# Chapter 1

# Introduction and Overview

The course has a website at

`http://www.theory.caltech.edu/~preskill/ph229`

General information can be found there, including a course outline and links to relevant references.

Our topic can be approached from a variety of points of view, but these lectures will adopt the perspective of a theoretical physicist (that is, it's my perspective and I'm a theoretical physicist). Because of the interdisciplinary character of the subject, I realize that the students will have a broad spectrum of backgrounds, and I will try to allow for that in the lectures. Please give me feedback if I am assuming things that you don't know.

## 1.1 Physics of information

Why is a physicist teaching a course about information? In fact, the *physics of information and computation* has been a recognized discipline for at least several decades. This is natural. Information, after all, is something that is encoded in the state of a physical system; a computation is something that can be carried out on an actual physically realizable device. So the study of information and computation should be linked to the study of the underlying physical processes. Certainly, from an engineering perspective, mastery of principles of physics and materials science is needed to develop state-of-the-art computing hardware. (Carver Mead calls his Caltech research group, dedicated to advancing the art of chip design, the "Physics of Computation" (Physcmp) group).

From a more abstract theoretical perspective, there have been noteworthy milestones in our understanding of how physics constrains our ability to use and manipulate information. For example:

• **Landauer's principle**. Rolf Landauer pointed out in 1961 that erasure of information is necessarily a *dissipative* process. His insight is that erasure always involves the compression of phase space, and so is irreversible.

For example, I can store one bit of information by placing a single molecule in a box, either on the left side or the right side of a partition that divides the box. Erasure means that we move the molecule to the left side (say) irrespective of whether it started out on the left or right. I can suddenly remove the partition, and then slowly compress the one-molecule "gas" with a piston until the molecule is definitely on the left side. This procedure reduces the entropy of the gas by $\Delta S = k \ln 2$ and there is an associated flow of heat from the box to the environment. If the process is isothermal at temperature $T$, then work $W = kT \ln 2$ is performed on the box, work that I have to provide. If I am to erase information, someone will have to pay the power bill.

• **Reversible computation**. The logic gates used to perform computation are typically *irreversible*, e.g., the NAND gate

$$(a, b) \to \neg(a \wedge b) \tag{1.1}$$

has two input bits and one output bit, and we can't recover a unique input from the output bit. According to Landauer's principle, since about one bit is erased by the gate (averaged over its possible inputs), at least work $W = kT \ln 2$ is needed to operate the gate. If we have a finite supply of batteries, there appears to be a theoretical limit to how long a computation we can perform.

But Charles Bennett found in 1973 that any computation can be performed using only reversible steps, and so in principle requires no dissipation and no power expenditure. We can actually construct a reversible version of the NAND gate that preserves all the information about the input: For example, the (Toffoli) gate

$$(a, b, c) \to (a, b, c \oplus a \wedge b) \tag{1.2}$$

is a reversible 3-bit gate that flips the third bit if the first two both take the value 1 and does nothing otherwise. The third output bit becomes the NAND of $a$ and $b$ if $c = 1$. We can transform an irreversible computation to a reversible one by replacing the NAND gates by Toffoli gates. This computation could in principle be done with negligible dissipation.

However, in the process we generate a lot of extra junk, and one wonders whether we have only postponed the energy cost; we'll have to pay when we need to erase all the junk. Bennett addressed this issue by pointing out that a reversible computer can run forward to the end of a computation, print out a copy of the answer (a logically reversible operation) and then *reverse* all of its steps to return to its initial configuration. This procedure removes the junk without any energy cost.

In principle, then, we need not pay any power bill to compute. In practice, the (irreversible) computers in use today dissipate orders of magnitude more than $kT \ln 2$ per gate, anyway, so Landauer's limit is not an important engineering consideration. But as computing hardware continues to shrink in size, it may become important to beat Landauer's limit to prevent the components from melting, and then reversible computation may be the only option.

• **Maxwell's demon**. The insights of Landauer and Bennett led Bennett in 1982 to the reconciliation of Maxwell's demon with the second law of thermodynamics. Maxwell had envisioned a gas in a box, divided by a partition into two parts $A$ and $B$. The partition contains a shutter operated by the demon. The demon observes the molecules in the box as they approach the shutter, allowing fast ones to pass from $A$ to $B$, and slow ones from $B$ to $A$. Hence, $A$ cools and $B$ heats up, with a negligible expenditure of work. Heat flows from a cold place to a hot place at no cost, in apparent violation of the second law.

The resolution is that the demon must collect and store information about the molecules. If the demon has a finite memory capacity, he cannot continue to cool the gas indefinitely; eventually, information must be erased. At that point, we finally pay the power bill for the cooling we achieved. (If the demon does not erase his record, or if we want to do the thermodynamic accounting before the erasure, then we should associate some entropy with the recorded information.)

These insights were largely anticipated by Leo Szilard in 1929; he was truly a pioneer of the physics of information. Szilard, in *his* analysis of the Maxwell demon, invented the concept of a *bit* of information, (the *name* "bit" was introduced later, by Tukey) and associated the entropy $\Delta S = k \ln 2$ with the acquisition of one bit (though Szilard does not seem to have fully grasped Landauer's principle, that it is the *erasure* of the bit that carries an inevitable cost).

These examples illustrate that work at the interface of physics and information has generated noteworthy results of interest to both physicists and computer scientists.

## 1.2 Quantum information

The moral we draw is that "information is physical." and it is instructive to consider what physics has to tell us about information. But fundamentally, the universe is quantum mechanical. How does quantum theory shed light on the nature of information?

It must have been clear already in the early days of quantum theory that classical ideas about information would need revision under the new physics. For example, the clicks registered in a detector that monitors a radioactive source are described by a *truly random* Poisson process. In contrast, there is no place for true randomness in deterministic classical dynamics (although of course a complex (chaotic) classical system can exhibit behavior that is in practice indistinguishable from random).

Furthermore, in quantum theory, noncommuting observables cannot simultaneously have precisely defined values (the uncertainty principle), and in fact performing a measurement of one observable $A$ will necessarily influence the outcome of a subsequent measurement of an observable $B$, if $A$ and $B$ do not commute. Hence, the act of acquiring information about a physical system inevitably disturbs the state of the system. There is no counterpart of this limitation in classical physics.

The tradeoff between acquiring information and creating a disturbance is related to quantum randomness. It is because the outcome of a measurement has a random element that we are unable to infer the initial state of the system from the measurement outcome.

That acquiring information causes a disturbance is also connected with another essential distinction between quantum and classical information: quantum information cannot be copied with perfect fidelity (the no-cloning principle annunciated by Wootters and Zurek and by Dieks in 1982). If we *could* make a perfect copy of a quantum state, we could measure an observable of the copy without disturbing the original and we could defeat the principle of disturbance. On the other hand, nothing prevents us from copying classical information perfectly (a welcome feature when you need to back up your hard disk).

These properties of quantum information are important, but the really deep way in which quantum information differs from classical information emerged from the work of John Bell (1964), who showed that the predictions of quantum mechanics cannot be reproduced by any local hidden variable theory. Bell showed that quantum information can be (in fact, typically is) encoded in nonlocal correlations between the different parts of a physical system, correlations with no classical counterpart. We will discuss Bell's theorem in detail later on, and I will also return to it later in this lecture.

The study of quantum information as a coherent discipline began to emerge in the 1980's, and it has blossomed in the 1990's. Many of the central results of classical information theory have quantum analogs that have been discovered and developed recently, and we will discuss some of these developments later in the course, including: compression of quantum information, bounds on classical information encoded in quantum systems, bounds on quantum information sent reliably over a noisy quantum channel.

## 1.3 Efficient quantum algorithms

Given that quantum information has many unusual properties, it might have been expected that quantum theory would have a profound impact on our understanding of computation. That this is spectacularly true came to many of us as a bolt from the blue unleashed by Peter Shor (an AT&T computer scientist and a former Caltech undergraduate) in April, 1994. Shor demonstrated that, at least in principle, a quantum computer can *factor* a large number efficiently.

Factoring (finding the prime factors of a composite number) is an example of an *intractable* problem with the property:

— The solution can be *easily verified*, once found.

— But the solution is *hard* to find.

That is, if $p$ and $q$ are large prime numbers, the product $n = pq$ can be computed quickly (the number of elementary bit operations required is about $\log_2 p \cdot \log_2 q$). But given $n$, it is *hard* to find $p$ and $q$.

The time required to find the factors is strongly believed (though this has never been proved) to be *superpolynomial* in $\log(n)$. That is, as $n$ increases, the time needed in the worst case grows faster than any power of $\log(n)$. The best known factoring algorithm (the "number field sieve") requires

$$\text{time} \simeq \exp[c(\ln n)^{1/3}(\ln\ln n)^{2/3}] \tag{1.3}$$

where $c = (64/9)^{1/3} \sim 1.9$. The current state of the art is that the 65 digit factors of a 130 digit number can be found in the order of one month by a network of hundreds of work stations. Using this to estimate the prefactor in Eq. 1.3, we can estimate that factoring a 400 digit number would take about $10^{10}$ years, the age of the universe. So even with vast improvements in technology, factoring a 400 digit number will be out of reach for a while.

The factoring problem is interesting from the perspective of complexity theory, as an example of a problem presumed to be intractable; that is, a problem that can't be solved in a time bounded by a polynomial in the size of the input, in this case $\log n$. But it is also of practical importance, because the difficulty of factoring is the basis of schemes for public key cryptography, such as the widely used RSA scheme.

The exciting new result that Shor found is that a quantum computer can factor in polynomial time, *e.g.*, in time $O[(\ln n)^3]$. So if we had a quantum computer that could factor a 130 digit number in one month (of course we don't, at least not yet!), running Shor's algorithm it could factor that 400 digit number in less than 3 years. The harder the problem, the greater the advantage enjoyed by the quantum computer.

Shor's result spurred my own interest in quantum information (were it not for Shor, I don't suppose I would be teaching this course). It's fascinating to contemplate the implications for complexity theory, for quantum theory, for technology.

## 1.4 Quantum complexity

Of course, Shor's work had important antecedents. That a quantum system can perform a computation was first explicitly pointed out by Paul Benioff and Richard Feynman (independently) in 1982. In a way, this was a natural issue to wonder about in view of the relentless trend toward miniaturization in microcircuitry. If the trend continues, we will eventually approach the regime where quantum theory is highly relevant to how computing devices function. Perhaps this consideration provided some of the motivation behind Benioff's work. But Feynman's primary motivation was quite different and very interesting. To understand Feynman's viewpoint, we'll need to be more explicit about the mathematical description of quantum information and computation.

The indivisible unit of classical information is the bit: an object that can take either one of two values: 0 or 1. The corresponding unit of quantum information is the quantum bit or *qubit*. The qubit is a vector in a two-dimensional complex vector space with inner product; in deference to the classical bit we can call the elements of an orthonormal basis in this space $|0\rangle$ and $|1\rangle$. Then a normalized vector can be represented

$$|\psi\rangle = a|0\rangle + b|1\rangle, \quad |a|^2 + |b|^2 = 1. \tag{1.4}$$

where $a, b \in \mathbf{C}$. We can perform a measurement that projects $|\psi\rangle$ onto the basis $|0\rangle, |1\rangle$. The outcome of the measurement is not deterministic — the probability that we obtain the result $|0\rangle$ is $|a|^2$ and the probability that we obtain the result $|1\rangle$ is $|b|^2$.

The quantum state of $N$ qubits can be expressed as a vector in a space of dimension $2^N$. We can choose as an orthonormal basis for this space the states in which each qubit has a definite value, either $|0\rangle$ or $|1\rangle$. These can be labeled by binary strings such as

$$|01110010 \cdots 1001\rangle \tag{1.5}$$

A general normalized vector can be expanded in this basis as

$$\sum_{x=0}^{2^N-1} a_x |x\rangle \ , \tag{1.6}$$

where we have associated with each string the number that it represents in binary notation, ranging in value from 0 to $2^N - 1$. Here the $a_x$'s are complex numbers satisfying $\sum_x |a_x|^2 = 1$. If we measure all $N$ qubits by projecting each onto the $\{|0\rangle, |1\rangle\}$ basis, the probability of obtaining the outcome $|x\rangle$ is $|a_x|^2$.

Now, a quantum computation can be described this way. We assemble $N$ qubits, and prepare them in a standard initial state such as $|0\rangle|0\rangle \cdots |0\rangle$, or $|x = 0\rangle$. We then apply a unitary transformation $U$ to the $N$ qubits. (The transformation $U$ is constructed as a product of standard *quantum gates*, unitary transformations that act on just a few qubits at a time). After $U$ is applied, we measure all of the qubits by projecting onto the $\{|0\rangle, |1\rangle\}$ basis. The measurement outcome is the output of the computation. So the final output is classical information that can be printed out on a piece of paper, and published in Physical Review.

Notice that the algorithm performed by the quantum computer is a *probabilistic* algorithm. That is, we could run exactly the same program twice and obtain different results, because of the randomness of the quantum measurement process. The quantum algorithm actually generates a probability distribution of possible outputs. (In fact, Shor's factoring algorithm is not guaranteed to succeed in finding the prime factors; it just succeeds with a reasonable probability. That's okay, though, because it is easy to verify whether the factors are correct.)

It should be clear from this description that a quantum computer, though it may operate according to different physical principles than a classical computer, cannot do anything that a classical computer can't do. Classical computers can store vectors, rotate vectors, and can model the quantum measurement process by projecting a vector onto mutually orthogonal axes. So a classical computer can surely *simulate* a quantum computer to arbitrarily good accuracy. Our notion of what is *computable* will be the same, whether we use a classical computer or a quantum computer.

But we should also consider how long the simulation will take. Suppose we have a computer that operates on a modest number of qubits, like $N = 100$. Then to represent the typical quantum state of the computer, we would need to write down $2^N = 2^{100} \sim 10^{30}$ complex numbers! No existing or foreseeable digital computer will be able to do that. And performing a general rotation of a vector in a space of dimension $10^{30}$ is far beyond the computational capacity of any foreseeable classical computer.

(Of course, $N$ classical bits can take $2^N$ possible values. But for each one of these, it is very easy to write down a complete description of the configuration — a binary string of length $N$. Quantum information is very different in that writing down a complete description of just one typical configuration of $N$ qubits is enormously complex.)

So it is true that a classical computer can simulate a quantum computer, but the simulation becomes extremely inefficient as the number of qubits $N$ increases. Quantum mechanics is *hard* (computationally) because we must deal with huge matrices – there is too much room in Hilbert space. This observation led Feynman to speculate that a quantum computer would be able to perform certain tasks that are beyond the reach of any conceivable classical computer. (The quantum computer has no trouble simulating *itself*!) Shor's result seems to bolster this view.

Is this conclusion unavoidable? In the end, our simulation should provide a means of assigning probabilities to all the possible outcomes of the final measurement. It is not really necessary, then, for the classical simulation to track the complete description of the $N$-qubit quantum state. We would settle for a *probabilistic classical algorithm*, in which the outcome is not uniquely determined by the input, but in which various outcomes arise with a probability distribution that coincides with that generated by the quantum computation. We might hope to perform a *local* simulation, in which each qubit has a definite value at each time step, and each quantum gate can act on the qubits in various possible ways, one of which is selected as determined by a (pseudo)-random number generator. This simulation would be much easier than following the evolution of a vector in an exponentially large space.

But the conclusion of John Bell's powerful theorem is *precisely* that this simulation could never work: there is no *local probabilistic algorithm* that can reproduce the conclusions of quantum mechanics. Thus, while there is no known proof, it seems highly likely that simulating a quantum computer is a very hard problem for any classical computer.

To understand better why the mathematical description of quantum information is necessarily so complex, imagine we have a $3N$-qubit quantum system ($N \gg 1$) divided into three subsystems of $N$ qubits each (called subsystems (1),(2), and (3)). We randomly choose a quantum state of the $3N$ qubits, and then we separate the 3 subsystems, sending (1) to Santa Barbara and (3) to San Diego, while (2) remains in Pasadena. Now we would like to make some measurements to find out as much as we can about the quantum state. To make it easy on ourselves, let's imagine that we have a zillion copies of the state of the system so that we can measure any and all the observables we want.[1] Except for one proviso: we are restricted to carrying out each measurement within one of the subsystems — no collective measurements spanning the boundaries between the subsystems are allowed. Then for a *typical* state of the $3N$-qubit system, our measurements will reveal almost *nothing* about what the state is. Nearly all the information that distinguishes one state from another is in the *nonlocal correlations* between measurement outcomes in subsystem (1) (2), and (3). These are the nonlocal correlations that Bell found to be an essential part of the physical description.

[1]We cannot make copies of an unknown quantum state ourselves, but we can ask a friend to prepare many identical copies of the state (he can do it because he knows what the state is), and not tell us what he did.

We'll see that information content can be quantified by entropy (large entropy means little information.) If we choose a state for the $3N$ qubits randomly, we almost always find that the entropy of each subsystem is very close to

$$S \cong N - 2^{-(N+1)}, \tag{1.7}$$

a result found by Don Page. Here $N$ is the maximum possible value of the entropy, corresponding to the case in which the subsystem carries no accessible information at all. Thus, for large N we can access only an exponentially small amount of information by looking at each subsystem separately.

That is, the measurements reveal very little information if we don't consider how measurement results obtained in San Diego, Pasadena, and Santa Barbara are correlated with one another — in the language I am using, a measurement of a correlation is considered to be a "collective" measurement (even though it could actually be performed by experimenters who observe the separate parts of the same copy of the state, and then exchange phone calls to compare their results). By measuring the correlations we can learn much more; in principle, we can completely reconstruct the state.

Any satisfactory description of the state of the $3N$ qubits must characterize these nonlocal correlations, which are exceedingly complex. This is why a classical simulation of a large quantum system requires vast resources. (When such nonlocal correlations exist among the parts of a system, we say that the parts are "entangled," meaning that we can't fully decipher the state of the system by dividing the system up and studying the separate parts.)

## 1.5 Quantum parallelism

Feynman's idea was put in a more concrete form by David Deutsch in 1985. Deutsch emphasized that a quantum computer can best realize its computational potential by invoking what he called "quantum parallelism." To understand what this means, it is best to consider an example.

Following Deutsch, imagine we have a black box that computes a function that takes a single bit $x$ to a single bit $f(x)$. We don't know what is happening inside the box, but it must be something complicated, because the computation takes 24 hours. There are four possible functions $f(x)$ (because each of $f(0)$ and $f(1)$ can take either one of two possible values) and we'd like to know what the box is computing. It would take 48 hours to find out both $f(0)$ and $f(1)$.

But we don't have that much time; we need the answer in 24 hours, not 48. And it turns out that we would be satisfied to know whether $f(x)$ is *constant* ($f(0) = f(1)$) or balanced ($f(0) \neq f(1)$). Even so, it takes 48 hours to get the answer.

Now suppose we have a quantum black box that computes $f(x)$. Of course $f(x)$ might not be invertible, while the action of our quantum computer is unitary and must be invertible, so we'll need a transformation $U_f$ that takes two qubits to two:

$$U_f : |x\rangle|y\rangle \to |x\rangle|y \oplus f(x)\rangle \ . \tag{1.8}$$

(This machine flips the second qubit if $f$ acting on the first qubit is 1, and doesn't do anything if $f$ acting on the first qubit is 0.) We can determine if $f(x)$ is constant or balanced by using the quantum black box twice. But it still takes a day for it to produce one output, so that won't do. Can we get the answer (in 24 hours) by running the quantum black box *just once*. (This is "Deutsch's problem.")

Because the black box is a quantum computer, we can choose the input state to be a *superposition* of $|0\rangle$ and $|1\rangle$. If the second qubit is initially prepared in the state $\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle)$, then

$$\begin{aligned} U_f : |x\rangle\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \ &\to \ |x\rangle\frac{1}{\sqrt{2}}(|f(x)\rangle - |1 \oplus f(x)\rangle) \\ &= \ |x\rangle(-1)^{f(x)}\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle), \end{aligned} \tag{1.9}$$

so we have isolated the function $f$ in an $x$-dependent phase. Now suppose we prepare the first qubit as $\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$. Then the black box acts as

$$\begin{aligned} U_f : \quad & \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \to \\ & \frac{1}{\sqrt{2}}\left[(-1)^{f(0)}|0\rangle + (-1)^{f(1)}|1\rangle\right]\frac{1}{\sqrt{2}}(|0\rangle - |1\rangle) \ . \end{aligned} \tag{1.10}$$

Finally, we can perform a measurement that projects the first qubit onto the basis

$$|\pm\rangle = \frac{1}{\sqrt{2}}(|0\rangle \pm |1\rangle). \tag{1.11}$$

Evidently, we will always obtain $|+\rangle$ if the function is balanced, and $|-\rangle$ if the function is constant.[2]

So we have solved Deutsch's problem, and we have found a separation between what a classical computer and a quantum computer can achieve. The classical computer has to run the black box twice to distinguish a balanced function from a constant function, but a quantum computer does the job in one go!

This is possible because the quantum computer is not limited to computing either $f(0)$ or $f(1)$. It can act on a superposition of $|0\rangle$ and $|1\rangle$, and thereby extract "global" information about the function, information that depends on both $f(0)$ and $f(1)$. This is quantum parallelism.

Now suppose we are interested in global properties of a function that acts on $N$ bits, a function with $2^N$ possible arguments. To compute a complete table of values of $f(x)$, we would have to calculate $f$ $2^N$ times, completely infeasible for $N \gg 1$ (*e.g.*, $10^{30}$ times for $N = 100$). But with a quantum computer that acts according to

$$U_f : |x\rangle|0\rangle \to |x\rangle|f(x)\rangle \ , \tag{1.12}$$

we could choose the input register to be in a state

$$\left[\frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)\right]^N = \frac{1}{2^{N/2}}\sum_{x=0}^{2^N-1}|x\rangle \ , \tag{1.13}$$

and by computing $f(x)$ only once, we can generate a state

$$\frac{1}{2^{N/2}}\sum_{x=0}^{2^N-1}|x\rangle|f(x)\rangle \ . \tag{1.14}$$

Global properties of $f$ are encoded in this state, and we might be able to extract some of those properties if we can only think of an efficient way to do it.

This quantum computation exhibits "*massive* quantum parallelism;" a simulation of the preparation of this state on a classical computer would

[2] In our earlier description of a quantum computation, we stated that the final measurement would project each qubit onto the $\{|0\rangle, |1\rangle\}$ basis, but here we are allowing measurement in a different basis. To describe the procedure in the earlier framework, we would apply an appropriate unitary change of basis to each qubit before performing the final measurement.

require us to compute $f$ an unimaginably large number of times (for $N \gg 1$). Yet we have done it with the quantum computer in only one go. It is just this kind of massive parallelism that Shor invokes in his factoring algorithm.

As noted earlier, a characteristic feature of quantum information is that it can be encoded in nonlocal correlations among different parts of a physical system. Indeed, this is the case in Eq. (1.14); the properties of the function $f$ are stored as correlations between the "input register" and "output register" of our quantum computer. This nonlocal information, however, is not so easy to decipher.

If, for example, I were to measure the input register, I would obtain a result $|x_0\rangle$, where $x_0$ is chosen completely at random from the $2^N$ possible values. This procedure would prepare a state

$$|x_0\rangle|f(x_0)\rangle. \tag{1.15}$$

We could proceed to measure the output register to find the value of $f(x_0)$. But because Eq. (1.14) has been destroyed by the measurement, the intricate correlations among the registers have been lost, and we get no opportunity to determine $f(y_0)$ for any $y_0 \neq x_0$ by making further measurements. In this case, then, the quantum computation provided no advantage over a classical one.

The lesson of the solution to Deutsch's problem is that we can sometimes be more clever in exploiting the correlations encoded in Eq. (1.14). Much of the art of designing quantum algorithms involves finding ways to make efficient use of the nonlocal correlations.

## 1.6 A new classification of complexity

The computer on your desktop is not a quantum computer, but still it is a remarkable device: in principle, it is capable of performing any conceivable computation. In practice there are computations that you can't do — you either run out of time or you run out of memory. But if you provide an unlimited amount of memory, and you are willing to wait as long as it takes, then anything that deserves to be called a computation can be done by your little PC. We say, therefore, that it is a "universal computer."

Classical complexity theory is the study of which problems are hard and which ones are easy. Usually, "hard" and "easy" are defined in terms of how much time and/or memory are needed. But how can we make meaningful distinctions between hard and easy without specifying the hardware we will be using? A problem might be hard on the PC, but perhaps I could design a special purpose machine that could solve that problem much faster. Or maybe in the future a much better general purpose computer will be available that solves the problem far more efficiently. Truly meaningful distinctions between hard and easy should be *universal* — they ought not to depend on which machine we are using.

Much of complexity theory focuses on the distinction between "polynomial time" and "exponential time" algorithms. For any algorithm $A$, which can act on an input of variable length, we may associate a *complexity function* $T_A(N)$, where $N$ is the length of the input in bits. $T_A(N)$ is the longest "time" (that is, number of elementary steps) it takes for the algorithm to run to completion, for any $N$-bit input. (For example, if $A$ is a factoring algorithm, $T_A(N)$ is the time needed to factor an $N$-bit number in the worst possible case.) We say that $A$ is polynomial time if

$$T_A(N) \leq \text{ Poly } (N), \tag{1.16}$$

where Poly $(N)$ denotes a polynomial of $N$. Hence, polynomial time means that the time needed to solve the problem does not grow faster than a power of the number of input bits.

If the problem is not polynomial time, we say it is exponential time (though this is really a misnomer, because of course that are superpolynomial functions like $N^{\log N}$ that actually increase much more slowly than an exponential). This is a reasonable way to draw the line between easy and hard. But the truly compelling reason to make the distinction this way is that it is machine-independent: it does not matter what computer we are using. The universality of the distinction between polynomial and exponential follows from one of the central results of computer science: one universal (classical) computer can simulate another with at worst "polynomial overhead." This means that if an algorithm runs on your computer in polynomial time, then I can always run it on my computer in polynomial time. If I can't think of a better way to do it, I can always have my computer emulate how yours operates; the cost of running the emulation is only polynomial time. Similarly, your computer can emulate mine, so we will always agree on which algorithms are polynomial time.[3]

[3] To make this statement precise, we need to be a little careful. For example, we should exclude certain kinds of "unreasonable" machines, like a parallel computer with an unlimited number of nodes.

Now it is true that information and computation in the physical world are fundamentally quantum mechanical, but this insight, however dear to physicists, would not be of great interest (at least from the viewpoint of complexity theory) were it possible to simulate a quantum computer on a classical computer with polynomial overhead. Quantum algorithms might prove to be of technological interest, but perhaps no more so than future advances in classical algorithms that might speed up the solution of certain problems.

But if, as is indicated (but not proved!) by Shor's algorithm, no polynomial-time simulation of a quantum computer is possible, that changes everything. Thirty years of work on complexity theory will still stand as mathematical truth, as theorems characterizing the capabilities of classical universal computers. But it may fall as physical truth, because a classical Turing machine is not an appropriate model of the computations that can really be performed in the physical world.

If the quantum classification of complexity is indeed different than the classical classification (as is suspected but not proved), then this result will shake the foundations of computer science. In the long term, it may also strongly impact technology. But what is its significance for physics?

I'm not sure. But perhaps it is telling that no conceivable classical computation can accurately predict the behavior of even a modest number of qubits (of order 100). This may suggest that relatively small quantum systems have greater potential than we suspected to surprise, baffle, and delight us.

## 1.7 What about errors?

As significant as Shor's factoring algorithm may prove to be, there is another recently discovered feature of quantum information that may be just as important: the discovery of quantum error correction. Indeed, were it not for this development, the prospects for quantum computing technology would not seem bright.

As we have noted, the essential property of quantum information that a quantum computer exploits is the existence of nonlocal correlations among the different parts of a physical system. If I look at only part of the system at a time, I can decipher only very little of the information encoded in the system.

Unfortunately, these nonlocal correlations are extremely fragile and tend to decay very rapidly in practice. The problem is that our quantum system is inevitably in contact with a much larger system, its environment. It is virtually impossible to perfectly isolate a big quantum system from its environment, even if we make a heroic effort to do so. Interactions between a quantum device and its environment establish nonlocal correlations between the two. Eventually the quantum information that we initially encoded in the device becomes encoded, instead, in correlations between the device and the environment. At that stage, we can no longer access the information by observing only the device. In practice, the information is irrevocably lost. Even if the coupling between device and environment is quite weak, this happens to a macroscopic device remarkably quickly.

Erwin Schrödinger chided the proponents of the mainstream interpretation of quantum mechanics by observing that the theory will allow a quantum state of a cat of the form

$$|\mathrm{cat}\rangle = \frac{1}{\sqrt{2}}\left(|\mathrm{dead}\rangle + |\mathrm{alive}\rangle\right). \tag{1.17}$$

To Schrödinger, the possibility of such states was a blemish on the theory, because every cat he had seen was either dead or alive, not half dead and half alive.

One of the most important advances in quantum theory over the past 15 years is that we have learned how to answer Schrödinger with growing confidence. The state $|\mathrm{cat}\rangle$ is possible in principle, but is rarely seen because it is *extremely* unstable. The cats Schrödinger observed were never well isolated from the environment. If someone were to prepare the state $|\mathrm{cat}\rangle$, the quantum information encoded in the superposition of $|\mathrm{dead}\rangle$ and $|\mathrm{alive}\rangle$ would *immediately* be transferred to correlations between the cat and the environment, and become completely inaccessible. In effect, the environment continually measures the cat, projecting it onto either the state $|\mathrm{alive}\rangle$ or $|\mathrm{dead}\rangle$. This process is called *decoherence*. We will return to the study of decoherence later in the course.

Now, to perform a complex quantum computation, we need to prepare a delicate superposition of states of a relatively large quantum system (though perhaps not as large as a cat). Unfortunately, this system cannot be perfectly isolated from the environment, so this superposition, like the state $|\mathrm{cat}\rangle$, decays very rapidly. The encoded quantum information is quickly lost, and our quantum computer crashes.

To put it another way, contact between the computer and the environment (decoherence) causes *errors* that degrade the quantum information. To operate a quantum computer reliably, we must find some way to prevent or correct these errors.

Actually, decoherence is not our only problem. Even if we could achieve perfect isolation from the environment, we could not expect to operate a quantum computer with perfect accuracy. The quantum gates that the machine executes are unitary transformations that operate on a few qubits at a time, let's say $4 \times 4$ unitary matrices acting on two qubits. Of course, these unitary matrices form a continuum. We may have a protocol for applying $U_0$ to 2 qubits, but our execution of the protocol will not be flawless, so the actual transformation

$$U = U_0 \left(1 + O(\varepsilon)\right) \tag{1.18}$$

will differ from the intended $U_0$ by some amount of order $\varepsilon$. After about $1/\varepsilon$ gates are applied, these errors will accumulate and induce a serious failure. Classical analog devices suffer from a similar problem, but small errors are much less of a problem for devices that perform discrete logic.

In fact, modern digital circuits are remarkably reliable. They achieve such high accuracy with help from *dissipation*. We can envision a classical gate that acts on a bit, encoded as a ball residing at one of the two minima of a double-lobed potential. The gate may push the ball over the intervening barrier to the other side of the potential. Of course, the gate won't be implemented perfectly; it may push the ball a little too hard. Over time, these imperfections might accumulate, causing an error.

To improve the performance, we *cool* the bit (in effect) after each gate. This is a dissipative process that releases heat to the environment and compresses the phase space of the ball, bringing it close to the local minimum of the potential. So the small errors that we may make wind up heating the environment rather than compromising the performance of the device.

But we can't cool a quantum computer this way. Contact with the environment may enhance the reliability of classical information, but it would destroy encoded quantum information. More generally, accumulation of error will be a problem for classical reversible computation as well. To prevent errors from building up we need to discard the information about the errors, and throwing away information is always a dissipative process.

Still, let's not give up too easily. A sophisticated machinery has been developed to contend with errors in classical information, the theory of error correcting codes. To what extent can we coopt this wisdom to protect quantum information as well?

How does classical error correction work? The simplest example of a classical error-correcting code is a repetition code: we replace the bit we wish to protect by 3 copies of the bit,

$$\begin{aligned} 0 &\rightarrow (000), \\ 1 &\rightarrow (111). \end{aligned} \tag{1.19}$$

Now an error may occur that causes one of the three bits to flip; if it's the first bit, say,

$$\begin{aligned} (000) &\rightarrow (100), \\ (111) &\rightarrow (011). \end{aligned} \tag{1.20}$$

Now in spite of the error, we can still decode the bit correctly, by majority voting.

Of course, if the probability of error in each bit were $p$, it would be possible for two of the three bits to flip, or even for all three to flip. A double flip can happen in three different ways, so the probability of a double flip is $3p^2(1-p)$, while the probability of a triple flip is $p^3$. Altogether, then, the probability that majority voting fails is $3p^2(1-p)+p^3 = 3p^2 - 2p^3$. But for

$$3p^2 - 2p^3 < p \quad \text{or} \quad p < \frac{1}{2}, \tag{1.21}$$

the code improves the reliability of the information.

We can improve the reliability further by using a longer code. One such code (though far from the most efficient) is an $N$-bit repetition code. The probability distribution for the average value of the bit, by the central limit theorem, approaches a Gaussian with width $1/\sqrt{N}$ as $N \rightarrow \infty$. If $P = \frac{1}{2} + \varepsilon$ is the probability that each bit has the correct value, then the probability that the majority vote fails (for large $N$) is

$$P_{error} \sim e^{-N\varepsilon^2}, \tag{1.22}$$

arising from the tail of the Gaussian. Thus, for any $\varepsilon > 0$, by introducing enough redundancy we can achieve arbitrarily good reliability. Even for $\varepsilon < 0$, we'll be okay if we always assume that majority voting gives the *wrong* result. Only for $P = \frac{1}{2}$ is the cause lost, for then our block of $N$ bits will be *random*, and encode no information.

In the 50's, John Von Neumann showed that a classical computer with noisy components can work reliably, by employing sufficient redundancy. He pointed out that, if necessary, we can compute each logic gate many times, and accept the majority result. (Von Neumann was especially interested in how his brain was able to function so well, in spite of the unreliability of neurons. He was pleased to explain why he was so smart.)

But now we want to use error correction to keep a *quantum computer* on track, and we can immediately see that there are difficulties:

1. **Phase errors**. With quantum information, more things can go wrong. In addition to bit-flip errors
$$\begin{aligned} |0\rangle &\rightarrow |1\rangle, \\ |1\rangle &\rightarrow |0\rangle. \end{aligned} \tag{1.23}$$
there can also be *phase* errors
$$\begin{aligned} |0\rangle &\rightarrow |0\rangle, \\ |1\rangle &\rightarrow -|1\rangle. \end{aligned} \tag{1.24}$$
A phase error is serious, because it makes the state $\frac{1}{\sqrt{2}}[|0\rangle + |1\rangle]$ flip to the orthogonal state $\frac{1}{\sqrt{2}}[|0\rangle - |1\rangle]$. But the classical coding provided no protection against phase errors.

2. **Small errors**. As already noted, quantum information is continuous. If a qubit is intended to be in the state
$$a|0\rangle + b|1\rangle, \tag{1.25}$$
an error might change $a$ and $b$ by an amount of order $\varepsilon$, and these small errors can accumulate over time. The classical method is designed to correct large (bit flip) errors.

3. **Measurement causes disturbance**. In the majority voting scheme, it seemed that we needed to *measure* the bits in the code to detect and correct the errors. But we can't measure qubits without *disturbing* the quantum information that they encode.

4. **No cloning**. With classical coding, we protected information by making extra copies of it. But we know that quantum information cannot be copied with perfect fidelity.

## 1.8 Quantum error-correcting codes

Despite these obstacles, it turns out that quantum error correction really is possible. The first example of a quantum error-correcting code was constructed about two years ago by (guess who!) Peter Shor. This discovery ushered in a new discipline that has matured remarkably quickly – the theory of quantum error-correcting codes. We will study this theory later in the course.

Probably the best way to understand how quantum error correction works is to examine Shor's original code. It is the most straightforward quantum generalization of the classical 3-bit repetition code.

Let's look at that 3-bit code one more time, but this time mindful of the requirement that, with a quantum code, we will need to be able to correct the errors without measuring any of the encoded information.

Suppose we encode a single qubit with 3 qubits:

$$\begin{aligned} |0\rangle &\rightarrow |\bar{0}\rangle \equiv |000\rangle, \\ |1\rangle &\rightarrow |\bar{1}\rangle \equiv |111\rangle, \end{aligned} \tag{1.26}$$

or, in other words, we encode a superposition

$$a|0\rangle + b|1\rangle \rightarrow a|\bar{0}\rangle + b|\bar{1}\rangle = a|000\rangle + b|111\rangle \ . \tag{1.27}$$

We would like to be able to correct a bit flip error without destroying this superposition.

Of course, it won't do to measure a single qubit. If I measure the first qubit and get the result $|0\rangle$, then I have prepared the state $|\bar{0}\rangle$ of all three qubits, and we have lost the quantum information encoded in the coefficients $a$ and $b$.

But there is no need to restrict our attention to single-qubit measurements. I could also perform collective measurements on two-qubits at once, and collective measurements suffice to diagnose a bit-flip error. For a 3-qubit state $|x, y, z\rangle$ I could measure, say, the two-qubit observables $y \oplus z$, or $x \oplus z$ (where $\oplus$ denotes addition modulo 2). For both $|x, y, z\rangle = |000\rangle$ and $|111\rangle$ these would be 0, but if any one bit flips, then at least one of these quantities will be 1. In fact, if there is a single bit flip, the two bits

$$(y \oplus z, x \oplus z), \tag{1.28}$$

just designate in binary notation the position (1,2 or 3) of the bit that flipped. These two bits constitute a *syndrome* that diagnoses the error that occurred.

For example, if the first bit flips,

$$a|000\rangle + b|111\rangle \to a|100\rangle + b|011\rangle, \tag{1.29}$$

then the measurement of $(y \oplus z, x \oplus z)$ yields the result $(0, 1)$, which instructs us to flip the first bit; this indeed repairs the error.

Of course, instead of a (large) bit flip there could be a small error:

$$\begin{aligned} |000\rangle &\to |000\rangle + \varepsilon|100\rangle \\ |111\rangle &\to |111\rangle - \varepsilon|011\rangle. \end{aligned} \tag{1.30}$$

But even in this case the above procedure would work fine. In measuring $(y \oplus z, x \oplus z)$, we would project out an eigenstate of this observable. Most of the time (probability $1 - |\varepsilon|^2$) we obtain the result $(0, 0)$ and project the damaged state back to the original state, and so correct the error. Occasionally (probability $|\varepsilon|^2$) we obtain the result $(0, 1)$ and project the state onto Eq. 1.29. But then the syndrome instructs us to flip the first bit, which restores the original state. Similarly, if there is an amplitude of order $\varepsilon$ for each of the three qubits to flip, then with a probability of order $|\varepsilon|^2$ the syndrome measurement will project the state to one in which one of the three bits is flipped, and the syndrome will tell us which one.

So we have already overcome 3 of the 4 obstacles cited earlier. We see that it is possible to make a measurement that diagnoses the error without damaging the information (answering (3)), and that a quantum measurement can project a state with a small error to either a state with no error or a state with a large discrete error that we know how to correct (answering (2)). As for (4), the issue didn't come up, because the state $a|\bar{0}\rangle + b|\bar{1}\rangle$ is not obtained by cloning – it is not the same as $(a|0\rangle + b|1\rangle)^3$; that is, it differs from three copies of the unencoded state.

Only one challenge remains: (1) phase errors. Our code does not yet provide any protection against phase errors, for if any one of the three qubits undergoes a phase error then our encoded state $a|\bar{0}\rangle + b|\bar{1}\rangle$ is transformed to $a|\bar{0}\rangle - b|\bar{1}\rangle$, and the encoded quantum information is damaged. In fact, phase errors have become three times more likely than if we hadn't used the code. But with the methods in hand that conquered problems (2)-(4), we can approach problem (1) with new confidence. Having protected against bit-flip errors by encoding bits redundantly, we are led to protect against phase-flip errors by encoding phases redundantly.

Following Shor, we encode a single qubit using nine qubits, according to

$$
\begin{aligned}
|0\rangle \rightarrow |\bar{0}\rangle &\equiv \frac{1}{2^{3/2}}\left(|000\rangle + |111\rangle\right)\left(|000\rangle + |111\rangle\right)\left(|000\rangle + |111\rangle\right), \\
|1\rangle \rightarrow |\bar{1}\rangle &\equiv \frac{1}{2^{3/2}}\left(|000\rangle - |111\rangle\right)\left(|000\rangle - |111\rangle\right)\left(|000\rangle - |111\rangle\right). \qquad (1.31)
\end{aligned}
$$

Both $|\bar{0}\rangle$ and $|\bar{1}\rangle$ consist of three clusters of three qubits each, with each cluster prepared in the same quantum state. Each of the clusters has triple bit redundancy, so we can correct a single bit flip in any cluster by the method discussed above.

Now suppose that a phase flip occurs in one of the clusters. The error changes the relative sign of $|000\rangle$ and $|111\rangle$ in that cluster so that

$$
\begin{aligned}
|000\rangle + |111\rangle &\rightarrow |000\rangle - |111\rangle, \\
|000\rangle - |111\rangle &\rightarrow |000\rangle + |111\rangle. \qquad (1.32)
\end{aligned}
$$

This means that the relative phase of the damaged cluster differs from the phases of the other two clusters. Thus, as in our discussion of bit-flip correction, we can identify the damaged cluster, not by *measuring* the relative phase in each cluster (which would disturb the encoded information) but by *comparing* the phases of pairs of clusters. In this case, we need to measure a six-qubit observable to do the comparison, e.g., the observable that flips qubits 1 through 6. Since flipping twice is the identity, this observable squares to 1, and has eigenvalues $\pm 1$. A pair of clusters with the same sign is an eigenstate with eigenvalue $+1$, and a pair of clusters with opposite sign is an eigenstate with eigenvalue $-1$. By measuring the six-qubit observable for a second pair of clusters, we can determine which cluster has a different sign than the others. Then, we apply a unitary phase transformation to one of the qubits in that cluster to reverse the sign and correct the error.

Now suppose that a unitary error $U = 1 + 0(\varepsilon)$ occurs for each of the 9 qubits. The most general single-qubit unitary transformation (aside from a physically irrelevant overall phase) can be expanded to order $\varepsilon$ as

$$
U = 1 + i\varepsilon_x \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} + i\varepsilon_y \begin{pmatrix} 0 & -i \\ i & 0 \end{pmatrix} + i\varepsilon_z \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}. \qquad (1.33)
$$

the three terms of order $\varepsilon$ in the expansion can be interpreted as a bit flip operator, a phase flip operator, and an operator in which both a bit flip and a phase flip occur. If we prepare an encoded state $a|\bar{0}\rangle + b|\bar{1}\rangle$, allow the unitary errors to occur on each qubit, and then measure the bit-flip and phase-flip syndromes, then most of the time we will project the state back to its original form, but with a probability of order $|\varepsilon|^2$, one qubit will have a large error: a bit flip, a phase flip, or both. From the syndrome, we learn which bit flipped, and which cluster had a phase error, so we can apply the suitable one-qubit unitary operator to fix the error.

Error recovery will fail if, after the syndrome measurement, there are two bit flip errors in each of two clusters (which induces a phase error in the encoded data) or if phase errors occur in two different clusters (which induces a bit-flip error in the encoded data). But the probability of such a double phase error is of order $|\varepsilon|^4$. So for $|\varepsilon|$ small enough, coding improves the reliability of the quantum information.

The code also protects against decoherence. By restoring the quantum state irrespective of the nature of the error, our procedure removes any entanglement between the quantum state and the environment.

Here as always, error correction is a dissipative process, since information about the nature of the errors is flushed out of the quantum system. In this case, that information resides in our recorded measurement results, and heat will be dissipated when that record is erased.

Further developments in quantum error correction will be discussed later in the course, including:

• As with classical coding it turns out that there are "good" quantum codes that allow us to achieve arbitrarily high reliability as long as the error rate per qubit is small enough.

• We've assumed that the error recovery procedure is itself executed flawlessly. But the syndrome measurement was complicated – we needed to measure two-qubit and six-qubit collective observables to diagnose the errors – so we actually might further damage the data when we try to correct it. We'll show, though, that error correction can be carried out so that it still works effectively even if we make occasional errors during the recovery process.

• To operate a quantum computer we'll want not only to store quantum information reliably, but also to process it. We'll show that it is possible to apply quantum gates to encoded information.

Let's summarize the essential ideas that underlie our quantum error correction scheme:

1. We *digitized* the errors. Although the errors in the quantum information were small, we performed measurements that projected our state onto either a state with no error, or a state with one of a discrete set of errors that we knew how to convert.

2. We measured the errors without measuring the data. Our measurements revealed the nature of the errors without revealing (and hence disturbing) the encoded information.

3. The errors are local, and the encoded information is nonlocal. It is important to emphasize the central assumption underlying the construction of the code – that errors affecting different qubits are, to a good approximation, *uncorrelated*. We have tacitly assumed that an event that causes errors in two qubits is much less likely than an event causing an error in a single qubit. It is of course a physics question whether this assumption is justified or not – we can easily envision processes that will cause errors in two qubits at once. If such correlated errors are common, coding will fail to improve reliability.

The code takes advantage of the presumed local nature of the errors by encoding the information in a nonlocal way - that is the information is stored in *correlations* involving several qubits. There is no way to distinguish $|\bar{0}\rangle$ and $|\bar{1}\rangle$ by measuring a single qubit of the nine. If we measure one qubit we will find $|0\rangle$ with probability $\frac{1}{2}$ and $|1\rangle$ with probability $\frac{1}{2}$ irrespective of the value of the encoded qubit. To access the encoded information we need to measure a 3-qubit observable (the operator that flips all three qubits in a cluster can distinguish $|000\rangle + |111\rangle$ from $|000\rangle - |111\rangle$).

The environment might occasionally kick one of the qubits, in effect "measuring" it. But the encoded information cannot be damaged by disturbing that one qubit, because a single qubit, by itself, actually carries no information at all. Nonlocally encoded information is invulnerable to local influences – this is the central principle on which quantum error-correcting codes are founded.

## 1.9 Quantum hardware

The theoretical developments concerning quantum complexity and quantum error correction have been accompanied by a burgeoning experimental effort to process coherent quantum information. I'll briefly describe some of this activity here.

To build hardware for a quantum computer, we'll need technology that enables us to manipulate qubits. The hardware will need to meet some stringent specifications:

1. **Storage**: We'll need to store qubits for a long time, long enough to complete an interesting computation.

2. **Isolation**: The qubits must be well isolated from the environment, to minimize decoherence errors.

3. **Readout**: We'll need to measure the qubits efficiently and reliably.

4. **Gates**: We'll need to manipulate the quantum states of individual qubits, and to induce controlled interactions among qubits, so that we can perform quantum gates.

5. **Precision**: The quantum gates should be implemented with high precision if the device is to perform reliably.

### 1.9.1 Ion Trap

One possible way to achieve these goals was suggested by Ignacio Cirac and Peter Zoller, and has been pursued by Dave Wineland's group at the National Institute for Standards and Technology (NIST), as well as other groups. In this scheme, each qubit is carried by a single ion held in a linear Paul trap. The quantum state of each ion is a linear combination of the ground state $|g\rangle$ (interpreted as $|0\rangle$) and a particular long-lived metastable excited state $|e\rangle$ (interpreted as $|1\rangle$). A coherent linear combination of the two levels,

$$a|g\rangle + be^{i\omega t}|e\rangle, \tag{1.34}$$

can survive for a time comparable to the lifetime of the excited state (though of course the relative phase oscillates as shown because of the energy splitting $\hbar\omega$ between the levels). The ions are so well isolated that spontaneous decay can be the dominant form of decoherence.

It is easy to read out the ions by performing a measurement that projects onto the $\{|g\rangle, |e\rangle\}$ basis. A laser is tuned to a transition from the state $|g\rangle$ to a short-lived excited state $|e'\rangle$. When the laser illuminates the ions, each qubit with the value $|0\rangle$ repeatedly absorbs and reemits the laser light, so that it flows visibly (fluoresces). Qubits with the value $|1\rangle$ remain dark.

Because of their mutual Coulomb repulsion, the ions are sufficiently well separated that they can be individually addressed by pulsed lasers. If a laser is tuned to the frequency $\omega$ of the transition and is focused on the $n$th ion, then Rabi oscillations are induced between $|0\rangle$ and $|1\rangle$. By timing the laser pulse properly and choosing the phase of the laser appropriately, we can apply any one-qubit unitary transformation. In particular, acting on $|0\rangle$, the laser pulse can prepare any desired linear combination of $|0\rangle$ and $|1\rangle$.

But the most difficult part of designing and building quantum computing hardware is getting two qubits to interact with one another. In the ion trap, interactions arise because of the Coulomb repulsion between the ions. Because of the mutual Couloumb repulsion, there is a spectrum of coupled normal modes of vibration for the trapped ions. When the ion absorbs or emits a laser photon, the center of mass of the ion recoils. But if the laser is properly tuned, then when a single ion absorbs or emits, a normal mode involving many ions will recoil coherently (the Mössbauer effect).

The vibrational mode of lowest frequency (frequency $\nu$) is the center-of-mass ($cm$) mode, in which the ions oscillate in lockstep in the harmonic well of the trap. The ions can be laser cooled to a temperature much less than $\nu$, so that each vibrational mode is very likely to occupy its quantum-mechanical ground state. Now imagine that a laser tuned to the frequency $\omega - \nu$ shines on the $n$th ion. For a properly time pulse the state $|e\rangle_n$ will rotate to $|g\rangle_n$, while the $cm$ oscillator makes a transition from its ground state $|0\rangle_{cm}$ to its first excited state $|1\rangle_{cm}$ (a $cm$ "phonon" is produced). However, the state $|g\rangle_n|0\rangle_{cm}$ is not on resonance for any transition and so is unaffected by the pulse. Thus the laser pulse induces a unitary transformation acting as

$$\begin{aligned} |g\rangle_n|0\rangle_{cm} &\to |g\rangle_n|0\rangle_{cm}, \\ |e\rangle_n|0\rangle_{cm} &\to -i|g\rangle_n|1\rangle_{cm}. \end{aligned} \tag{1.35}$$

This operation removes a bit of information that is initially stored in the internal state of the $n$th ion, and deposits that bit in the *collective* state of motion of *all* the ions.

This means that the state of motion of the $m$th ion ($m \neq n$) has been influenced by the internal state of the $n$th ion. In this sense, we have succeeded in inducing an interaction between the ions. To complete the quantum gate, we should transfer the quantum information from the $cm$ phonon back to the internal state of one of the ions. The procedure should be designed so that the $cm$ mode always returns to its ground state $|0\rangle_{cm}$ at the conclusion of the gate implementation. For example, Cirac and Zoller showed that the quantum $XOR$ (or controlled not) gate

$$|x,y\rangle \to |x, y \oplus x\rangle, \tag{1.36}$$

can be implemented in an ion trap with altogether 5 laser pulses. The conditional excitation of a phonon, Eq. (1.35) has been demonstrated experimentally, for a single trapped ion, by the NIST group.

One big drawback of the ion trap computer is that it is an intrinsically slow device. Its speed is ultimately limited by the energy-time uncertainty relation. Since the uncertainty in the energy of the laser photons should be small compared to the characteristic vibrational splitting $\nu$, each laser pulse should last a time long compared to $\nu^{-1}$. In practice, $\nu$ is likely to be of order 100 kHz.

### 1.9.2 Cavity QED

An alternative hardware design (suggested by Pellizzari, Gardiner, Cirac, and Zoller) is being pursued by Jeff Kimble's group here at Caltech. The idea is to trap several neutral atoms inside a small high finesse optical cavity. Quantum information can again be stored in the internal states of the atoms. But here the atoms interact because they all couple to the normal modes of the electromagnetic field in the cavity (instead of the vibrational modes as in the ion trap). Again, by driving transitions with pulsed lasers, we can induce a transition in one atom that is conditioned on the internal state of another atom.

Another possibility is to store a qubit, not in the internal state of an ion, but in the polarization of a photon. Then a trapped atom can be used as the intermediary that causes one photon to interact with another (instead of a photon being used to couple one atom to another). In their "flying qubit" experiment two years ago. The Kimble group demonstrated the operation of a two-photon quantum gate, in which the circular polarization of one photon influences the *phase* of another photon:

$$\begin{aligned}
|L\rangle_1|L\rangle_2 &\rightarrow |L\rangle_1|L\rangle_2 \\
|L\rangle_1|R\rangle_2 &\rightarrow |L\rangle_1|R\rangle_2 \\
|R\rangle_1|L\rangle_2 &\rightarrow |R\rangle_1|L\rangle_2 \\
|R\rangle_1|R\rangle_2 &\rightarrow e^{i\Delta}|R\rangle_1|R\rangle_2
\end{aligned} \tag{1.37}$$

where $|L\rangle, |R\rangle$ denote photon states with left and right circular polarization. To achieve this interaction, one photon is stored in the cavity, where the $|L\rangle$ polarization does not couple to the atom, but the $|R\rangle$ polarization couples strongly. A second photon transverses the cavity, and for the second photon as well, one polarization interacts with the atom preferentially. The second photon wave pocket acquires a particular phase shift $e^{i\Delta}$ only if both photons have $|R\rangle$ polarization. Because the phase shift is conditioned on the polarization of *both* photons, this is a nontrivial two-qubit quantum gate.

### 1.9.3 NMR

A third (dark horse) hardware scheme has sprung up in the past year, and has leap frogged over the ion trap and cavity QED to take the current lead in coherent quantum processing. The new scheme uses nuclear magnetic resonance (NMR) technology. Now qubits are carried by certain nuclear spins in a particular molecule. Each spin can either be aligned ($|\uparrow\rangle = |0\rangle$) or antialigned ($|\downarrow\rangle = |1\rangle$) with an applied constant magnetic field. The spins take a long time to relax or decohere, so the qubits can be stored for a reasonable time.

We can also turn on a pulsed rotating magnetic field with frequency $\omega$ (where the $\omega$ is the energy splitting between the spin-up and spin-down states), and induce Rabi oscillations of the spin. By timing the pulse suitably, we can perform a desired unitary transformation on a single spin (just as in our discussion of the ion trap). All the spins in the molecule are exposed to the rotating magnetic field but only those on resonance respond.

Furthermore, the spins have dipole-dipole interactions, and this coupling can be exploited to perform a gate. The splitting between $|\uparrow\rangle$ and $|\downarrow\rangle$ for one spin actually depends on the state of neighboring spins. So whether a driving pulse is on resonance to tip the spin over is conditioned on the state of another spin.

All this has been known to chemists for decades. Yet it was only in the past year that Gershenfeld and Chuang, and independently Cory, Fahmy, and Havel, pointed out that NMR provides a useful implementation of quantum computation. This was not obvious for several reasons. Most importantly, NMR systems are very *hot*. The typical temperature of the spins (room temperature, say) might be of order a million times larger than the energy splitting between $|0\rangle$ and $|1\rangle$. This means that the quantum state of our computer (the spins in a single molecule) is very noisy – it is subject to strong random thermal fluctuations. This noise will disguise the quantum information. Furthermore, we actually perform our processing not on a single molecule, but on a macroscopic sample containing of order $10^{23}$ "computers," and the signal we read out of this device is actually averaged over this ensemble. But quantum algorithms are *probabilistic*, because of the randomness of quantum measurement. Hence averaging over the ensemble is not equivalent to running the computation on a single device; averaging may obscure the results.

Gershenfeld and Chuang and Cory, Fahmy, and Havel, explained how to overcome these difficulties. They described how "effective pure states" can be prepared, manipulated, and monitored by performing suitable operations on the thermal ensemble. The idea is to arrange for the fluctuating properties of the molecule to average out when the signal is detected, so that only the underlying coherent properties are measured. They also pointed out that some quantum algorithms (including Shor's factoring algorithm) can be cast in a deterministic form (so that at least a large fraction of the computers give the same answer); then averaging over many computations will not spoil the result.

Quite recently, NMR methods have been used to prepare a maximally entangled state of three qubits, which had never been achieved before.

Clearly, quantum computing hardware is in its infancy. Existing hardware will need to be scaled up by many orders of magnitude (both in the number of stored qubits, and the number of gates that can be applied) before ambitious computations can be attempted. In the case of the NMR method, there is a particularly serious limitation that arises as a matter of principle, because the ratio of the coherent signal to the background declines exponentially with the number of spins per molecule. In practice, it will be very challenging to perform an NMR quantum computation with more than of order 10 qubits.

Probably, if quantum computers are eventually to become practical devices, new ideas about how to construct quantum hardware will be needed.

## 1.10 Summary

This concludes our introductory overview to quantum computation. We have seen that three converging factors have combined to make this subject exciting.

1. **Quantum computers can solve hard problems**. It seems that a new classification of complexity has been erected, a classification better founded on the fundamental laws of physics than traditional complexity theory. (But it remains to characterize more precisely the class of problems for which quantum computers have a big advantage over classical computers.)

2. **Quantum errors can be corrected**. With suitable coding methods, we can protect a complicated quantum system from the debilitating effects of decoherence. We may never see an actual cat that is half dead and half alive, but perhaps we can prepare and preserve an *encoded cat* that is half dead and half alive.

3. **Quantum hardware can be constructed**. We are privileged to be witnessing the dawn of the age of coherent manipulation of quantum information in the laboratory.

Our aim, in this course, will be to deepen our understanding of points (1), (2), and (3).