Schrödinger's Cyber-Cat: How to Simulate Quantum Mechanics on a Computer

David Strozzi

Department of Physics,

Princeton University

Abstract

We consider three different schemes for simulating quantum mechanical systems with computers. This is a traditionally difficult problem, with all known algorithms for modern computers needing unattainable storage and processor resources for the simplest simulations. For example, simulating the motion of n particles on a d-dimensional spatial lattice with l lattice sites on a side requires $\sim l^{nd}$ storage space and processor time. Since one of the principal differences between classical and quantum physics is that the latter is probabilistic, one is tempted to design a direct simulation, or emulation, of a quantum system on a probabilistic classical computer. However, the nonlocal nature of quantum mechanics, and the impossibility of reproducing its predictions with any hidden-variables theory, makes such an emulation impossible. In the last few years, algorithms have been developed for a quantum computer, or one which as a physical device behaves quantum-mechanically, that can simulate many-particle systems with an exponential performance improvement over classical algorithms. We will study such an algorithm developed by Boghosian and Taylor that performs the same simulation outlined above using $\sim l^d$ memory and processor time for $n \ll l^d$.

1 Overview

Simulations of physics, from rocketry to plasmas to the weather, have always been an important use of computers. Unfortunately, simulating quantum mechanics has invariably been inefficient, with the needed time and memory growing exponentially with the size of the studied system. All of these algorithms have been devised for classical machines, whose physical parts are such that quantum mechanics may be neglected. The probabilistic nature of quantum mechanics suggests that a classical, probabilistic computer may be able to simulate quantum systems more efficiently. Such a machine would not return the same output every time it runs a given algorithm on the same output; instead, the distribution of output after many runs would approach that of the quantum system being simulated. This hope is shattered by the fact that quantum mechanics is not a local theory, in that no classical hidden-variable theory can reproduce its results. "Quantum computers," or devices dominated by quantum effects, offer new possibilities for efficient simulations of quantum physics. Quantum algorithms have developed which simulate quantum systems with memory and time that is exponentially better than classical computers.

In this paper, we compare the simulation of quantum mechanics on these three kinds of computers. We develop a simulation of nonrelativistic particles governed by the Schrödinger equation on a deterministic classical computer and estimate its memory needs and running time. We will then study the nonlocal nature of quantum theory via the EPR paradox and Bell's inequalities, and see why they prevent us from emulating a quantum system on a probabilistic classical computer. We then explore the general structure of a quantum computer, and closely study Taylor's exponentially-faster simulation of the motion of such particles. Our motivation here is threefold. First, an efficient qunatum simulation is an intrinsically interesting result, with significant practical ramifications if ever implemented experimentally. Because this simulation runs on a quantum computer, studying it will elucidate the basic ideas of quantum computation. In addition, trying to simulate quantum physics will deepen our understanding of quantum theory itself.

2 Simulations on a Deterministic Classical Computer

Let us now develop an algorithm for simulating quantum mechanics on a classical, deterministic, digital computer. A deterministic computer is one which, given a certain input and algorithm, will always return the same output. As a physical device, this means that specifying the current state of the computer and its input will uniquely specify how it evolves. It is possible to conceive of computers which do not appear to behave deterministically but obey classical physics on a microscopic or "hidden" level. We will clarify these concepts and consider the prospects for quantum-mechanical simulations on such a device in detail below.

I stipulate the computer is digital to distinguish it from an analog machine. For example, one could easily build an analog integrator from just a capacitor and a resistor. While analog computers have been used in the past and can be blindingly fast, their circuitry must be specifically tailored to the computation they perform. Analog computers are therefore single-purpose, as opposed to the digital computers in use today which can execute a huge variety of programs. In fact, the theory of algorithms is based upon the assumption that a kind of computer called a Turing machine can compute any algorithm we'd be interested in computing. A full discussion of Turing machines is not important here, and the interested reader should consult [1]. The relevant result from computer science is that digital computers can serve as Turing machines, and can therefore calculate a wide range of algorithms. The basic functions performed by a digital computer are logical operations acting on two bits (or binary digits), or 2-bit logic gates. Since any 2-bit logic gate can be constructed from a series of one logic gate (such as the NOT AND or NAND gate) [2], digital computers are easy to construct: simply figure out how to build this gate, and then wire a large number of them together. While it is straightforward to write a calculator or a text editor for a digital computer, implementing them on an analog computer requires developing two specific analog circuits. For these reasons, analog computers have fallen by the wayside, and only a discussion of digital computers would relate to current technology.

Another difference between analog and digital computers is that they store continuous

and discrete values, respectively. Following standard practice, we assume digital computers store information as a series of bits (ie, a string of 0's and 1's). We store each bit in a physical system that has only two accessible states, such as a switch being on or off, or the voltage on a transistor being HIGH or LOW. Since we can arrange a finite number of bits a finite number of ways, the value represented by these bits varies over a finite, discrete range. Therefore, to simulate any physical system on a digital computer, we must find a discrete, finite model for space and time so that we can store space and time coordinates. We treat space as a d-dimensional array of points spaced Δx apart with l lattice sites to a side, similar to a crystal lattice. Our time jumps discontinuously in steps of Δt and starts at t_0. We choose units where $\hbar = \Delta t = \Delta x = 1$; we can do this since we have three units to scale: mass, length, and time.

Just as we need to discretize space and time, we must also find a way to store the wavefunction Ψ of the simulated system on a digital computer. Recall that Ψ is a complex-valued function of some set of variables needed to completely specify the system's state. For a single particle, we will use its three spatial coordinates to specify its state (along with a spin wavefunction which we ignore here), so that Ψ is in effect a function of the spatial coordinates and time. However, when considering a multi-particle system, Ψ is evaluated not at every point in space but at every point in the configuration space of the whole system $\{\mathbf{r}_1, \mathbf{r}_2, ..., \mathbf{r}_n\}$, where $\mathbf{r}_i$ is the position of the i^{th} particle. One way to represent Ψ on our computer is to use the positions of the particles to specify their state, and record the value of Ψ at every point of this dn-dimensional lattice. To store the values of Ψ, we can treat the complex plane as a finite, two-dimensional lattice just as we did space, and then record a complex number as a pair of real numbers (the real and imaginary part).

Consider the simple case of a particle moving on a 1-dimensional lattice l points long. It will start at some initial location x_0 at time t_0, and then diffuse via the Schrödinger equation. A measurement of the system will find the particle at one lattice site, and all the other ones unoccupied. Ψ evaluates to a complex number at each lattice site, so we need to specify

the amplitude of Ψ at all l lattice sites. The time evolution of our nonrelativistic system is governed by the Schrödinger equation, which in 1 dimension is

$$i\frac{\partial\Psi[x,t]}{\partial t} = H[x,t]\Psi[x,t] = -\frac{1}{2m}\frac{\partial^2\Psi}{\partial x^2} + V[x]\Psi^1, \tag{1}$$

where m denotes the particle mass and H the Hamiltonian operator. Since our space and time are discrete, we can simplify the notation by writing $\Psi_{i,j} \equiv \Psi[x_0 + i, t_0 + j]$ (recall that we've set the x and t spacing to 1). Eqn. (1) involves continuous functions and their derivatives, so we must make a discrete approximation to this equation in order to simulate it on our computer. Besides evaluating Ψ and V at discretely-spaced lattice sites, we also must replace derivatives with finite fractions:

$$\frac{\partial\Psi[x,t]}{\partial t} = \lim_{\Delta t\to 0}\frac{\Psi[x,t+\Delta t] - \Psi[x,t]}{\Delta t} \to \Psi_{i,j+1} - \Psi_{i,j}. \tag{2}$$

Making a similar substitution for $\frac{\partial\Psi}{\partial x}$, we can write a discrete version of (1):

$$\Psi_{i,j+1} - \Psi_{i,j} = \frac{-1}{2m}\frac{\partial}{\partial x}(\Psi_{i+1,j} - \Psi_{i,j}) + V[x]\Psi_{i,j} = \frac{-1}{2m}(\Psi_{i+2,j} - 2\Psi_{i+1,j} + \Psi_{i,j}) + V[x]\Psi_{i,j}. \tag{3}$$

Solving for $\Psi_{i,j+1}$, we have

$$\Psi_{i,j+1} = \Psi_{i,j} - \frac{-1}{2m}(\Psi_{i+2,j} - 2\Psi_{i+1,j} + \Psi_{i,j}) + V[x]\Psi_{i,j}. \tag{4}$$

This formula gives us an algorithm to calculate how Ψ evolves over time on a digital, classical, deterministic computer, which is precisely a simulation. What is the running time for such an algorithm? To find $\Psi_{i,j+1}$, we must apply eqn. (4) for each of the l lattice points of our 1-dimensional space. Since the number operations τ needed to evaluate eqn. (4) does not depend on Ψ or l, the running time is τl. It is common practice in computer science to neglect constant factors and only keep the fastest-growing term when estimating

[1] Here, as in Mathematica, [...] denotes the argument of a function, (...) denotes grouping.

an algorithm's running time. Doing that here, we approximate the running time of our simulation as l.

We can now generalize our analysis to d-dimensional space and n particles. The only difference in d dimensions is that the number of lattice sites is l^d instead of l: we can think of the space as a direct product of d linear lattices. Since we need to store the value of Ψ at each point on the lattice, we need l^d values. The wavefunction of a system of n *distinguishable* particles is a linear combination of tensor products of single-particle basis states: $\Psi = \sum c_{i_1 i_2 \ldots i_n} \psi_{i_1} \otimes \psi_{i_2} \otimes \ldots \otimes \psi_{i_n}$. Since there are l^d possible single-particle states, we must keep track of $l^d \cdot l^d \cdot \ldots \cdot l^d = l^{nd}$ amplitudes c. To advance Ψ in time, we need to apply eqn. (4) as before. If the particles are identical, we must either symmetrize or antisymmetrize Ψ, depending on whether we have boson or fermions. This constraint will reduce the number of independent basis states for Ψ, and we will only need to store $\sim \frac{1}{n!} l^{nd}$ coefficients. This will not stop the exponential explosion of the memory needed, so we will assume the particles are distinguishable for simplicity.

Unfortunately, the amount of storage space and running time needed for such a simulation is ridiculous. If we have 20 particles moving on a 3-dimensional lattice with 10 sites on each side, this requires $l^{dn} = 10^{60}$ complex numbers. If we allocate 32 bits (4 bytes) for each real number (as many compilers do by default), we would need $2 \cdot 32 \cdot 10^{60} = 64 \cdot 10^{60}$ bits to store Ψ (for indistinguishable particles this becomes $\sim 10^{42}$, which is still huge). Considering a terabyte holds $(2^{10})^4 \cdot 8 \sim 10^{13}$ bits and that the world's largest databases are on the order of terabytes, building a machine with enough memory for our simulation is unthinkable. Since the running time is proportional to the number of amplitude coefficients we need, we must perform $\sim 10^{60}$ operations to advance the simulation one time step. The fastest computers available today perform $\sim 10^{12}$ instructions per second, again making our simulation impossible.

3 Simulations on a Probabilistic Classical Computer

Our attempt to simulate quantum mechanics with reasonable computing resources failed miserably. However, some very complex classical systems have been successfully simulated on computers, such as the solar system. What differences between classical and quantum mechanics make it so hard to simulate the second but easy to simulate the first? Probably the most-emphasized difference between the two is the probabilistic nature of quantum mechanics. Quantum mechanics only allows us to predict the probability we will observe a system in a given state; on the other hand, classical physics tells us precisely how a system will evolve given its initial state. This suggests we consider using a computer that itself behaves probabilitistically to simulate quantum systems. A "probabilistic" computer is one which does not always return the same output when running the same algorithm on the same input. Such a machine appears in computer science as the notion of the nondeterministic Turing machine [1]. In this section, we will study the prospect of efficiently simulating quantum mechanics on a nondeterministic Turing machine, or a classical probabilistic computer.

If our computer itself behaves "classically," how can it be probabilistic - won't the computer, as a physical system, evolve deterministically from its initial to final state? One way to build such a machine is to use a statistical or chaotic system. For instance, consider a box with a mole of gas molecules in it. Classical mechanics holds that this system is deterministic on the level of molecular motion. However, it is impossible for us to analyze a system of $\sim 10^{23}$ particles by keeping track of their individual trajectories. We instead look at macroscopic properties such as temperature and pressure. We can use this apparatus to perform probabilistic computation, say by counting the number of molecules which strike a small region of the container wall in a fixed time. What separates such a probabilistic system from a quantum system? As we shall see below, the difference is that quantum systems are nonlocal, while classical systems are local. While there are variables which cause a probabilistic classical system to evolve deterministically, they remain "hidden" from us, causing us to see the system as probabilisitic. No such hidden variables can underlie quantum mechanics, so

a probabilistic and local computer cannot accurately simulate quantum systems [3].

Since the resources necessary to store all the independent components of the wavefunction are unattainable, we will instead try to construct a computer which when observed yields the same probability distribution as the quantum system being studied. In effect, we are looking for a probabilistic classical system which can replicate all the behaviors of a quantum one. To do this, we will interpret our computer being in a certain physical state to represent the quantum system being in a certain state Ψ. We will then run our computer many times, and know to within a certain statistical accuracy the probability that the quantum system evolves to a certain final state. In general, the computer will start in some initial state Ψ_0 (since the state of the computer corresponds to the simulated system being in some state Ψ, we will for simplicity describe the state of the computer by the Ψ it represents). There will be some set of states to which the computer can evolve, with each transition occurring with some probability. In the next cycle, there will be some new set of target states, with the probabilities of transition depending on the current state Ψ_1, and so on.

We can formalize this evolution as follows. Suppose we simulate a single particle moving on a discrete line of l lattice sites. Our computer could consist of a set of two-state physical systems, with each one corresponding to the presence or absence of a particle at a specified lattice site. The state of the particle is completely specified by the state of the system at each lattice point. Using s_i to denote the state of the i^{th} lattice site and P for probability, we have

$$P[\Psi_{j+1} = \{s'_1 ... s'_l\}] = \sum_{\Psi_j = \{s_1 ... s_l\}} (P[\Psi_j = \{s_1 ... s_l\}] P[\{s_1 ... s_l\} \rightarrow \{s'_1 ... s'_l\}]) \,. \tag{5}$$

Since $P[s_1 ... s_l \rightarrow s'_1 ... s'_l] = P[s_1 \rightarrow s'_1 | \Psi_j] * ... * P[s'_l \rightarrow s'_l | \Psi_j] = \prod_{k=1}^{l} P[i_k \rightarrow s'_k | \Psi_j]$, where $|\Psi_j$ denotes given a certain Ψ_j,

$$P[\Psi_{j+1} = \{s'_1 ... s'_l\}] = \sum_{\Psi_j} \left(P[\Psi_j] \prod_{k=1}^{l} P[s_k \rightarrow s'_k | \Psi_j] \right) . \tag{6}$$

This approach accurately represents what physically happens in a quantum system: given an initial state, the system will evolve with some probability to one of a set of final states. However, when we specify that our probabilistic computer is classical, and therefore local, this simulation cannot reproduce quantum effects.

We will see why this is by discussing the Einstein-Podolsky-Rosen (hereafter EPR) paradox and Bell's inequalities, which clearly demonstrate the nonlocality of quantum physics. The EPR Paradox was proposed in 1935 to show why quantum mechanics was unsatisfactory. EPR assert that any acceptable physical theory should be "complete," meaning any prediction that can be made with unit probability must correspond to an element of physical reality [4]. For example, classical mechanics dictates that momentum is conserved. Accordingly, if an initially at-rest rocket explodes into two pieces and I observe half of it with momentum $\mathbf{p}$, I know with certainty that the other half has momentum $-\mathbf{p}$. The completeness of classical mechanics would imply that there exist a physically real entity corresponding to this definite prediction (here, it would be the half of the rocket with momentum $-\mathbf{p}$). The other property EPR demand of any physical theory is locality, or that no effects propagate instantly (ie, there is no action at a distance).

We illustrate the EPR paradox by studying a system with no total spin which decays into two spin-1/2 particles, as suggested by D. Bohm. Suppose the particles, labeled 1 and 2, travel in opposite directions away from the common source. We know that the total spin of the system is 0, so measuring $S_{z,1}$ (the z-spin of 1) immediately tells us $S_{z,2}$ (it will be $-S_{z,1}$). Since we can predict this fact about 2 with unit probability, completeness implies a real entity exists which corresponds to 2 having this z-spin. We know this certain fact about 2 instantly after making our observation, so the corresponding real entity must exist immediately after we measure 1. However, if we wait for 1 and 2 to be far apart before measuring, and if this real entity came into existence exactly when we make our measurement, we would have an instantaneous effect. Therefore, the real entity must exist *before* we make our measurement. Note that regardless of the value we get for $S_{z,1}$, $S_{z,2}$ will have a definite value, so some

corresponding real entity must exist before we measure 1.

The problem occurs if we were to measure $S_{x,1}$ instead of $S_{z,1}$. Following the same logic as above, this would tell us exactly the value of $S_{x,2}$. Therefore, some real entity exists before we measure $S_{x,1}$ which corresponds to $S_{x,2}$ having a definite value. However, $S_{z,1}$ and $S_{x,1}$ do not commute, so if we measure $S_{z,1}$, we induce an uncertainty in $S_{x,1}$ and also in $S_{x,2}$. But, we just established that there must be a real entity corresponding to $S_{x,2}$ having a definite value *before we measure 1*! If such a real entity exists before we make our measurement, and if we perform the same experiment many times, how could the observer at 2 get different values of $S_{x,2}$? Quantum mechanics must therefore be incomplete or contain actions at a distance.

The so-called EPR Paradox has sparked a 50-year debate about the nature and validity of quantum mechanics. Physicists who agree with EPR have searched for a theory which reproduces the statistical predictions of quantum mechanics, but is still local and complete. Such theories have been dubbed "hidden-variable" theories, since they rely on some other physical quantities not yet observed determining the value of the observed quantum variables. In other words, there would exist some classical variable λ which completely determines the values of observables such as spin (having a set of such λ's would not change the spirit of our analysis). Not knowing about this variable, we would naïvely think that spin and other quantum observables behave probabilistically. However, a result known as Bell's inequalities constrains the statistical predictions of any hidden-variables theory in ways that quantum theory violates. With the experimental verification of these forbidden predictions, physicists have by and large rejected the hope of finding a hidden-variable quantum theory.

Suppose there are two observers who will measure the spin of 1 along some direction $\hat{\mathbf{a}}$ and of 2 along $\hat{\mathbf{b}}$, both in the plane perpendicular to the motion of the particles (see figure 1). Denote the results of our measurements by A and B respectively, both of which can only take on values of ± 1 (when we measure spin along an axis, we get up or down. We also neglect the factor of $\hbar/2$ that would normally be here). If our hidden-variable theory is

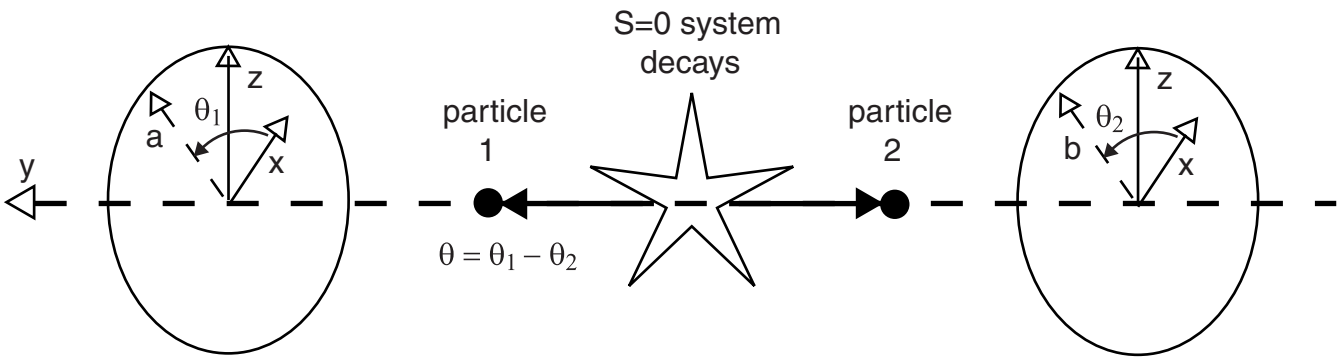

Figure 1: Diagram of EPR experiment

complete, the value of λ should be sufficient to determine the results of our measurement. If we stipulate that each observer chooses the direction along which to measure instantly before measuring, then the local nature of our theory requires that A and B not be functions of $\hat{\mathbf{b}}$ and $\hat{\mathbf{a}}$, respectively (that is, the direction which the *other* observer chooses). This reasoning implies that $A = A[\hat{\mathbf{a}}, \lambda]$ and $B = B[\hat{\mathbf{b}}, \lambda]$. Also, the fact that the total spin of the system is zero along any direction implies $A[\hat{\mathbf{a}}, \lambda] = -B[\hat{\mathbf{a}}, \lambda]$. Let us further assume that the distribution of λ is normalized to unity: $\int p[\lambda] d\lambda = 1$. If we measure A and B many times for fixed values of $\hat{\mathbf{a}}$ and $\hat{\mathbf{b}}$, the average value is

$$C[\hat{\mathbf{a}}, \hat{\mathbf{b}}] = \int p[\lambda] A[\hat{\mathbf{a}}, \lambda] B[\hat{\mathbf{b}}, \lambda] d\lambda. \tag{7}$$

We use C because it reflects the correlation between the two measurements: C is 1 if the measurements agree (both up or both down) and -1 if they do not.

The crucial step in establishing Bell's inequalities is to consider the classical correlation difference $\Delta_C \equiv C[\hat{\mathbf{a}}, \hat{\mathbf{b}}] - C[\hat{\mathbf{a}}, \hat{\mathbf{c}}]$, where $\hat{\mathbf{b}} \neq \hat{\mathbf{c}}$. From eqn. (7), we have

$$\begin{aligned} \Delta_C &= \int p[\lambda] (A[\hat{\mathbf{a}}, \lambda] B[\hat{\mathbf{b}}, \lambda] - A[\hat{\mathbf{a}}, \lambda] B[\hat{\mathbf{c}}, \lambda]) d\lambda \\ &= -\int p[\lambda] A[\hat{\mathbf{a}}, \lambda] A[\hat{\mathbf{b}}, \lambda] (1 + A[\hat{\mathbf{b}}, \lambda] B[\hat{\mathbf{c}}, \lambda]) d\lambda. \end{aligned} \tag{8}$$

We arrived at the last line by using $(A[\hat{\mathbf{b}}, \lambda])^2 = 1$ for any direction $\hat{\mathbf{b}}$. Recall from analysis

that $|\int f[x]dx| \leq \int |f[x]|dx$. Applying this to eqn. (8),

$$\begin{aligned} |\Delta_C| &\leq \int |p[\lambda]| \left|A[\hat{\mathbf{a}},\lambda]B[\hat{\mathbf{b}},\lambda]\right| \left|(1+A[\hat{\mathbf{b}},\lambda]B[\hat{\mathbf{c}},\lambda])\right| d\lambda \\ &= \int p[\lambda](1+A[\hat{\mathbf{b}},\lambda]B[\hat{\mathbf{c}},\lambda])d\lambda \\ &= \int p[\lambda]d\lambda + \int p[\lambda]A[\hat{\mathbf{b}},\lambda]B[\hat{\mathbf{c}},\lambda])d\lambda = 1 + C[\hat{\mathbf{b}},\hat{\mathbf{c}}] \\ \Rightarrow \left|\Delta_C[\hat{\mathbf{a}},\hat{\mathbf{b}},\hat{\mathbf{c}}]\right| &\leq 1 + C[\hat{\mathbf{b}},\hat{\mathbf{c}}]. \end{aligned} \tag{9}$$

We have used the fact that $p[\lambda]$ is normalized to unity, that A only takes on values of ± 1, and that AB is no less than -1 (so that 1+AB is nonnegative). Eqn. (9) is one of a family of such relations known as Bell's inequalities, first derived by J. S. Bell in 1964 [5].

We will now analyze the same experiment from the quantum standpoint, and see that quantum mechanics predicts violations of eqn. (9). Our system consists of 2 identical spin-1/2 fermions with a total spin of 0, and must therefore be in the singlet state. Choosing α (β) to denote the z spin-up (-down) eigenstate and $\hat{\mathbf{z}} = \hat{\mathbf{a}}$, the system's overall wavefunction $\chi = \frac{1}{\sqrt{2}}(\alpha_1\beta_2 - \beta_1\alpha_2)$. Let $S_{a,1}$ denote the spin of particle 1 along the direction $\hat{\mathbf{a}}$. We know that $S_{a,1} = \mathbf{S}\cdot\hat{\mathbf{a}} = \boldsymbol{\sigma}[1]\cdot\hat{\mathbf{a}}$, where $\boldsymbol{\sigma}$ contains the three Pauli spin matrices and we again neglect factors of $\hbar/2$. We wish to make a joint measurement of the spin of 1 along $\hat{\mathbf{a}}$ and 2 along $\hat{\mathbf{b}}$. As before, we consider an observable called the correlation, which returns 1 if the two measurements agree and -1 if they do not. The operator for correlation is then $\boldsymbol{\sigma}[1]\cdot\hat{\mathbf{a}}\boldsymbol{\sigma}[2]\cdot\hat{\mathbf{b}}$. Defining $\mathcal{C}$ to be the expectation value of such a joint measurement, we have

$$\begin{aligned} \mathcal{C}[\hat{\mathbf{a}},\hat{\mathbf{b}}] &= \langle\chi\left|\boldsymbol{\sigma}[1]\cdot\hat{\mathbf{a}}\boldsymbol{\sigma}[2]\cdot\hat{\mathbf{b}}\right|\chi\rangle \\ &= \tfrac{1}{2}\langle\alpha_1\beta_2 - \alpha_2\beta_1\left|\sigma_{z,1}\left(\sigma_{z,2}\cos\theta + \sigma_{x,2}\sin\theta\right)\right|\alpha_1\beta_2 - \alpha_2\beta_1\rangle \\ \mathcal{C}[\hat{\mathbf{a}},\hat{\mathbf{b}}] &= -\cos\theta. \end{aligned} \tag{10}$$

to arrive at eqn. (10), note that $\mathcal{C}$ has 4 terms involving $\sigma_{z,1}\sigma_{z,2}$, which together yield $-\cos\theta$. Since σ_x acting on a spin-z eigenstate returns ± 1 on a 50-50 basis, the 4 terms involving $\sigma_{z,1}\sigma_{x,2}$ vanish. To compare this with Bell's inequality, we calculate $\Delta_Q[\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}] \equiv \mathcal{C}[\hat{\mathbf{a}}, \hat{\mathbf{b}}] - \mathcal{C}[\hat{\mathbf{a}}, \hat{\mathbf{c}}] = -\cos\theta_{ab} + \cos\theta_{ac}$. Choose $\theta_{ab} = \pi/3$ and $\theta_{ac} = 2\pi/3$. Thus, $\Delta_Q = -\cos[\pi/3] + \cos[2\pi/3] = -1$, and $1 + \mathcal{C}[\hat{\mathbf{b}}, \hat{\mathbf{c}}] = 1 - \cos[\pi/3] = 1/2$. For these values of θ_{ab} and θ_{bc},

$$\left|\Delta_Q[\hat{\mathbf{a}}, \hat{\mathbf{b}}, \hat{\mathbf{c}}]\right| = 1 > 1 + \mathcal{C}[\hat{\mathbf{b}}, \hat{\mathbf{c}}] = \tfrac{1}{2}, \tag{11}$$

in violation of Bell's inequality.

Eqn. (11) has profound implications for quantum theory. The only assumptions we made in deriving Bell's inequality (eqn. (9)) was that there exists a classical analog of the wavefunction, λ, which completely describes the system, and that the system's evolution was local. This allowed us to assume that A did not depend on $\hat{\mathbf{b}}$, and write $A = A[\lambda, \hat{\mathbf{a}}]$. If quantum mechanics violates Bell's inequality, then it violates the assumption of locality. The beauty of Bell's inequality resides in its production of a parameter Δ whose measurement would either invalidate quantum mechanics, or rule out *all possible hidden-variable theories.* Extensive experiments performed by A. Aspect in the 1980s, using photons and polarizers instead of spin-1/2 particles, have vindicated quantum theory [6]. Nonetheless, the idea of a nonlocal universe is very counterintuitive and even disturbing. Heisenberg asserted that a nonlocal theory is unsatisfactory if it allows us to transmit a signal, or communicate, instantaneously [7]. However, when we observe the spin of 1, we collapse the entire system's wavefunction instantly, but we cannot control what value we measure, and therefore what value $S_{z,2}$ assumes. Quantum mechanics therefore does not allow faster-than-light communication, which led Heisenberg to conclude it is consistent with special relativity.

By showing that no hidden-variable theory can reproduce the results of quantum mechanics, we have also destroyed our hope of constructing a probabilistic, local computer that can simulate quantum mechanics! When we say a classical system is probabilistic, this means that deterministic laws govern its evolution; we just aren't keeping track of them.

Thus, when we reset our gaseous computer with the same macroscopic parameters, we still measure different numbers of collisions during each run. We believe that there are hidden variables, namely the precise microscopic positions and momenta of the gas molecules, that cause the system's evolution to actually be local and causal. Bell's inequality shows that no such hidden-variable theory can be at work underneath quantum mechanics. If we could emulate a quantum system with a probabilistic classical computer, we would in effect be basing quantum mechanics on a hidden-variable theory. Our proposed emulation would not be inefficient; it would be wrong.

4 Simulations on a Quantum Computer

So far, we have tried to use both deterministic and nondeterministic classical computers to simulate quantum mechanics. We will now consider computers whose behavior is dominated by quantum mechanics. As we showed above, quantum systems are not governed by any local theory, so the difficulty encountered with trying to use a probabilistic classical computer disappears. We will present an algorithm which simulates the motion of n nonrelativistic particles on a lattice in a time proportional to l^d, as long as $n \ll l^d$. This represents an exponential improvement over our simulation on a deterministic classical computer, which requires storage space and running time of $\sim l^{nd}$.

The machine we will study is a quantum version of the digital computer, or Turing machine. The data in such a quantum computer is stored in discrete units called quantum bits, or "qubits." The number of qubits we can store in a quantum system, q, is determined by $d_H = 2^q$, where d_H=dimension of the system's Hilbert space. We usually consider a set of systems which can be in only two possible states, labelled $|0\rangle$ and $|1\rangle$. A common illustration of this is an array of spin-1/2 particles. Since each particle can be in either the spin-up or spin-down state, a system of m such particles requires a 2^m-dimensional Hilbert space, so the system can store m qubits.

We know how to store data, but how can we manipulate it? In analogy with digital

computers, we will operate on data with logic gates, and build all our algorithms from them. For simplicity, we will only consider quantum logic gates that act on two qubits and return a third. Acting on data with a logic gate involves performing a certain physical process on the data. In this light, computers are nothing more than devices for doing physics experiments - we humans interpret the initial conditions, the experiment being done, and the final state to represent certain information and algorithms. On a quantum computer, a logic gate manifests itself as the evolution of a quantum system subjected to a certain Hamiltonian. As a quantum systems, our quantum computer evolves according to the Schrödinger equation:

$$i\frac{\Psi[x,t]}{\partial t} = H\psi[x,t] = \frac{-1}{2m}\nabla^2\Psi + V[x]\Psi. \tag{12}$$

Since this equation involves the first time derivative of Ψ, we can use it to determine Ψ at any time t if we know Ψ at some prior time t_0. We can then construct an evolution operator U such that $\Psi[t] = U[t_0, t]\Psi[t_0]$. Any elementary quantum mechanics text, such as [8], shows U is related to H by

$$U[t_0, t] = \exp[-iH(t - t_0)]. \tag{13}$$

Since H is a Hermitian operator, and the exponential of any Hermitian operator multiplied by i is a unitary operator, the evolution operator U is unitary. All unitary operators are invertible, implying that the evolution of a quantum system must be reversible. In particular, quantum logic gates must be reversible processes. There exists a set of reversible gates, NOT (1-bit), CONTROLLED NOT (2-bit), and CONTROLLED CONTROLLED NOT (3-bit), which can generate all 2-bit logic gates [2]. This makes quantum computers in principle quite powerful, since knowing how to build these three gates and store qubits is enough to build a computer that can perform a huge class of algorithms. See [9] for a thorough overview of quantum computing.

We now turn to the specific problem of simulating a quantum system on a quantum computer. We start with the simple case of one free particle moving on a lattice l sites

long. We will write the particle's wavefunction over the basis states $|1\rangle, |2\rangle, ..., |l\rangle$, where $|i\rangle$ denotes the state where the particle is located at the i^{th} lattice site. We will represent this on our quantum computer by an array of l qubits. When we observe a qubit, we will find it in either the state $|0\rangle$ or $|1\rangle$, and we will take this to mean that site is either unoccupied or occupied. If we assume that at most one particle can occupy a point in space, then such a scheme of l qubits forms a basis for a system of l particles moving on the lattice. Although we initially restrict our attention to the 1-particle subspace, the ability to simulate a multi-particle system with no extra memory will yield an exponential improvement over the classical simulation described in section 2.

We can write the single-particle wavefunction $\psi[t]$ as

$$\psi[t] = \sum_{i=1}^{l} c_i |i\rangle. \tag{14}$$

We must now find an evolution operator D such that

$$\psi[t+1] = D\psi[t] \tag{15}$$

and D reduces to the U given by $\exp[iH(t-t_0)]$ in the limit of continuous space and time. We will also require that D is unitary and can be implemented by a series of two-qubit logic gates. We present an operator found by Boghosian and Taylor, which acts on the Hilbert space for all the qubits but reduces to the Schrödinger equation on the single-particle subspace [10]. We define two operators on the single-particle subspace

$$D_1 = \begin{pmatrix} b & a & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ a & b & 0 & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & b & a & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & a & b & \cdots & 0 & 0 & 0 & 0 \\ \vdots & & & \ddots & & & & \vdots & \\ 0 & 0 & 0 & 0 & \cdots & b & a & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & a & b & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & b & a \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & a & b \end{pmatrix} \qquad D_2 = \begin{pmatrix} b & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & a \\ 0 & b & a & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & a & b & 0 & \cdots & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & b & \cdots & 0 & 0 & 0 & 0 \\ \vdots & & & \ddots & & & & \vdots & \\ 0 & 0 & 0 & 0 & \cdots & b & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & b & a & 0 \\ 0 & 0 & 0 & 0 & \cdots & 0 & a & b & 0 \\ a & 0 & 0 & 0 & \cdots & 0 & 0 & 0 & b \end{pmatrix} \tag{16}$$

where $|a|^2 + |b|^2 = 1$ and $ab^* + a^*b = 0$ so that both operators are unitary (* denotes complex conjugate). We can write an algorithm with these two unitary operators that yields the motion of a single particle:

$$\psi[t+2] = D_1 D_2 \psi[t]. \tag{17}$$

The crucial fact about these operators is that they can be constructed from an operator s which acts only on 2 qubits:

$$s = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & b & a & 0 \\ 0 & a & b & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \tag{18}$$

The authors of [10] show that the operator $D_1 \cdot D_2$ does indeed reduce to the Schrödinger equation, with the potential $V[x] = 0$. We now have a two-qubit operator that simulates the motion of a single particle on a linear lattice. It requires l qubits (one at each lattice site), and offers no improvement over the classical algorithm of section 1.

The real power of the quantum computer resides in its ability to simulate many identical particles moving on the lattice without storing any more qubits. To simulate the motion of a single particle, we only studied the single-particle subspace of the larger, l-particle Hilbert space spanned by the l qubits. For a system of n particles, the wavefunction $\Psi[x_1, x_2, ..., x_n] = \sum c_{i_1 \ldots i_n} \psi_{i_1} \otimes \psi_{i_2} \otimes \ldots \otimes \psi_{i_n}$, a linear combinations of all states with n lattice sites occupied. The only thing preventing a direct generalization to n particles of our earlier method is that the operator s fails to include particle interactions. Suppose the potential in the n-

particle problem can be expressed in terms of pairwise functions of the distance between two particles. We can implement this potential as an operation which at every time step acts on each 2-qubit pair. This will not increase the storage space at all, and will only add a constant number of instructions to each time step. The memory and running time will both grow like l, instead of growing like l^n in the classical case (d=1 for now).

Although our approach encounters some problems when we generalize to d dimensions, we can modify it so that the memory and time requirements grow only as l for $n \ll l^d$. The astute reader will have noticed that the endpoints of the linear lattice evolve differently under the operators D_1 and D_2 discussed above, and in general that the 2^d corners of space will behave this way as well. Moreover, if only one particle can occupy a lattice site at a time, we cannot model collisions in a way that is symmetric in all directions. The authors of [10] propose using a Quantum Lattice-Gas Automaton to solve these difficulties. The basic idea is to allow one particle at each lattice site for each direction of motion on a lattice; in 2 dimensions, for instance, there are 4 possible directions to move, and in d dimensions there are $2d$. We can do this if the lattice is sparsely populated (ie, $n \ll l^d$), since the likelihood of two particles occupying the same lattice site with the same velocity will then be small. We will now in effect have $2dl^d$ lattice sites, and therefore have a $2dl^d$-dimensional Hilbert space. The time required to advance the simulation on step in time will scale like d^2l^d, since we must consider collisions between all $2d$ possible particles at a lattice site with particles in neighboring sites. We see that these resource estimates are independent of n, and are exponentially smaller than the l^{nd} needed on a digital computer. For the same numbers used above, $(l, n, d) = (10, 20, 3)$, we will need only $6 \cdot 10^3 = 6000$ qubits of storage space and ~ 9000 instructions, as opposed to the 10^{60} needed by a classical computer.

We have a theoretical algorithm for simulating quantum systems with manageable resources, but it is still unclear how exactly we can extract results from our quantum computer. The simulated system's wavefunction is stored in the computer's qubits, which we must somehow measure. Unfortunately, measuring any part of the computer will collapse it into an

eigenstate, and disturb the information contained in the other qubits! The only way to proceed is to run the algorithm many times and record how frequently we find the machine in a given state. This will lets us know, to some statistical certainty, the magnitude of the coefficients of the wavefunction, or the probability of finding the system in the corresponding eigenstate. This is the best we can do in a quantum-mechanical experiment anyway, since the theory is inherently probabilistic and the phases of the amplitude coefficients do not influence the distribution of the results of measurement (that is, they are not directly measurable). It is also important to realize that the idea we use here to get answers from a quantum computer matches the analysis behind the classical probabilistic computer in section 3. The only difference between our quantum algorithm and the classical-probabilistic approach is that the physics underlying the latter kind of computer prevented it from reproducing the statistical predictions of quantum mechanics.

5 Conclusions and Future Prospects

The quantum algorithm described above for simulating quantum mechanics offers an exponential speedup over the classical approach presented in section 2. Another exciting result is the algorithm developed by Shor for factoring large numbers in polynomial time: all known classical algorithms are slow, exponential-time ones [11]. These developments show that quantum computers are not just a theoretical curiosity, but hold concrete, practical advantages over classical computers. This begs the question of how much progress has been made toward constructing a real quantum computing device. Unfortunately, only the most basic systems capable of storing a few qubits have been realized in the laboratory. Efforts are focused right now on using ion traps or nuclear magnetic resonance systems to construct basic quantum computers with 10 to 40 qubits of memory.

One of the main pitfalls in constructing a working quantum computer is error. All computers must be robust against the occasional flipping of a bit, and the theory of error-correcting codes for digital computers is well-established. The sources of error, however,

are much more insidious in the quantum realm. The wavefunction of the computer cannot be separated from that of its environment the way we can shield a classical system from contact with its environment. This coupling to the environment leads to all sorts of problems, not the least of which is decoherence, or the disturbance of the coherent superposition needed to perform computation. To show how exacerbating the problem of decoherence is, merely observing the computer may destroy a computation since it collapses the computer's wavefunction!

From a more theoretical perspective, studying computation has helped illuminate aspects of physical theory itself. For instance, Feynman suggested that since computers as we conceive them can only work with discrete quantities, space and time may not be continuous but discrete. These and other connections between physics and computation are profound, but we are only starting to explore them. One of the main bridges between these fields is the notion of information, which as we mentioned above underlies Heisenberg's acceptance of quantum nonlocality (it cannot be used to transmit information). Perhaps large parts of physical theory can be translated from statements about momentum conservation or symmetry into statements about information. It seems clear that the simulation of nature by computer, digital, quantum, or otherwise, still has many diverse challenges and rewards to offer us.

References

[1] M. Sipser. *Introduction to the Theory of Computation.* PWS Publishing Company, preliminary edition, 1996.

[2] R. P. Feynman. Quantum mechanical computers. *Foundations of Physics*, 16(6), 1986.

[3] R. P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7), 1982.

[4] A. Einstein, B. Podolsky, and N. Rosen. *Physical Review*, 47, 1935.

[5] J. S. Bell. *Physics*, 1, 1964.

[6] A. Aspect, P. Grangier, and G. Roger. *Physical Review Letters*, 28, 1972.

[7] P. Eberhard. The epr paradox: Roots and ramifications. In W. Schommers, editor, *Quantum Theory and Pictures of Reality.* Springer Verlag, 1989.

[8] B. H. Bransden and C. J. Joachain. *Introduction to Quantum Mechanics.* Longman Scientific & Technical, 1989.

[9] A. Steane. Quantum computing, July 1997.

[10] *Simulating Quantum Mechanics on a Quantum Computer*, Nov. 1996. Based on talk given at PhysComp '96 conference, Boston University.

[11] P. W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, Aug. 1995. AT&T preprint quantum-ph/9508027.

This paper represents my own work, written in accordance with University regulations.