, minus 1 to the 0, 0, so we get 0 plus 1. So same, then our state is 0 plus 1. And if they're different, then we get the state 0 minus 1 Everyone agree with that?

But if they're the same we get 0 plus 1, and different we get 0 minus 1. Still doesn't work for us, because if we measure, what's the probability we get 0 here? 1/2. And the probability that we get 1 is 1/2. Similar, if we measure here. It was 0, 1/2. 1, 1/2

On the other hand these states are familiar to us because they're what you get by Hadamarding. So why don't we take these, from these states to these-- by doing the inverse of the Hadamard, which, as it turns out, is Hadamard itself.

So four. Hadamard the first bit. And the output is the state, psi out, is equal to 1/2, 1 plus, minus 1 to the f of 0 plus f 1, 0 plus 1/2, 1 minus, minus 1 to the f of 0 plus f 1, 1.

And now, if f of 0 and f of 1 are the same, this is a 0. We get 1 plus 1. We get just 0, and this is 0, because this is 1, this is 1. They subtract we get 0. They're same you get this state 0, properly normalized. If they're not the same, you get this state 1, properly normalized. Now if we measure 0, we know they're the same, and if we measure 1, we know they're different.

And so with absolute certainty, now five. Measure the first qubit. And we get 0 implies the same, and 1 implies different. And we did all of this with a single evaluation of our function f, right here. This is where we apply our function evaluation. We apply the function evaluation once, and we deterministically get the result, whether they're the same or different.

So with one call to Matt, to my Oracle, with one call to Matt, which cost me one million dollars, we get the answer to whether it's the same or different. And that's a factor of 2 faster than the best classical algorithm.

But that's not so satisfying. This was supposed to be exponentially better. And so that's where Jozsa comes in, and together with Deutsche, Deutsche and Jozsa the show the following. That there's an exactly analogous problem for n qubits. Wow. There's exactly analogous from for n qubits, the Deutsche-Jozsa problem.

And now, how many different strings of integers could you put in? There are now 2 to the n possible states. And if you want to know whether f is the same for all of them, the worst case scenario is you evaluate f on the first possible combination, 0, 0, 0, 0, and you get some number. You measure f on 0, 0, 0, 0, 1, and get the same number, and just keep doing that forever. And you still don't know if they're all the same, until you get to the very last one. So, order 2 to the n is the worst case scenario. But technical scales are of a order 2 to the n.

So classically, it takes an enormous number of observations. But in the quantum Deutsche-Jozsa algorithm-- and now in the n qubit Deutsch-Jozsa problem-- one quantum operation, and you get a deterministic result. And all of this, you evaluate it once, and you know. You've solved the problem. So instead of 2 to the n operations, it takes a single one. And now, for a large number n of bits, for example, for large integers-- dealing with very large numbers-- this is dramatically, exponentially more efficient than the classical algorithm.

So at this point, people start really seriously thinking about quantum computation, whether you could get it to work. And how much we juice you can get out of actually building such a quantum computer. And this has developed into a whole theory in the whole field of the theory of computation.

The thing I want to emphasize is that the crucial move is observing that, in quantum mechanics, you can entangle degrees of freedom. The crucial move is observing that you can entangle degrees of freedom, quantum mechanically. And that's what

gave us all of the nice effects. We have these interference effects. And these interference effects lead to the deterministic outcome being correlated with the result of the computation. The interference is crucial.

And this brings us to the last point, which is exactly what's so troubling about entanglement. And so here is where EPR come in. And Einstein, Podolsky, and Rosen say the following. They say, look, there are two things that are deeply upsetting about this entanglement story. Let me just give you a precise experiment, they say. They say, let me give you precise experiment that embodies all the weirdness of this.

Suppose I take two of these qubits. And I put the qubits in an entangled state, up, up, plus down, down. OK. Let's normalize this with a 1 upon root 2. So there's our state. And then we take the first qubit, so there's our two bits, we take the first qubit, we send it somewhere faraway. And someone named Alice is sitting here and is holding on to that bit. And we take the second bit far away. And someone named Bob, conventionally, is sitting here and holds a second bit.

Now given this initial of configuration, what is the probability that Alice will measure the spin to be up? Her spin to be up. 1/2, right? And 1/2 down. Similarly, Bob 1/2 up and down. Once Alice has measured the state to be up, immediately she knows something about Bob's spin. Bob's state will be up, because I chose this one. I could have chosen the other, which is the more popular [INAUDIBLE]. So Bob's will also be up. Now if you look at this list-- you do this over and over and over again-- their list just some random list of ups and downs, ups and downs, but they're exactly correlated amongst each other.

So at this point, EPR were really upset. Because they say, look, there are two possibilities. Either there was an answer to the question all the way along of whether Alice's was up and Bob's was up, or Alice's was down and Bob's was down. Or there's some deep non-locality in the universe, such that a distant measurement, causally disconnected, can have an influence on Bob's state, such that they're correlated. This may seem random, but it's certainly not random, because it's

correlated with Alice, even though Alice is wildly disconnected, a distant observer. Nothing could have traveled across that distance in the time it took to do the measurement.

So they're sort of three responses you could take to this. The first response is, look, there isn't a problem here. It's just saying that quantum mechanics is insufficient. There's secretly a hidden variable, a variable you haven't observed yet, a property of an electron that determines whether it's going to be up or down early on. And the fact that it looks probabilistic just means that there's some classical dynamics for this hidden variable that effectively is probabilistic, like a particle moving in a fluid. Like dust pollen grains in a fluid, it just moves around randomly. Bu it just looks random, and it's not actually random. That's because there's an underlying classical mechanism controlling the probability distribution.

The second version is a quantum mechanical version. The second interpretation is to say that, look, this may look upsetting. And I grant you that it looks upsetting, but I'm a quantum mechanic. And quantum mechanics works like a champ. And I'm not about to throw it out, and say that there's some secret, hidden variables. It just works. So just give up on your naive notions of locality, let it go, and just do the quantum mechanical calculation.

Practicing physicists look at this, and just yawn. If you're a practicing physicist, you just forget it. Like, obviously, it works, so there's no more conversation to be had.

So meanwhile, there's a second version of this, which is slightly more disturbing. Suppose Alice measures up-- and this is all on the z-direction-- but Alice measures up in the z-direction. She thus knows that Bob's particle is up in the z-direction. But simultaneously, Bob could measure spin in the x-direction, and determine that his spin is up in the x-direction as well. At that point, EPR say, look, we measured, empirically, that the particle is both up in the z-direction and up in the x-direction. It's just that we did that measurement using entanglement.

But at the beginning of the day, we had that Sx and Sz don't commute. So you can't have a state with definite Sx and definite Sz. You cannot possibly. It doesn't mean

anything to say so.

Einstein wants to say that this is because quantum mechanics is great, but incomplete. The rest of us want to say that, no it's not, but that sounds like a philosophical question. And that's the way it was treated for a very long time, until you come along to Bell.

And Bill made a beautiful observation. Bell said, look, telling me that there's an underlying probabilistic classical description tells me enough to make this an empirical question. Because it's saying that the statistics, the random statistics for Bob and Alice, are correlated by a classical dynamics rather than independent.

So here's Bell's version. So remember at the very beginning, we talked about Bell's experiment. We said, consider three binary properties, a, b, and c. The number of some set that are a and not b, plus the number that are b but not c, is always greater than or equal to the number that are a but not c. And the way we proved this was just by noting that, if these are classical deterministic binary properties, then a not b means a not b and c, or a not b and not c. And ditto for each of these other guys. And we ended up with an expression which was, number that are a-- using that logically to, number that are a not b and c plus the number that are not a, b, and not c, is greater than or equal to 0. And that's clearly true. You can't have a number of elements be negative.

So this is trivially true, quantum mechanically. But now here's the experiment I want to do. I want to do actually the EPR experiment. And here's the experiment I want to run.

Alice is going to measure up or down at 0, and Bob is going to measure up or down at theta. Alice is then going to measure up and down at theta, and Bob is going to measure up and down at 2 theta. And the third experiment is going to be Alice going to measure up and down at 0, and Bob is going to measure up and down at 2 theta.

So a is up or down at-- up at 0, b is up at theta, b is up at theta-- and, sorry, this could be down at theta, not b is down a theta. b is going to be up at theta, and c is

going to be down at 2 theta. And a, again, up at 0 and not c is down at 2 theta.

So we can rephrase this as the probability that given that we are up at 0, what is the probability that we are subsequently down at theta? Plus the probability that we are up at theta, what is the probability that we're up at theta and subsequently down at 2 theta? And then the probability that we are up at 0 and down to theta, using an EPR measurement, where one is performed by Alice and the other is performed by Bob. Exactly as EPR wanted.

And we computed this last time-- in fact I just erased, because I'm excited about this, I guess-- I just erased the wave function that we needed, the state we needed.

And the state that we needed was that if we are down at the angle theta, then this is equal to cosine of theta upon 2 down at 0, plus i sin theta upon 2, up at 0. And this is enough to answer our question. This is the quantum mechanical prediction.

What's the probability that given that we're down at the angle theta, we're up at the angle 0? Well, if we're down at theta, the probably that we're up at 0-- the coefficient is i sine theta upon 2, and the probability is the norm squared of that expansion coefficient. So the probability is sine squared of theta upon 2.

Similarly, the probability that we're up at theta and down the 2 theta, well, by rotation by theta, this gives me exactly the same thing. So it's, again, going to be sine squared, sine squared theta upon 2.

The probability that we're up at 0 and down at 2 theta, well, just taking a factor of 2 for theta everywhere. And that gives me sine squared of theta.

Now, I ask you, is left the left hand side always greater than or equal to the right hand side? And this is easy to check.

Let's do this for a very small theta. For a very small theta, sine squared theta. So for small theta, much less than 1, sine theta squared is theta upon 2 the angle squared, which is equal to theta squared upon 4.

And the next one is the same thing, plus theta upon 2 squared, which is equal to

theta squared upon 4, theta squared upon 4, so theta squared upon two.

And the right hand side is sine squared theta, which is theta squared. And is theta squared upon 2 greater than or equal to theta? Certainly not.

So quantum mechanics predicts that if we do the EPR experiment, using these observables repeatedly, and built up statistics, what we'll find is an explicit violation of the Bell Inequality.

And what that would represent if it were actually true, if we actually observed it, would be a conclusive empirical proof that there are no classical definite configurations underlying the probability of quantum mechanical events. It would say that it's impossible to build a classical theory with hidden variables that are randomly distributed such that you reproduce the predictions of quantum mechanics. We see already that it doesn't agree with quantum mechanics. The question is does it agree with the real world?

So someone has to build this experiment and check. And this was done by Alain Aspect. And it violates Bell's Inequality.

There is no classical description underlying quantum mechanics. The universe around you is inescapably probabilistic. It evolves in a deterministic fashion through Schrodinger evolution.

But when we measure things, we measure results with probabilities. And those probabilities cannot be explained through some underlying classical dynamics. If there's something else underlying quantum mechanics, whatever else we know about it, is it is not classical. And this property of probabilistic evolution, or probabilistic measurement, is an inescapable and empirically verified property of the reality around us. And that's quantum mechanics. Thanks guys.