Chapter 4

Quantum computation

In chapter 2 we saw various ways in which fundamental physics limits what we can do with computers. But even within these limits, physics may also afford new, previously unforeseen opportunities.

In this chapter we review the field of *quantum computation*, which studies a particular new kind of computation that takes advantage of known physical principles in a new way which appears to be fundamentally more powerful than the kinds of computation employed in any existing computers. Quantum computing may or may not turn out to be practical in the long run, but its eventual feasibility has not yet been conclusively ruled out. Therefore it is important for us, when considering the ultimate limits of computing, to at least be aware of the ways in which quantum computing may potentially belie the conventional wisdom about what is possible.

Moreover, the basic operations used in quantum computation are inherently reversible, and thus constitute a potential realm of application for some of the reversible algorithm techniques discussed later in this thesis.

Quantitative Church's Thesis. The "Quantitative Church's Thesis" [176, 175] claims that Turing machines are as efficient as any realistic computer, within a polynomial factor. However, Feynman [59] has pointed out that Turing machines seem to be unable to efficiently simulate quantum physics; that is, they seem to require an exponential slowdown to simulate it (although this has not been proven). This leads naturally to the supposition that a computer that was designed to take full advantage of quantum physical principles might be found to be exponentially faster than a Turing machine, at least for some problems, thus disproving the Quantitative Church's Thesis. Such a development could lead to eventual practical applications, if and when such *quantum computers* become buildable.

Shor's Factoring Algorithm. However, this idea remained pure speculation until the last several years, when a series of papers on the power of quantum computers

[47, 23, 22, 21, 149] culminated in Peter Shor's 1994 proof [146, 147] that a (somewhat idealized) quantum computer could factor large integers in polynomial time in the number of bits n in the integer. This was an astounding discovery, since mathematicians throughout history have searched for an efficient way to factor numbers without success, since at least the time of Euclid. The best known classical algorithm [102] takes exponential time.[1]

4.1 Some fundamental quantum concepts

Hilbert spaces. The most important thing to understand about the difference between the quantum and classical paradigms of physics is that in the quantum paradigm, the classical notion of the state of a system is not sufficient to describe the dynamical evolution of the system. In fact, if $\mathcal{S}$ is the space of possible states of a classical system, then a corresponding quantum description of the system involves a consideration of the much larger space of *functions* from $\mathcal{S}$ to the complex numbers $\mathbb{C}$, *i.e.* the space $\mathcal{H} = \mathbb{C}^{\mathcal{S}}$, to use the set theory notation. The number of dimensions of $\mathcal{H}$ is equal to the number of *elements* of $\mathcal{S}$. This space $\mathcal{H}$ is called the *Hilbert space* of the system.

Amplitudes and probabilities. So again, the quantum state of a system consists of a function $\Psi : \mathcal{S} \to \mathbb{C}$ from the classical states of the system to the complex numbers. These complex numbers are called *amplitudes* and their physical significance is that the square of the absolute value of the amplitude of a classical state x is interpreted as the probability that the system would be found to be in state x if the system were observed, *i.e.* $\Pr(x|\Psi) = |\Psi(x)|^2 = \Re(\Psi(x))^2 + \Im(\Psi(x))^2$, where $\Re(z)$ and $\Im(z)$ represent the real and imaginary parts of the complex number z, respectively. A simple probability distribution over classical states would suffice instead, if a static description of the system were all that was required. But if the dynamical evolution of the system is to be modeled in a quantum-theoretic way, the full power of a complex function is necessary; the probabilities alone do not suffice to describe how a quantum system can change over time. The function from states to amplitudes is commonly referred to as a *superposition of states*.

Unitary transformations. The fundamental principle of change in quantum systems is the *unitary transformation*. If we imagine the amplitude function Ψ being identified with a vector of complex numbers indexed by the states in $\mathcal{S}$, then a unitary transformation is simply a multiplication of those vectors by a transformation matrix whose inverse equals its conjugate transpose. (To briefly review some definitions from linear algebra and complex arithmetic, the *transpose* M^{T} of a matrix M is defined

[1] More precisely, $O(\exp(n^{1/3}\log(n^{2/3})))$.

by $M^{\mathrm{T}}(x, y) = M(y, x)$ *i.e.* the matrix is flipped around its diagonal axis, and the *conjugate* M^* of M is defined by $M^*(x, y) = \Re(M(x, y)) - \Im(M(x, y))i$, that is, the sign of the imaginary part of each element is negated. The inverse M^{-1} of M is the matrix such that $M^{-1}M = I$, the identity matrix.)

A unitary transformation, intuitively speaking, corresponds to a length-preserving and information-preserving rotation in the vector space. The length-preserving requirement ensures that the total probability of the set of states always remains the same (1 if normalized), and the information-preserving requirement reflects what seems to be a universal conservation principle in fundamental physics (which we discussed in §2.5), namely that all changes are information-preserving on a microscopic level; or, in quantum-mechanical terms, the complete quantum state (amplitude vector) of an isolated system at any time determines the quantum state of the system at all past and future times.

Measurement phenomena. We have already mentioned that a measurement of the state of a quantum system (or of part of the state) yields a result with a probability equal to the square of that state's amplitude. (Or, if only part of the state is measured, the probability is equal to the sum of the squares of the amplitudes of all the global states that are consistent with the state of the part being measured.) However, such a measurement has an apparent side-effect on the system, namely that the probabilities of all states inconsistent with the observed result drop to zero, and the amplitudes of all the states that are consistent with the observed result are scaled up accordingly, so that the total probability over all states remains 1.

In other words, after observing a measurement (that is, allowing the measurement to affect the outside world), one cannot in practice just undo the measurement to restore the amplitudes of all states to their original values, putting the genie back in the bottle, as it were. Instead, the original superposition seems to be effectively destroyed, and all further measurements on the system will be in accordance with the new *collapsed* superposition.

In the traditional *Copenhagen* interpretation of this phenomenon, this event of the collapse of the global wavefunction (which would be a non-unitary and non-local transition) is said to *actually* occur. But there is an alternative explanation for the observed phenomena that is, in our view, philosophically much simpler and thus more plausible. It does not require postulating any new basic principle of "wavefunction collapse," rather, the apparent phenomenon of collapse can be seen to follow automatically from the basic nature of quantum mechanics.

This alternative is Everett's theory of the Universal Wavefunction [57] which says that regardless of whatever interactions occur between a quantum system and the external world, these interactions continue to obey unitarity, and the entire system as a whole (system being studied, plus rest of world) remains in the superposed state predicted by quantum mechanics. If the appropriate inverse transformation could be applied to the system as a whole, the original state could in principle be restored. In a measurement experiment, if information about the measurement does not leak out of the measurement apparatus into the outside world, there is no reason why the measurement, if carefully controlled, cannot be "undone" to restore the original state. (In fact, recent experiments reported in [189] confirm this.)

However, we do not have control over the quantum interactions that take place after a photon (say) that is carrying state information leaves an experiment and strikes, say, an observer's eyeball; after that interaction takes place, the information about the state of the experiment gets all mixed up with the states of billions of particles, and although a superposition is still present, the states corresponding to the different outcomes of the experiment have drifted so far about from each other through random interactions with other particles that there is essentially zero probability that they will ever drift back together to become the same state again, which is necessary in order for their amplitudes to again add up and interfere with each other. It doesn't really matter whether the photon strikes an eyeball or a rock. The different states corresponding to different outcomes of the experiment drift so far apart from each other (in Hilbert space) that it is an excellent approximation to treat them as completely independent and non-interacting. (But if the photon strikes a waveguide that directs it back into the experiment in a controlled way, that's different.)

Therefore, within the context of any state in which information has leaked out and interacted with an uncontrolled, un-modeled external environment, we can appropriately shift to using a model in which the value of this "measured" information is simply chosen according to the $|\Psi|^2$ distribution, and the amplitudes of the other states (those that are inconsistent with the leaked information) are zero. The only alternative would be to include all of the zillions of interacting particles of the external environment in one's model of the state of the system; this alternative is of course impossible for any "environment" that is not itself a known and carefully-controlled quantum system.

In any case, to avoid confusion, a more precise statement of the "measurement phenomenon" described earlier would define a measurement as any event in which state information about a part of a system interacts with an uncontrolled, unmodeled environment.

Unfortunately, distinguishing experimentally between the Copenhagen and Everett interpretations is difficult, if not impossible. (However, for an argument that it is possible, see [46].) Many philosophers have found the Everett interpretation to be highly implausible, and have dismissed it out of hand, because it implies the existence of enormous numbers of alternate versions of our universe that we are unable to interact with. However, based on my own personal view of the ideal rational method for comparing scientific theories, Everett's is clearly the *simpler* and less *ad hoc* theory, in the sense of having a more concise mathematical formulation, and therefore it is more likely; the fact that it predicts this amazing and untestable consequence of a universe that is far more elaborate than we can possibly observe is, in my view, completely irrelevant to assessing the theory's plausibility as a scientific explanation that thoroughly explains the phenomena that we *can* observe.

Interestingly, the future development of quantum computers (if successful) can be seen as stretching the bounds of the plausibility of the Copenhagen interpretation, because the functionality of a quantum computer depends on the assumption that global superposition states of large and complex systems are indeed possible. The larger the quantum computer that we can successfully build, the more implausible seems the Copenhagen viewpoint, which arbitrarily demands that the simple quantum theory (which says that all physics works through unitary transformations), depite applying perfectly well to large, complex quantum computing systems, cannot be applied identically all the way up to cover extremely large systems, *e.g.*, the whole universe.

4.2 Quantum complexity theory

After Deutsch introduced his quantum generalization of the Turing machine [45], researchers wondered whether this computational model has computational capabilities greater than those of classical Turing machines. In his original paper, Deutsch showed that quantum computers could exploit "quantum parallelism" to simultaneously compute function values for N inputs using only one mechanism. This works because the unitary transforms that apply to quantum states operate on the amplitudes of all the possible classical states of the system simultaneously. Thus, a single unitary transformation that implements the transition function f of a computation can simultaneously take the state x_i to $f(x_i)$ for all i in some index of the N initial states.

Unfortunately, this parallelism does not effectively allow one to do N times as much work with it as without it, because the N results of one computation cannot all be measured, since (as described in the previous section) communication of state information to the outside world effectively isolates the possible values of the measured state variable from each other, and so effectively causes states inconsistent with the measured information to have nonzero amplitude. The only way that information about the amplitudes of *different* mutually exclusive states can be combined is by taking unitary, linear transformations of those amplitudes *before* measuring state information.

However, for certain problems, such unitary combinations of the amplitudes of different states may provide information useful for solving the problem. Deutsch hinted at this in his original paper, showing that the XOR of the values of a boolean function on two different inputs could be computed in the time needed to evaluate the function once, if a certain transformation of the result states was performed before measuring them. The catch is that half the time, at random, the measurement yields no information—so the expected *rate* of finding these XORs is the same as with a classical algorithm that first computed the value for one input, and then the other, and XORed them.

In a later paper [47], Deutsch did much better—together with Jozsa he showed that a certain property of functions could be determined with certainty exponentially faster by quantum programs than by classical ones, if the function is given as a black box as input to the program. The property (for a function that returns 0 or 1) is whether the function is *variable* (it has value 0 for some inputs, and 1 for some inputs), or *biased* (it has one value for more inputs than the other value) if we are given that it is not both. If, on the contrary, it is both—say if its value is 1 on two-thirds of the inputs—the quantum algorithm may return either answer.

Unfortunately, it is hard to think of a realistic scenario where such an ability might be useful. For example, if the given function is a simple boolean formula applied to its input bits, we may be interested in knowing whether the function is variable (which corresponds to the famous SAT or "satisfiability" problem), but if it is, who cares whether it is biased! Unfortunately, if the function is highly biased, as is generally the case for hard SAT problems, then the algorithm will almost always answer "biased" instead of "variable," giving us no help with the satisfiability question. A classical algorithm could do just as well on SAT by trying input assignments at random. The Deutsch-Jozsa algorithm is really only helpful if we know that the input function is either constant or unbiased, and we cannot tolerate any non-zero probability of failure in determining which one it is. This seems like an unnatural problem.

But in any case, following the Deutsch-Jozsa paper, analysis of the power of quantum computers developed rapidly with papers [23, 22, 21] that defined various quantum complexity classes and compared them with various classical complexity classes in relativized oracle settings similar to Deutsch and Jozsa's. Quantum operations were also found to have uses in implementing various cryptographic operations; see the end of [23] for a summary. Quantum analogues to the popular classical complexity classes such as **BPP** (bounded-error probabilistic polynomial-time) and **ZPP** (zero-error probabilistic polynomial-time) were defined, and various of the quantum classes were shown to be larger than the various classical classes—but only in relativized oracle settings, such as we used in §3.4.2.

However, none of the oracle problems addressed seemed particularly evocative of real problems until Simon's [149] paper, which showed that the following problem was in **ZQP** (zero-error quantum polynomial-time): We are given a function f, and told that either f is 1-to-1, or else it is 2-to-1 and there is some bit-mask s such that $f(x) = f(s \oplus x)$ (where $\oplus$ is bitwise exclusive-or) for all input bit-patterns x. The problem is to determine whether the former or the latter is true, and if the latter, to find s. This seems a better than the earlier problems because it actually returns a significant amount of information about its input in the form of finding the XOR-mask s (if it exists). Anyway, Simon showed that quantum computers could solve this problem with certainty using a polynomial number of queries of the input function. Classical algorithms require exponentially many tries to achieve a reasonable probability of success.

The extraordinary thing about Simon's construction was its use of a particular unitary transformation equivalent to a special-case of the discrete Fourier transformation that had been introduced earlier by Bernstein and Vazirani [21]. Originally this Fourier transform was used to solve a certain simple oracle problem using $\mathcal{O}(1)$ queries on a quantum computer as opposed to the $\mathbf{\Theta}(n)$ queries that were classically required. The Fourier transform is linear and invertible; it turns out that it is unitary as well, and a discrete Fourier transform on functions of n-bit inputs can be performed on a quantum computer in time polynomial in n using a recursive procedure related to the classical fast-Fourier-transform (FFT) algorithm [91].

Simon's ingenious use of the quantum Fourier-transform algorithm to reduce an exponentially-hard problem to polynomial time was the original inspiration for Shor's application of a more general version of the transform to a difficult and plausibly important problem: factoring large integers. (The problem is important, at least to certain government agencies, because efficient factoring is the key to cracking RSA ([38] §33.7, p. 831), the popular public-key cryptography algorithm.) We now summarize Shor's algorithm.

4.3 Outline of Shor's Algorithm

Shor's algorithm depends on an old reduction from number theory, which translates the problem of the factorization of N to the problem of finding the *order* of a number $x \pmod{N}$. To understand the "order" concept, recall that if x is relatively prime to N, then multiplication by $x \pmod{N}$ is one-to-one. Therefore, the series $x^0, x^1, x^2, \ldots \pmod{N}$ eventually gets back to 1 and cycles around again, *i.e.*, $\exists r > 0 : x^0 \equiv x^r \pmod{N}$. The order of $x \pmod{N}$ is defined to be the least such r. The connection with factoring is that if r is even, then either $(x^{r/2} - 1) \bmod N$ or $(x^{r/2} + 1) \bmod N$ has a common factor with N, which can then be easily found using Euclid's algorithm for finding the greatest common divisor (gcd) ([38], §33.2, p. 808.)

Therefore, if a polynomial-time algorithm for finding the order of a number mod N is available, then it can be used to factor N as follows:

1. Pick a random $x < N$.

2. Compute $f = \gcd(x, N)$; if $f \neq 1$, return f. (It's a factor.)
3. Find the least r such that $x^r \equiv 1 \pmod{N}$.
4. If either $\gcd(x^{r/2} - 1, N)$ or $\gcd(x^{r/2} + 1, N)$ is not 1, return it, it's a factor.
5. Otherwise, go to step 1 and repeat.

The number of repitions of the above loop required to find a factor with probability ≥ 0.5 can be shown to be only polynomial in the length of N, therefore if all the steps 1-4 are polynomial, then the algorithm as a whole takes only polynomial time.

The bottleneck in the algorithm is of course the computation of r, which Shor implements via a clever application of the Fourier transform to quantum parallelism. The quantum computer is made to compute x^r for all $r < N^2$ simultaneously. As discussed earlier, the series of x^r ($\pmod{N}$) is cyclically repeating if x is relatively prime to N; the period of repitition is the order of x. This cycle of repeating values is stored simultaneously in N^2 distinct states corresponding to the N^2 different input values of r; each state is present with the same amplitude. Then, the quantum Fourier transform is applied to the superposition of result states. Instead of representing an equal superposition of all the r and $x^r \bmod N$, the function from states to amplitudes now encodes a superposition of frequency spectra for finding the different values of $x^r \bmod N$. These spectra will have amplitude peaks at points correspondsing to multiples of the basic repitition period of x^r. If we then measure the state, it will (with high probability) lie very near one of the peaks, and the value of the state measured will let us guess the repitition period (which is the answer we are looking for) with high probability.

Let us examine how this is done in more detail, with reference to the example illustrated in figs. 4.1 and 4.2. In this example, the number to factor is $N = 3 \times 11 = 33$. Let q be the smallest integral power of 2 greater than N^2, and let $\ell = \lg q$. In the real algorithm q would be 2048, but we will take $q = 256$ instead for ease of visualization. We will have two quantum registers (a, b), where a is an ℓ-bit register ranging from 0 to $q - 1$, and b ranges from 0 to $N - 1$.

In figs. 4.1 and 4.2 we illustrate the superposition state of the joint space of these two registers, following each stage of Shor's algorithm. In these figures, the position along the horizontal axis represents the state of register a, and the position along the vertical axis represents the state of register b. At every point on the resulting two-dimensional surface representing the combined state space, we place a blob of ink whose darkness corresponds to the absolute amplitude of that state. (Additionally, when displayed on a color output device, the hue of the ink in these figures indicates the phase of the amplitude.) White areas correspond to states having zero or nearly zero amplitude.

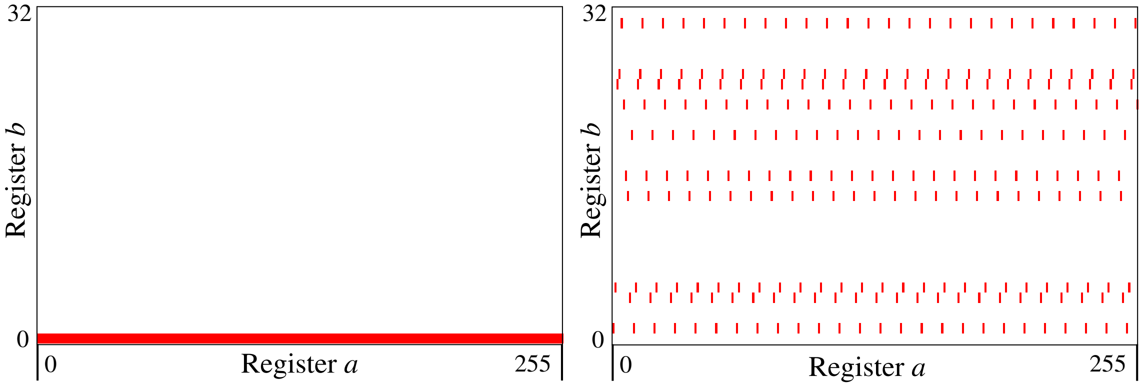

Figure 4.1: Example of Shor's algorithm with $N = 33$, $q = 256$, $x = 5$. Left: superposition over all values of a, with $b = 0$. Right: superposition after computing $b = x^a \mod N$. Across a there is periodic cycling of b through the 10 values 1, 5, 25, 26, 31, 23, 16, 14, 4, 20.

We start with an initial quantum state of $|0, 0\rangle$, that is, $a = 0$, $b = 0$. First we prepare an equal superposition of all values of a while leaving $b = 0$. The left side of fig. 4.1 shows the amplitude spread out over all the values of a, in the row corresponding to $b = 0$. This is the superposition

$$\sum_{a=0}^{q-1} |a, 0\rangle / \sqrt{q}.$$

Next, x is chosen at random (in our case, to be 5), and we perform a classical reversible computation to transform the value of b from 0 to $x^a \mod N$. The right side of fig. 4.1 shows how the blob of amplitude associated with each value of a is moved vertically to the state also having the correct value of b. With increasing a, we see that b cycles periodically among various values, with a period that is (by definition) the order of x mod N, which in this case is 10.

Finally, leaving b, alone we perform a Fourier transform over the value of register a. This is defined by

$$\psi'(a', b) = \sum_{a=0}^{q-1} \exp(2\pi i a a'/q) \psi(a, b) / \sqrt{q}$$

Figure 4.2 illustrates how the amplitude in each row associated with each value of b is moved horizontally to cluster in peaks, whose number corresponds to the period of the original amplitude distribution. Since the period is the same for each value of b, the peaks in each row line up, although they have a different phase in different rows. (The color version of this document shows the amplitudes in color, with the hue denoting the phase.)

After this, register a may be sampled, and several samples will be sufficient to tell us its period, 10 in our case. Now the rest is easy: Half of 10 is 5, and $x^5 \bmod N$ is 23 in our case. Twenty-three minus 1 is 22, which has a common factor (11) with our N, and twenty-three plus 1 is 24, which also has a common factor (3) with our N.

The details of exactly how the quantum Fourier transform works are beyond the scope of this short survey. For more detailed expositions of Shor's algorithm, see Ekert and Jozsa's description in [54, 53], and Shor's original papers [146, 147].

4.4 Important open problems

Here are some of what I believe to be the most important, and also the most difficult open questions in quantum computing:

4.4.1 Can quantum computers solve **NP**-complete problems in polynomial time?

This is perhaps the most interesting question about quantum computing. If the answer to this question were positive, then quantum computing could revolutionize computing, as we know it. There are a wide range of practical problems in constraint-satisfaction, combinatorial search, and other areas that have been shown to be in **NP**, but for which no efficient classical algorithms are known.

The ability to solve **NP**-complete problems efficiently would also revolutionize all of mathematics, because it would enable us to quickly determine, for any given mathematical statement, whether or not there is a fairly simple proof (or disproof) of the statement, and if there is, to find it. Automatically checking a (suitably formal) proof for correctness can be done in polynomial time in the proof length; therefore, finding a proof of a given length can be done in nondeterministic polynomial time in the target length. Therefore a method for solving **NP** problems in polynomial time could find proofs in time polynomial in the proof length.

Currently, it seems unlikely that quantum computers could solve **NP**-complete problems, due to the fact that the only known quantum algorithms that dominate classical algorithms either involve unrealistic oracle-dependent promise problems [21, 149], or introduce only polynomial speedups [77, 78], or only simulate quantum mechanics [25], or depend on the ability to reduce the problem to one involving periodicities for which the quantum Fourier transform is useful [147, 26]; one would not expect such periodicities *a priori* to be characteristic of all problems in **NP**.

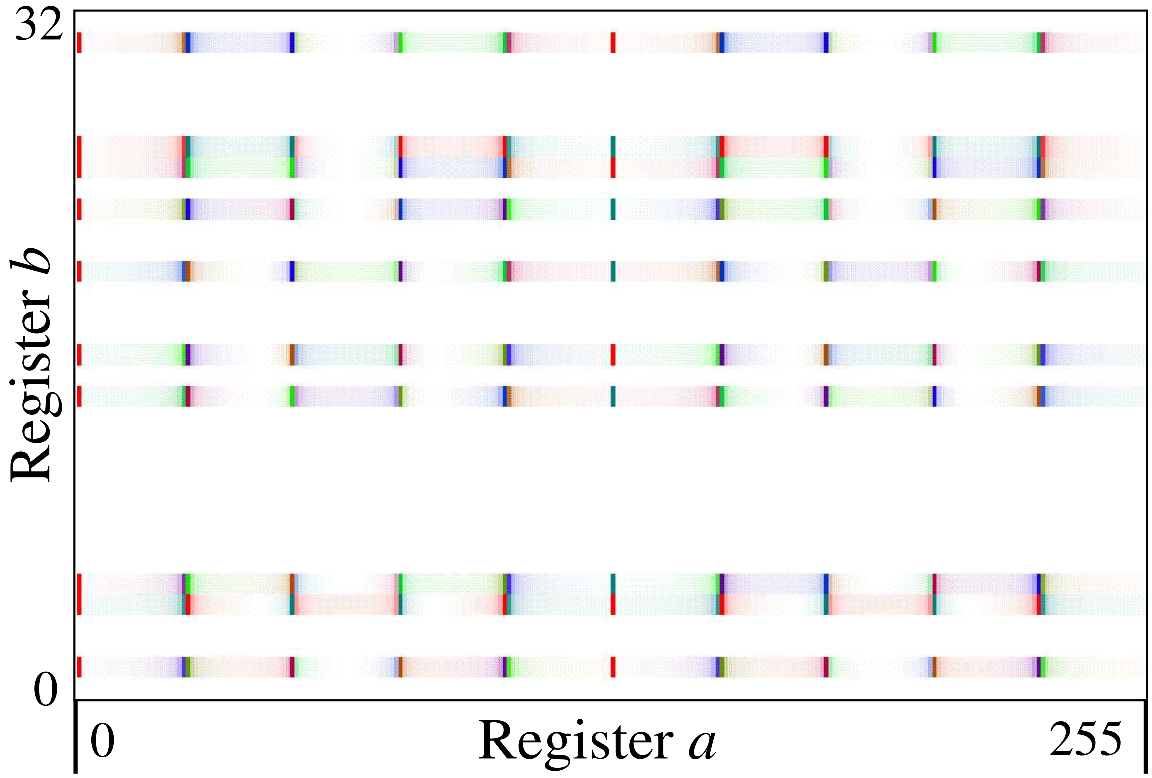

Figure 4.2: Superposition after performing quantum Fourier transform over register a in Shor's algorithm, when factoring $N = 33$ with the choice of $x = 5$. The value of a is now peaked at 10 points spaced 25.6 units apart. The number of peaks is the period (order) of the function 5^a mod N, shown in the right half of fig. 4.1. After the transform, sampling a permits determining the period, and halving it, we find 5^5 mod $33 = 23$, which is 1 away from numbers that have common factors with 33, and the problem is solved.

However, it is conceivable that someone may yet discover a clever quantum algorithm for general simulation of nondeterministic Turing machines. One possible approach might be to reduce some **NP**-hard problem to the factoring or discrete logarithm problem; however, complexity theorists have attempted such reductions for many years without success.

A more sophisticated approach might involve finding some other unitary transformation other than the Fourier transform that could allow states associated with parallel paths in a quantum simulation of a nondeterministic machine to constructively interfere in ways that might yield useful information about the structure of the search space, and help to pin down the solutions. Hogg [85] has investigated quantum algorithms that enhance the probability density found along solution paths in **NP** search problems, but not enough to allow measurements of the machine state for such problems to yield solutions in expected polynomial time.

On the other side of the question, Bennett *et al.* [20] have provided suggestive evidence against the $\mathbf{NP} \subseteq \mathbf{BQP}$ conjecture, by showing that when R is a random oracle, $\mathbf{NP}^R \supset \mathbf{BQP}^R$ with probability 1. However, it is worth noting that since $\mathbf{BQP} \supseteq \mathbf{P}$, an actual proof that $\mathbf{NP} \supset \mathbf{BQP}$ would imply that $\mathbf{NP} \supset \mathbf{P}$, a conjecture whose proof has long eluded complexity theorists. Thus, it seems unlikely that the possibility of quantum computers subsuming nondeterministic computers will be conclusively ruled out anytime soon.

4.4.2 Are quantum computers strictly more powerful than classical computers (with a bounded probability of error)? *I.e.*, $\mathbf{BQP} \supset \mathbf{BPP}$?

This question at first appears the same as the previous one, but there are two important differences. First, it may be the case that $\mathbf{P} = \mathbf{NP}$, in which case both quantum and classical computers could solve all problems in **NP** efficiently, and so the quantum computers might not be any more powerful than the classical ones.

Secondly, if the answer to the question is negative, *i.e.* if $\mathbf{BQP} \subseteq \mathbf{BPP}$, then this has important implications for physics, because it might mean that existing classical computers could therefore simulate arbitrary quantum systems with only a polynomial slowdown, which is not currently known to be possible [59]; current classical simulations of quantum systems all suffer from an exponential slowdown.

A faster method for simulating quantum physics would revolutionize much of theoretical physics, because it would allow many more predictions to be derived from quantum theories, predictions which could then be compared with experiment to refine the theories; and it would also reduce the current dependence on approximation methods in many important areas of applied physics, such as modeling molecular interactions, *e.g.* for drug design.

Unfortunately for these tantazlizing prospects, the existence of Shor's factoring algorithm, together with the long-standing failure of many generations of brilliant mathematicians to find a classical equivalent, seems to leave little hope for a quantum-classical equivalence. Even if quantum computers are not as powerful as nondeterministic Turing machines, they may be strictly more powerful than deterministic ones.

4.4.3 Can errors caused by imprecision and decoherence be controlled sufficiently to allow arbitrarily complex quantum computations to take place with an arbitrarily small probability of failure?

A number of papers have expressed pessimism regarding the question of error accumulation in quantum computers, *e.g.*, [96, 95, 173, 174, 33, 132]. These papers show that in the absence of error correction, the probability of error increases exponentially with both the time and space complexity of the computation, and the expected error-free running time for various experimental setups has been estimated to be roughly on the order of the time to perform a single computational step, seemingly ruling out the possibility of doing interesting quantum computations.

However, Coppersmith [36] has shown that simple imprecision does not cripple the quantum factoring algorithm, and several more recent papers [28, 32, 155] have addressed the more difficult issue of correcting errors due to decoherence of the quantum states. They work by encoding a bit value redundantly in a superposition of many bit values in such a way that up to n independent interactions of bits with the environment can take place without communicating any information about the value of the encoded bit to the environment. If fewer than n bits interact with the environment, than the system can exactly recover the originally-encoded superposition and then regenerate its redundant representation. The only problem is that this reconstruction process will in general be subject to errors as well. More sophisticated techniques might take that into account. In summary, although these papers appear to be on the right track to a solution, a more complete theory of quantum error correction is still needed, and remains to be worked out.

Cesar Miquel's 1996 preprint [127] reports results of some simulation experiments on error-correcting versions of Shor's algorithm in the presence of errors.

4.4.4 How do we build it, physically?

Although the question of how to implement quantum computations physically is of course a question of utmost importance for the future of the field, to a large extent the details of the physical implementation are orthogonal to most of the theoretical issues dealt with in the literature we have reviewed. There is another huge literature, mainly under the rubric of experimental physics (rather than quantum theory or computer science), which deals with constructing physical realizations of systems of controlled interactions between quantum states. For example, researchers in quantum optics study how to manipulate information encoded in the polarization staes of photons; "cavity QED" workers study the interactions between photons and electron spins on individual atoms [56, 168]; and other experimentalists work with vibrational states in assemblages of interacting atoms [35].

An intriguing recent development in implementation techniques has been the investigation of NMR "ensemble quantum computing," in which the nuclear spins of atoms in molecules in solution are made to interact using nuclear magnetic resonance techniques [39, 76, 40]. The NMR experiments have had the most success of any techniques to date; quantum logic operations involving 2 and 3 bits have been successfully demonstrated.

Even more recently, Mooij *et al.* [128] have described how to implement quantum computation on a chip surface in microlithgraphed superconducting Josephson-junction circuits. In this scheme, quantum bits are encoded in the direction of a quantized current flow in a superconducting loop. The system is projected to have very long decoherence times, making it fairly amenable to quantum error correction algorithms, and moreover, quantum circuits of arbitrary size and complexity can be readily patterned. At the moment, this approach seems the most likely candidate for implementing a practical quantum computer in the near future.

Some older proposals for implementation technologies for quantum computing, from various communities, include Teich *et al.* '88 [158], Lloyd '93 & '94 [112, 113], DiVincenzo '95 a [49], Sleator & Weinfurter '95 [150], Barenco *et al.* '95 b [10], and Chuang & Yamamoto '95 [34].

The main lesson to be learned from this long list of proposals is that the details of the physical implementation of quantum computers are "just" an engineering concern, rather than a theoretical issue of fundamental importance. Researchers since Feynman [60] have noted that there seems to be nothing fundamental in quantum physics that precludes using it for computation, and indeed, the multiplicity of ideas listed above seems to bear that out. Although certainly the development path of many particular techniques will be beset with problems, it seems likely that eventually our technologies for manipulating quantum particles will mature to the point where some form of complex controlled assemblages of quantum states will be built fairly readily—that is, if it is useful to do so. The question of whether it would be useful can only be answered by the theoretical studies such as those we surveyed in this chapter; no matter how a quantum computer is finally built, the theorems and algorithms produced by those studies will still apply.

4.5 Summary of quantum computation

Although it appears that the basic computational steps of quantum computers may soon be implementable, and that a large quantum computer may be able to factor numbers faster than a classical computer, many practical problems such as error-correction remain to be solved before we can scale up to large enough computers to be useful. Another possible show-stopper is that factoring and a handful of other problems relating to cryptography [26] may turn out be the only real-world problems amenable to fast quantum solutions, which may not provide enough motivation to support the development of quantum computers. Although cracking RSA is a tantalizing prospect, it would not necessarily change the world radically—RSA might just be replaced by another code that is less amenable to quantum solution, especially perhaps one using quantum crypography. However, there is hope that applications such as the use of quantum computers to efficiently simulate models of real quantum physical systems [25] might revolutionize physics as we know it.

In chapter 2 and in this chapter, we have seen how some basic physical principles affect the limits of what is possible with computers, sometimes in surprising ways. In the next chapter, we will discuss the benefits to be gained from developing "ultimate" models of computation that accurately reflect these limits, and we will propose some candidates for an ultimate model. The ultimate model may or may not turn out be able to use large-scale quantum coherence in the way that we have discussed in this chapter, but in chapter 6 we will see that, at the very least, the ultimate model must be reversible.