Toward Quantum Computation: A Five-Qubit Quantum Processor

QUANTUM PHYSICS PRESENTS INTRIGUING POSSIBILITIES FOR ACHIEVING COMPUTATIONAL GAINS AFTER CONVENTIONAL MINIATURIZATION REACHES ITS LIMITS. ACCORDINGLY, WE DESCRIBE A NUCLEAR MAGNETIC-RESONANCE QUANTUM COMPUTER DEMONSTRATING A QUANTUM ALGORITHM THAT EXPONENTIALLY OUTPERFORMS CLASSICAL ALGORITHMS.

Matthias Steffen
Lieven M.K. Vandersypen
Stanford University

Isaac L. Chuang
IBM Almaden Research Center

The unceasing miniaturization of semiconductor integrated circuits is widely expected to end within the next 20 years due to the fundamental impossibility of patterning features smaller than the length scale of single atoms and molecules.[1] However, the laws of quantum physics arising at such small scales provide intriguing possibilities for obtaining computational speedups by enabling algorithmic feats that would otherwise be impossible. Such possibility is the promise of the quantum computation and quantum information fields, which seek to exploit quantum physics for solving classical information processing and communication tasks.[2,3]

Quantum computers are extremely challenging to experimentally realize due to the difficulty of retaining quantum properties of systems while simultaneously controlling their dynamics. However, in the past three years, a laboratory technique based on nuclear magnetic resonance has been unexpectedly successful in implementing quantum computers with a few "quantum bits" (qubits) and in demonstrating simple quantum algorithms.[4,5]

In a recent experiment using such techniques,[6] we implemented a quantum computer with five qubits to test two key features in fast quantum algorithms—mathematical computation of modular exponentiation and the quantum fast Fourier transform. These features underlie Shor's famous quantum factoring algorithm, which exponentially outperforms the best-known classical algorithms for factoring integers.[7,8]

Theory of quantum computing

A description of the fundamental concepts behind quantum computation begins with the complexity of problems, qubits, and a look at quantum parallelism, quantum algorithms, and the challenges of building a quantum computer.

The power of quantum computers

The theoretical promise of quantum computers is their ability to fundamentally reduce the resources required to solve real and relevant mathematical problems. To understand how fast quantum computers are compared to their classical counterparts, we must compare their speed in a way that is technology

0272-1732/01/$10.00 © 2001 IEEE

independent. Specifically, modern classical computers equipped with MHz and GHz processors completely outperform all present quantum computers, which have clock frequencies on the order of hundreds to thousands of hertz. However, because of the way they use quantum interactions, quantum computers can perform instructions impossible on classical machines and so require fewer steps to solve certain mathematical problems.

Complexity theory provides the basis for comparing quantum and classical computers. Complexity theory analyzes the fundamentally minimal physical resources (time, space, energy) demanded by an algorithm to solve a given problem, as a function of problem size n. The key distinction in comparing different models of computation is whether the resources required are polynomial or exponential in n. Two n-digit numbers, for example, can be added in $O(n)$—a linear function of n—elementary operations such as NAND gates.

In contrast, factoring an n-digit integer into prime numbers requires exponentially many operations by the best-known algorithms, $O(e^{n^{1/3}})$to be precise. Since there is no efficient—that is, polynomial—classical algorithm for prime factorization, computer scientists currently consider this problem intractable on classical machines. By increasing n, it quickly becomes impossible to factor an n-digit integer in a reasonable time, even using the fastest conceivable supercomputers.

The allure of quantum computation rose dramatically in 1994 when Peter Shor showed that, on a quantum computer, factoring an n-digit number could be accomplished using only $O(n^3)$ elementary quantum operations.[7,8] This is exponentially fewer operations than is possible classically. A similar speedup is possible for simulating the dynamics of quantum systems, as Richard Feynman showed.[2]

Quantum computers provide a speedup for other problems including searching unsorted databases,[9] although here the advantage is only quadratic rather than exponential. Quantum mechanics also allows certain distributed computation and communication tasks to be sped up significantly.[2]

Quantum parallelism

How do quantum computers work? The starting point is a remarkable theorem stating that quantum computation subsumes classical computation. This is not obvious, since the laws of quantum mechanics demand microscopic reversibility, whereas today's classical computers are not at all reversible. As evidence, note that a computer generates heat and requires power to operate. However, in 1973, Charles Bennett proved that an ideal classical computer can, in principle, be made to dissipate no energy and thus operate reversibly.[2] This result implies that quantum machines can perform any classical computation, using the microscopic reversibility of the governing equation of quantum physics: Schrödinger's equation.

Given that quantum computation subsumes classical computation, it's natural to employ a language of quantum bits and quantum circuits analogous to the classical case. Any quantum system with two distinct discrete states can, in principle, serve as a quantum bit—a photon with vertical or horizontal polarization (↕ or ↔), an electron spin that points up or down (↑ or ↓), an electron located in one of two quantum dots, and so forth. The two quantum mechanical states are denoted $|0\rangle$ and $|1\rangle$, corresponding to logical zero and one. We can then devise quantum logic gates that act on the qubits exactly like (reversible) classical gates act on classical bits.

What really distinguishes quantum bits from classical bits is that qubits, unlike classical bits, can exist in so-called superposition states written as $a|0\rangle + b|1\rangle$, where a and b are complex numbers satisfying $|a|^2 + |b|^2 = 1$. In some sense, this means that a qubit can be in $|0\rangle$ and $|1\rangle$ at the same time. Consider now what happens if we evaluate a 1-bit logic gate that maps $|x\rangle$ to $|f(x)\rangle$ (where f is a classical Boolean function), when the input qubit is prepared in the state

$$\frac{1}{\sqrt{2}}\left|0\right\rangle + \frac{1}{\sqrt{2}}\left|1\right\rangle \qquad (1)$$

an "equal" superposition of $|0\rangle$ and $|1\rangle$. By linearity of quantum mechanics, the logic gate f transforms the qubit state to

$$\frac{1}{\sqrt{2}}\left|f(0)\right\rangle + \frac{1}{\sqrt{2}}\left|f(1)\right\rangle \qquad (2)$$

The output state is now a superposition of the two output values. In this sense, f is evaluated for both possible input values in one step.

Similarly, we may prepare each of two qubits in a superposition of $|0\rangle$ and $|1\rangle$, so together they are in a superposition of four states:

$$c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle \quad (3)$$

A 2-qubit logic gate g will transform this state to

$$c_0|g(00)\rangle + c_1|g(01)\rangle + c_2|g(10)\rangle + c_3|g(11)\rangle \quad (4)$$

so in a sense g has been evaluated for four input values in parallel. For every extra qubit involved in the computation, the number of parallel function evaluations doubles. This exponential parallelism became known as *quantum parallelism*.

Making use of quantum parallelism is tricky, however. Quantum mechanics dictates that when we measure a qubit or set of qubits in a superposition state, the superposition *collapses*—that is, only one term in the superposition is observed. So, upon measurement of the state of Equation 4, for example, we obtain $g(00)$ with probability $|c_0|^2$, $g(01)$ with probability $|c_1|^2$, and so forth. Thus, straightforward measurement does not simultaneously provide all output values produced by quantum parallelism, which limits the power of quantum computers. Nevertheless, special quantum algorithms let us exploit quantum parallelism to solve certain problems in far fewer steps than is possible classically.

Shor's factoring algorithm

The most remarkable quantum algorithm to date serves to efficiently determine a particular function's period. The significance of period finding is that from the period r of the function $f(x) = a^x \bmod N$ (the remainder of a^x divided by N), the prime factors of N can be computed quickly using results from number theory. No efficient classical algorithm is known to find periods, but with Shor's algorithm, we can find r using a number of quantum bits and gates polynomial in the length of N.[7,8]

The heart of Shor's algorithm is the quantum Fourier transform (QFT). The QFT is closely related to the well-known classical fast Fourier transform (FFT), but can be computed exponentially faster. The FFT takes as input a string of K complex numbers, x_k, and produces as output another string of K numbers, y_k, with

$$y_k = \frac{1}{\sqrt{K}} \sum_{j=0}^{K-1} x_j e^{2\pi ijk/K} \quad (5)$$

For an input string of K numbers, which repeat themselves with period r, the FFT produces an output string with period K/r, as illustrated in the following examples for the case of $K = 8$ (the output strings have been rescaled for clarity).

1 0 0 0 0 0 0 0 → 1 1 1 1 1 1 1 1
1 0 0 0 1 0 0 0 → 1 0 1 0 1 0 1 0
1 0 1 0 1 0 1 0 → 1 0 0 0 1 0 0 0
1 1 1 1 1 1 1 1 → 1 0 0 0 0 0 0 0

In case r divides K with a remainder, the inversion of the period is only approximate.

Furthermore, the FFT turns shifts in the input string into phase factors in the output string:

1 0 0 0 1 0 0 0 → 1 0 1 0 1 0 1 0
0 1 0 0 0 1 0 0 → 1 0 –i 0 –1 0 i 0
0 0 1 0 0 0 1 0 → 1 0 –1 0 1 0 –1 0
0 0 0 1 0 0 0 1 → 1 0 i 0 –1 0 –i 0

The QFT performs exactly the same transformation as the FFT, but the complex numbers are stored in the amplitude and phase of the terms in a superposition. For example, the 2-qubit superposition $c_0|00\rangle + c_1|01\rangle + c_2|10\rangle + c_3|11\rangle$ represents the complex numbers c_0, c_1, c_2, and c_3.

A 5-qubit example shows how quantum parallelism and use of the QFT make it possible to efficiently find the period of a function. For clarity, the states are written in decimal instead of binary notation, for example, $|010\rangle$ will be denoted $|2\rangle$.

Consider two registers (groups of qubits) in which the first register contains three qubits, each initialized to an equal superposition of $|0\rangle$ and $|1\rangle$, and with the second register consisting of two qubits set to $|0\rangle$. Suppressing normalization constants for clarity, the state of the system is thus written as

$$(|0\rangle + |1\rangle + |2\rangle + |3\rangle + |4\rangle + |5\rangle + |6\rangle + |7\rangle)|0\rangle \quad (6)$$

We now evaluate some function $f(x)$ with a period r, which is initially unknown. The first register represents x, and the output $f(x)$ is

stored in the second register. For some $f(x)$ with $r = 2$, the initial state is thus transformed to

$$|0\rangle|3\rangle + |1\rangle|1\rangle + |2\rangle|3\rangle + |3\rangle|1\rangle + |4\rangle|3\rangle + |5\rangle|1\rangle + |6\rangle|3\rangle + |7\rangle|1\rangle \quad (7)$$

or also

$$(|0\rangle + |2\rangle + |4\rangle + |6\rangle)|3\rangle + (|1\rangle + |3\rangle + |5\rangle + |7\rangle)|1\rangle \quad (8)$$

This is where quantum parallelism manifests itself—even though we apply the function $f(x)$ once, it is evaluated for eight possible values of x. However, no measurement can, due to the collapse of the quantum states, extract the outcomes for all eight input values simultaneously. Instead, we now apply the QFT to the first register, which transforms the state of Equation 8 to

$$(|0\rangle + |4\rangle)|3\rangle + (|0\rangle - |4\rangle)|1\rangle \quad (9)$$

We can verify that the QFT removed the shift in the first register seen in Equation 8 and turned it into a phase factor. Furthermore, the QFT inverts the period from $r = 2$ to $K/r = 8/2 = 4$. Now measurement of the first register yields useful information because the quantum states of the first register can only collapse to multiples of K/r—in this case $|0\rangle$ or $|4\rangle$. In contrast, for the state of Equation 8, measurement of the first register randomly returns one of the eight states $|0\rangle, \ldots, |7\rangle$. From the measurement result (a multiple of K/r), the inverted period K/r can be extracted efficiently on a classical computer via the continued fraction expansion—a technique from number theory—provided the first register is large enough. Then the period r can be immediately derived as well. Each of the above steps can be completed in polynomial effort, so period finding can indeed be accomplished efficiently on a quantum computer.

Requirements and challenges

Quantum computers' enormous theoretical promise motivates an investigation of the practical requirements for the experimental implementation of such a device. First, we need a system of qubits. Second, the qubits must be individually addressable and must interact with each other to provide for a universal set of logic gates. Third, it must be possible to initialize them to a known state because the result of a computation generally depends on its input state. Fourth, we must be able to extract a computation result from the qubits by some measurement.

The last three requirements mandate experimental access to and external control over the quantum system. However, interactions between a quantum system and the environment (everything outside the quantum system) necessarily cause the quantum system to lose its quantum properties. In particular, "open" quantum systems can sustain superposition states only for a limited time, known as the coherence time. A quantum computation must be performed within this time window because quantum algorithms inherently rely on superpositions. Thus, the fifth and final requirement is a long coherence time compared with an average logic gate's duration, such that many logic gates can be implemented within the coherence time.

Meeting all these requirements simultaneously poses a significant experimental challenge. How can we gain access to a quantum system and, at the same time, keep coherence times long?

Nuclear magnetic resonance quantum computing (NMR QC)

Nuclear magnetic resonance (NMR) techniques largely satisfy the practical requirements described in the previous section and have enabled the experimental exploration of small-scale quantum computers.

Concept

Many atoms such as ^{1}H, ^{13}C, and ^{19}F have a *spin-1/2* nucleus. A nuclear spin-1/2 can be thought of as a tiny bar magnet spinning about its own axis, with two well-defined states; it can be aligned or anti-aligned with respect to an external magnetic field (spin up or spin down) and thus can represent logical zero and one. Since a nuclear spin is extremely small, it's a quantum mechanical object and can exist in a superposition of up and down.

We can visualize the state of a single spin as a point on a sphere, as shown in Figure 1, analogous to a bar magnet pointing in a particular direction. However, this is only a pictorial representation—a spin that we describe as point-

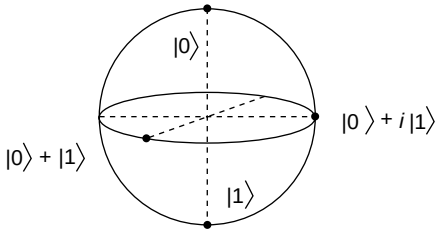

Figure 1. Representation of four different qubit states.

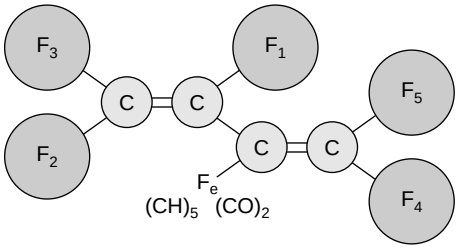

Figure 2. Molecule with five fluorine spins. All five spins are well-separated spectrally and are pairwise coupled. This molecule was used as a 5-qubit computer. Copyright © 2000, *The American Physical Society.*

ing somewhere halfway between up and down is really in a quantum mechanical superposition state. An atomic nucleus' spin can thus serve as a quantum bit.

An NMR quantum computer consists of several individual atoms with a spin-1/2 nucleus. We could place these atoms on a surface, bury them in a bulk material, or chemically bond them in a molecule. Because the first two approaches require substantial new technology development, we took the molecular approach in our quantum computing demonstrations.

Researchers identified nuclear spins early on as excellent candidates for quantum computers because of their long typical coherence times—often several seconds—compared to nanoseconds for electron spins in solids, for example.

Having satisfied the first and fifth requirements—establishing qubits with long coherence times—we now show how we can satisfy the second requirement: to implement arbitrary single-qubit operations and a specific 2-qubit operation (the controlled-NOT or CNOT gate). Together, these operations form a universal set of quantum logic gates, the equivalent of a NAND gate or a NOR gate for classical logic.

Arbitrary single-qubit operations are accomplished by applying electromagnetic fields in the transverse plane (perpendicular to the static magnetic field along *z*). When placed in a static magnetic field, a spin—and also a bar magnet—will precess about the magnetic field's axis with a frequency linear in the field strength. This motion is similar to how a spinning top precesses about the gravity axis. Typical frequencies are in the radio-frequency (RF) range. If, in addition, we apply a transverse RF field on resonance with the spin precession frequency, the spin state will gradually rotate about an axis in the transverse plane (the point on the sphere of Figure 1 will move).[10]

The exact axis of rotation depends on the phase of the RF field; the rotation angle is proportional to the RF field's duration and amplitude. A properly timed and calibrated RF pulse can thus flip the state of a spin from up to down or vice versa, thereby implementing a NOT gate. An RF pulse of half the duration of a NOT gate rotates a spin from up ($|0\rangle$) into a superposition of up and down ($(1/\sqrt{2}\,|0\rangle+1/\sqrt{2}\,|1\rangle)$).

Selective addressing of one spin without affecting the state of any other spin is possible because different kinds of atoms (^{1}H, ^{19}F, …) have different resonance frequencies ν. Furthermore, if the molecule exhibits sufficient asymmetry in its structure, the resonance frequencies of different atoms of the same kind (for example, two ^{19}F atoms) are also shifted with respect to each other (chemical shift). Figure 2 shows a molecule with five ^{19}F atoms that exhibit remarkably distinct spin precession frequencies, allowing each spin to be addressed individually.

All 2-qubit gates require an interaction between the qubits. In modern computers, transistors provide the means for two input voltages to "communicate," leading to a third output voltage that represents the result of a 2-bit interaction. In NMR, shared chemical bonds between atoms provides a natural interaction between spins.

Consider two neighboring spin-1/2 atoms (or two tiny bar magnets) in each other's vicinity. Spin 2 is subject to the stray field produced by spin 1 (either aligned or anti-aligned with *z*), in addition to the externally applied magnetic field. Since a spin's precession frequency is proportional to the magnetic field strength it's placed in, the precession frequency of spin 2 is now $J/2$ lower or higher depending on whether spin 1 is up ($|0\rangle$) or down ($|1\rangle$), where J is the coupling strength expressed in frequency units.

In NMR quantum computing, we desire molecules where the frequency differences between different spins are much larger than the J-coupling. We can then implement a simple 2-bit gate by applying a narrowband 180-degree pulse at $\nu_2 + J_{12}/2$, such that spin 2 is inverted if and only if spin 1 is $|1\rangle$. This is the CNOT gate; it performs a NOT operation on the target qubit if and only if the control qubit is $|1\rangle$: A CNOT of spin 1 onto spin 2 performs the transformation $|00\rangle \rightarrow |00\rangle$, $|01 \rightarrow |01\rangle$, $|10\rangle \rightarrow |11\rangle$, and $|11\rangle \rightarrow |10\rangle$.

Figure 3 shows an alternative and more widely used implementation of the CNOT gate.[4,5] The diagram shows the evolution of spin 2 in a coordinate system that rotates around the z axis at ν_2, when spin 1 is $|0\rangle$ (solid line) and $|1\rangle$ (dashed line).

First, an RF pulse centered at ν_2 and of a spectral bandwidth such that it covers the frequency range $\nu_2 \pm J_{12}$ but not ν_1, rotates spin 2 from $+z$ to $-y$. Then we allow the spin system to freely evolve for a duration of $1/2J_{12}$ seconds. Because the precession frequency of spin 2 is shifted by $\pm J_{12}/2$ depending on whether spin 1 is in $|1\rangle$ or $|0\rangle$, after $1/2J$ seconds spin 2 will have rotated to either $+x$ or to $-x$ (in the reference frame rotating at ν_2), depending on the state of spin 1. Finally, a 90-degree pulse on spin 2 about the $-y$ axis rotates spin 2 back to $+z$ if spin 1 is $|0\rangle$, or to $-z$ if spin 1 is in $|1\rangle$. The net result is that spin 2 is flipped if and only spin 1 is in $|1\rangle$.

Two complications arise when we want to implement 2-qubit gates for systems with more than two spins; Figure 4 illustrates the two extreme scenarios of coupling networks.

- If every spin is coupled to every other spin (possible only in relatively small molecules), the pulse sequence of Figure 3 must be supplemented by "refocusing" pulses, designed to remove the effect of all couplings except between the two spins involved in the 2-qubit gate.
- Performing a CNOT gate between two spins that aren't directly coupled to each other is still possible if a network of couplings connects them. For example, if spin 2 is coupled to 1 and 3, but 1 and 3 are not coupled to each other, we can perform a CNOT of spin 1 on spin 3 as follows: first swap the state of spins 1 and 2 (achieved with a sequence of three CNOTs), then perform a CNOT of 2 and 3, and finally swap 1 and 2 back.

A computation on an NMR quantum computer thus consists of the application of a carefully designed sequence of RF pulses separated by delay times. We can view those elementary instructions—pulses and delay times—as the machine language. Furthermore, the process of decomposing a high-level description of an algorithm into 1- and 2-qubit gates and, subsequently, into RF pulses and delay times, is analogous to compiling code on a classical computer.

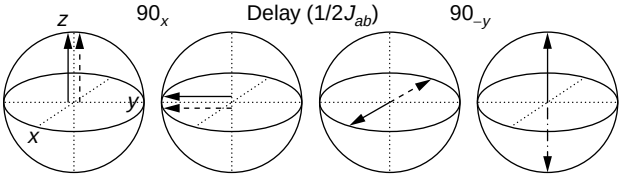

Figure 3. Implementing the CNOT gate.

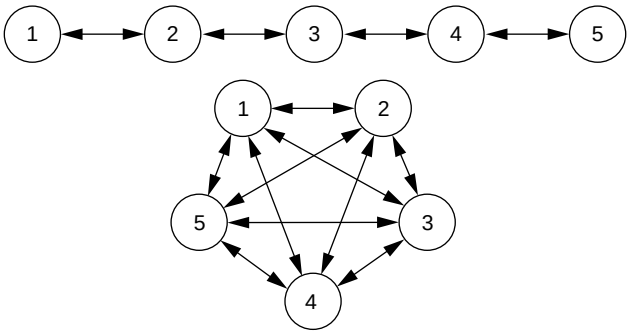

Figure 4. Two possible coupling networks for a five-spin molecule. The circles denote the qubits while arrows denote coupling between 2 qubits.

The third requirement for building a quantum computer is the ability for state initialization. The initial state that is experimentally most easily accessible is the thermal equilibrium state—equilibration simply means waiting a few minutes. However, the thermal equilibrium state of a nuclear spin at room temperature is highly random. The spin-up and spin-down states are almost equally likely, with a bias of only about 1 in 10^5. The desired initial state for quantum computations, in contrast, is a pure state, for example, the state where all the spins are in $|0\rangle$.

Although it's not currently possible to create a pure state with nuclear spins at room temperature, it is possible to create an "effective pure" state, which produces a signal proportional to the pure state signal. This insight, along with explicit procedures for creating effective pure states, was the main conceptual breakthrough that made NMR quantum computing possible [4,5]

The fourth requirement is that the final state of the qubits must be read out because it constitutes the result of the computation. This is done with a read-out pulse. If the final state of a spin is either $|0\rangle$ (along $+z$) or $|1\rangle$

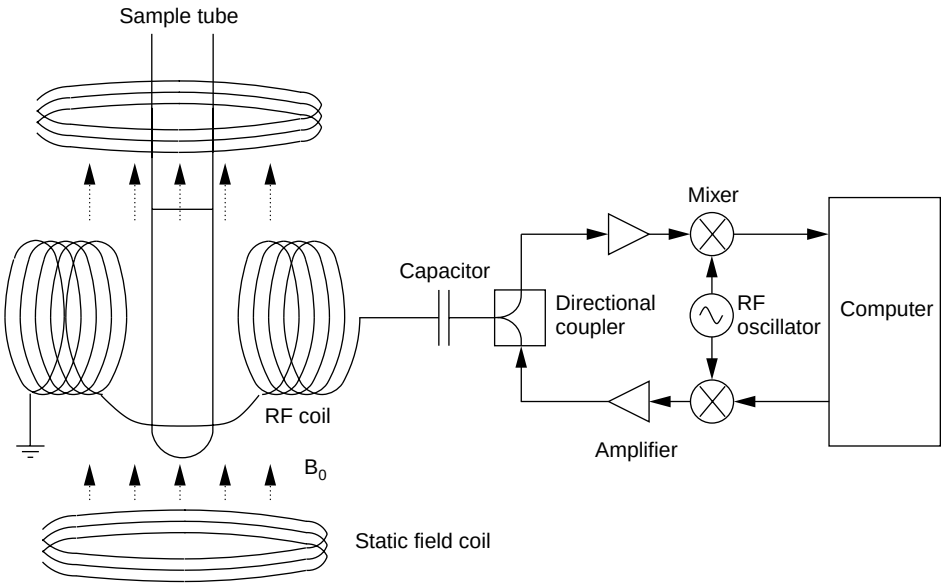

Figure 5. Schematic of an NMR spectrometer.

(along $-z$), a read-out pulse rotates the spin to $\pm x$. At this point the spins precess about the axis of the external magnetic field at a specific resonance frequency. These tiny precessing bar magnets produce a time-varying magnetic field in their spatial vicinity. This magnetic field can be measured, recorded, and Fourier transformed to obtain spectra that in turn can be integrated. With properly calibrated receiver phase settings, a positive integral indicates the spin was in $|0\rangle$, and a negative integral indicates the spin was in $|1\rangle$. By performing this procedure on all spins, we can learn the state of each spin and the result of a computation can thus be read out via the spectra.

Because of the J-coupling, extra information can be obtained from the spectrum of just a single spin. The spectrum of a spin coupled to m other spins will contain 2^m lines, at frequencies $\nu_i + \Sigma_j^m \pm J_{ij}/2$ where ν_i is the center frequency of the spectrum. We can assign each line to a specific state of the remaining m spins based on the values of the J-couplings. Depending on which of these lines we observe and whether it is up or down, we can tell the state of all spins just from one spectrum (assuming all couplings and lines are resolved).

Nuclear spins manipulated and read out using nuclear magnetic resonance thus, in principle, meet all the requirements for building a quantum computer.

Experimental apparatus

Figure 5 gives a schematic overview of what an NMR quantum computer looks like. As described previously, the heart of an NMR quantum computer is a molecule containing several atoms with a spin-1/2 nucleus. In practice, the signal from a single molecule is too weak for researchers to detect with current techniques. So, to boost the signal, we use about 10^{18} molecules. Each molecule in the ensemble acts as an individual quantum computer, and all 10^{18} quantum computers perform the same operations (thus, it is a SIMD machine, with no interprocessor communication). We dissolve the molecules in a liquid solvent such as acetone, ether, or chloroform. The liquid solution is held in a thin-walled glass sample tube of 5 mm in diameter, about 6 cm full.

We place the sample tube in the room-temperature bore of a superconducting solenoid. The solenoid is immersed in a bath of liquid helium, at a temperature of 4.2 degrees above absolute zero. The helium vessel is surrounded by a vacuum seal and a liquid nitrogen vessel. A persistent current of about 100 A through the solenoid's windings produces a magnetic field in the bore of more than 10 tesla (about 200,000 times the earth's magnetic field), resulting in typical spin resonance frequencies of 100 to 500 MHz. Strong fields are advantageous because the separation between the spectral lines of different nuclei (the chemical shift) increases linearly with the field strength.

A set of correction coils is mounted around the bore. By tuning the current through these coils, we can make variations in the strength of the static magnetic field smaller than 1 part in 10^9 over the sample volume. This extraordinary homogeneity ensures that the precession frequency of corresponding spins in different molecules varies by less than 1 Hz.

Next to the sample tube, saddle-shaped Helmholtz coils are mounted. The coils produce the pulsed electromagnetic fields necessary for single-qubit rotations. As for the static

field, good RF field homogeneity is crucial to rotate the spins in all the molecules over the same angle. The RF signals are generated by a very stable and accurate frequency source, then gated and phased-shifted in a transmitter board, and amplified in a power amplifier (50 to 300 W). The power amplifier output is coupled to the RF coils. Typical 90-degree pulses are between 10 μs and 10 ms in duration.

When several spins have comparable resonance frequencies, as in Figure 2, we use amplitude-shaped pulses to improve the pulses' frequency selectivity. We achieve this by dividing the pulse in short time slices (as short as 100 ns), and by changing the amplitude, slice by slice, to create the desired amplitude profile. Ideally, one transmitter channel is available for every spin, but we developed software solutions that let us use one transmitter channel to address several spins.

We also used the same RF coils to read out the spin states at the computation end. The oscillating magnetic signal produced by the spins induces an oscillating voltage in the coils. To maximize the coils' sensitivity, we incorporated them in a multiply resonant circuit tuned to the spin frequencies of interest. The induced voltage is sent through a preamplifier (approximately 0.8 dB input noise figure), mixed down to audio-frequencies, filtered, and digitized. Because this signal is very small (about –120 dBm) and immediately follows the high-power pulse, low-noise electronics and quiet amplifiers are essential.

The whole experiment is controlled by a Varian-designed system with an embedded Motorola Vx-Works processor. Pulse sequences in C and C++ compiled to low-level instruction for the spectrometer result in RF pulses and delay times. On the receiver end, the processor reads data from the digitizer then uploads it to a workstation for display and analysis.

Great care must be taken in selecting a suitable molecule. Since the duration of a single 2-qubit gate is on the order of $1/2J$, the J-couplings must be large compared to the decoherence rate, such that many operations can be completed within the coherence time. Furthermore, the chemical shifts must be large compared to the J-couplings, such that shaped pulses can be designed that cover the entire multiplet of one spin without affecting the multiplets of other spins. Reasonable chemical shifts are in the range of a few kHz to a few tens of kHz. Good values for J-couplings are a few tens to a few hundred Hz; logic gates thus last on the order of a few milliseconds to tens of milliseconds. Finally, reasonable values for the coherence times are tenths of seconds to several seconds.

The main experimental challenge is to achieve sufficient coherent control over the dynamics of a set of coupled nuclear spins. This involves molecule selection and synthesis, pulse shape design, pulse sequence design, and hardware configuration. Currently, it's possible to successfully concatenate tens to hundreds of logic gates on a handful of spins. Realistically, this is not enough to perform computations beyond the reach of classical computers. However, it does let us demonstrate the principles of quantum computation and acquire a practical understanding of what implementing quantum computers entails.

Quantum algorithm implementation

The invention of NMR QC led to the first-ever demonstrations of simple quantum computations and of quantum parallelism to solve certain problems in fewer steps than is possible classically.[11] However, an algorithm with the structure of Shor's algorithm has thus far remained beyond the reach of these small-scale realizations. This structure is common to all algorithms that achieve an exponential advantage compared to classical machines.

We've successfully implemented a generalization of Shor's algorithm, using NMR techniques, demonstrating for the first time the algorithm's two major components working together to find the order of a permutation.[6] This experiment was made possible by the synthesis of a five-spin molecule (Figure 2) with excellent spectral properties, and by the development of new methods and experimental techniques for initial state preparation and control over the dynamics of five spins.

Finding the order of a permutation π can be described as follows. Imagine 2^n rooms and one-way corridors connecting the rooms with exactly one entrance and one exit in each room. Note that a corridor may loop right back to the room it came from. This setup ensures that when making transitions from one room to the next, we eventually come back to the starting

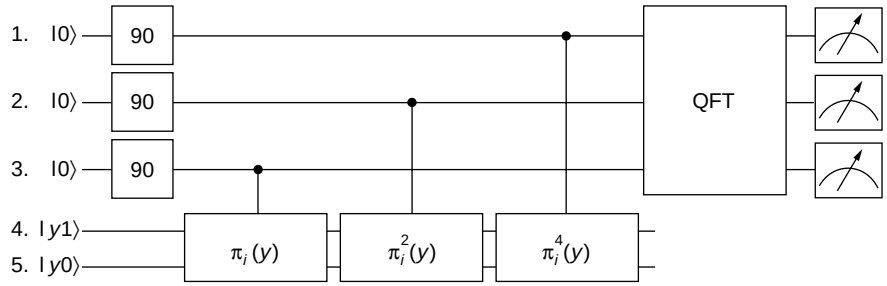

Figure 6. Quantum circuit implementing the order-finding algorithm. Each horizontal line corresponds to one qubit, and the boxes represent operations acting on the qubits (time goes from left to right). The vertical lines connecting a black dot and a box denotes that the box is to be executed if and only if the qubit indicated by the black dot is set to 1. The oracle call was implemented in three steps, using the fact that $\pi^x(y)=\pi^{x_0}\pi^{2x_1}\pi^{4x_2}$, in which $x_2\ x_1\ x_0$ is the binary representation of x.

room. We then define the order r as the minimum number of transitions needed to return to the starting room. This number generally depends on the starting room. The task is to determine r solely by trials of the type "make x transitions using π starting from room y and check which room you are in." Mathematically, such trials are described as queries of a black box or oracle that outputs $\pi^x(y)$.

We implemented the order-finding algorithm to determine the order of a representative subset of all 4! = 24 permutations on four input elements. Classically, the optimal way of finding the order of a permutation on four elements is to ask the oracle for the result of $\pi^3(y)$. If the answer is y, then the order r must be 1 or 3; otherwise, it must be 2 or 4. So, with a probability of 50%, we can guess the answer correctly after one query.

On a quantum computer running the order-finding quantum algorithm, we can guess the correct answer with a probability of 55% with only one oracle query because, in some sense, a quantum computer can make transitions to many rooms at once. (In the NMR apparatus, the probability is boosted to virtually 100% because of the large number of molecules in the sample.) While in this case the quantum computer advantage is only small, the gap between classical and quantum algorithms grows exponentially for increasing n.

The main steps in the algorithm are outlined in the quantum circuit of Figure 6. After the state initialization procedure, all qubits start off in the state $|0\rangle$. Qubits 1 through 3 are then rotated into an equal superposition of $|0\rangle$ and $|1\rangle$ by applying 90-degree pulses. Qubits 4 and 5 are set to y_1y_0, a binary representation of the starting room y. The next step is an oracle query of the type $\pi^x(y)$, where x is the state of the first register (which at this point is in a superposition of $|0\rangle$ through $|7\rangle$), and the output $\pi^x(y)$ will be stored in the second register.

The oracle call performing the permutation is implemented via a sequence of pulses and delay times, where the exact pulse sequence depends on π (systematic methods exist to design actual pulse sequences starting from a high-level description). Next, we apply the QFT on the first register, also by a sequence of pulses and delays. The final step is to acquire output spectra for the first three spins.

We custom-synthesized the molecule, shown in Figure 2, to have five specially placed fluorine spins with distinct frequencies, which served as the qubits. We hand-compiled the algorithm into pulse sequences involving between 50 and 200 RF pulses, for a total duration of 50 to 500 ms, depending on which permutation was implemented. Each building block in the pulse sequence was tested independently to confirm its proper operation.

Upon completion of the algorithm, we can immediately determine the order r of the permutation that was implemented by inspecting the output spectra. In fact, the spectrum of just spin 1 is sufficient to determine r. The theoretical prediction is that if $r = 1$, we should see only one (positive) line in the multiplet of spin 1; for $r = 2$, four lines should be visible; for r = 4, 16 lines should be visible; and for $r = 3$, some lines should be positive, some negative, and others dispersive, but the net area under the spectral lines should be zero.

Figure 7 shows this final experimental output, the spectra of spin 1. Aside from slight deviations from the ideal case we described (attributed to decoherence and imperfect pulses), the results are in excellent agreement with theoretical predictions.

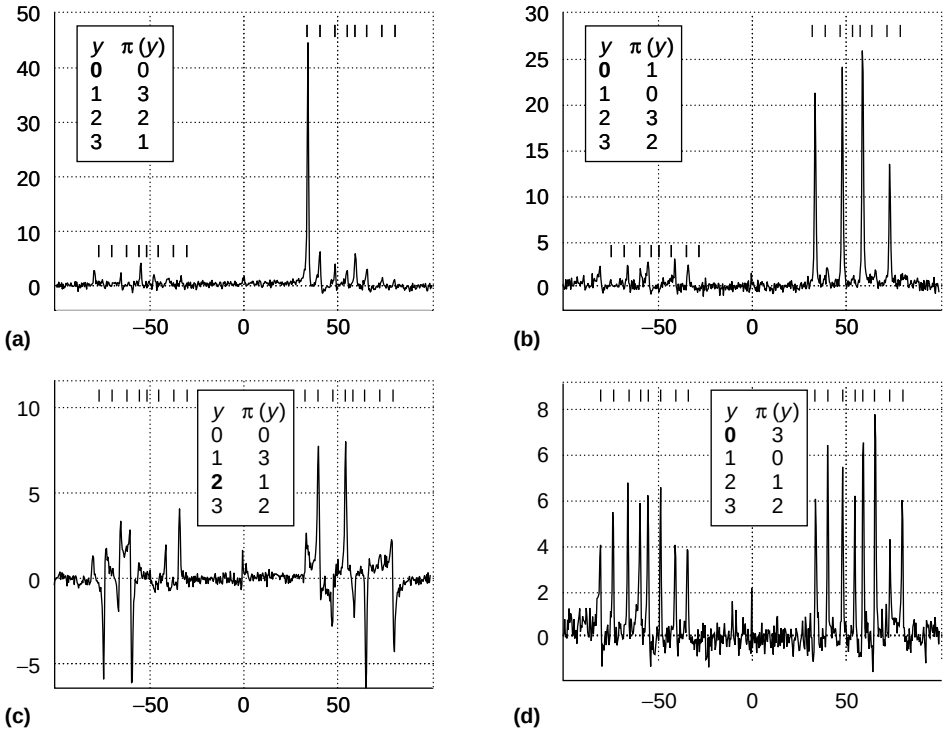

Figure 7. Experimentally measured spectra of spin 1, acquired after executing the order-finding algorithm, for four different cases. The respective permutations are shown in inset, with the starting element in bold. The markers above the spectra indicate the position of the lines in the full multiplet. *Copyright © 2000, The American Physical Society*

Future of quantum computing

Despite the promise of the small quantum computers implemented to date, extraordinary challenges remain to be solved before quantum processors become useful. The liquid-state NMR techniques we used in our experiments will work straightforwardly up to several tens of qubits (for certain applications) but are difficult to scale beyond several hundred. Meaningful quantum computation applications (known today) require thousands of qubits on a perfect machine, and millions if error correction is utilized to compensate for inevitable errors.[3] Solid-state technologies thus offer enormous promise for realizing large-scale quantum computers, leveraging current trends in nanoscale engineering. Molecular electronics, which uses nature's ability to chemically assemble functional nanometer-sized machines, may also be able to access the regime of quantum mechanics. The challenge is to engineer such systems to make them behave just like NMR systems: coherently and effectively as implementations of quantum circuits.

Hand in hand with the desire for large-scale quantum computers is finding uses for them other than factoring, search, and simulation. Information-theoretic tasks such as cryptography, distributed computation, and communication are key areas for quantum information science. Applications discovered so far include superdense coding (sending two bits with one qubit), state teleportation, and fast clock synchronization.[2] However, whether quantum-assisted protocols can be put to practical use remains to be seen.

Driving this field is a tremendous desire by researchers to understand how and why quantum resources can help information processing. Perhaps the most striking observation is the quantum computing community's consensus after over eight years' work in this area: It might be uncertain that a practical quantum computer will ever be built but, to our best knowledge, there are no principles of physics prohibiting large-scale quantum com-

puting. Thus, if we fail in implementing such machines, we stand to gain new understanding of fundamental physics. On the other hand, if we succeed, the foundations of computer science (namely, the modern version of Church's theorem)[2] must be overhauled to say that the complexity of a problem depends on the laws of physics. While the answers to these questions might lie still decades into the future, continued experimental progress in realizing small quantum processors will provide useful insight into the realm of quantum computing and give us a glimpse of what may lie ahead. MICRO

References

1. *International Technology Roadmap for Semiconductors*, Semiconductor Industry Assoc., San Jose, Calif., 1999.
2. M.A. Nielsen and I.L. Chuang, *Quantum Computation and Quantum Information*, Cambridge Press, Cambridge, UK, 2000.
3. C.H. Bennett and D.P. DiVincenzo, "Quantum Information and Computation," *Nature,* vol. 404, no. 6775, 2000, pp. 247-254.
4. N. Gershenfeld and I.L. Chuang, "Bulk Spin-Resonance Quantum Computing," *Science,* vol. 275, 1997, pp. 350-356.
5. D.G. Cory, A.F. Fahmy, and T.F. Havel, "Nuclear Magnetic Resonance Spectroscopy: An Experimentally Accessible Paradigm for Quantum Computing," *Proc.*, vol. 94, Nat'l Academy of Sciences, Washington, D.C., 1997, pp. 1634-1639.
6. L.M.K. Vandersypen et al., "Experimental Realization of an Order-Finding Algorithm with an NMR Quantum Computer" *Physical Review Letters*, vol. 85, no. 25, 2000, pp. 5452-5455.
7. P. Shor, "Algorithms for Quantum Computation: Discrete Logarithms and Factoring," *Proc. 35th Ann. Symp. Foundations of Computer Science*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1999, pp. 124-134.
8. A. Ekert and R. Jozsa, "Quantum Computation and Shor's Factoring Algorithm," *Reviews of Modern Physics,* vol. 68, no. 3, 1996, pp. 733-753.
9. L.K. Grover, "Quantum Computers Can Search Arbitrarily Large Databases by A Single Query," *Physical Review Letters,* vol. 79, no. 325, 1997, pp. 4709-4012.
10. R. Freeman, *Spin Choreography*, Spektrum, Oxford, UK, 1997.
11. J.A. Jones, "NMR Quantum Computing," to appear in *Progress in NMR Spectroscopy*; also available at http://xxx.lanl.gov/abs/quant-ph/0009002.

Matthias Steffen is a PhD candidate in the Electrical Engineering Department at Stanford University and a research scientist at the IBM Almaden Research Center in California. He received a BS in physics at Emory University. His research interests are in quantum information and experimental NMR techniques.

Lieven M.K. Vandersypen is a PhD candidate in electrical engineering at Stanford University as a Yansouni Family Stanford Graduate Fellow, and is a research scientist at the IBM Almaden Research Center. He received a BS/MS in mechanical engineering from the Katholieke Universiteit Leuven and an MS in electrical engineering from Stanford University as a Francqui Fellow of the Belgian American Educational Foundation. His research interests are in the physics of information and the coherent control of quantum systems.

Isaac L. Chuang is a research staff member at the IBM Almaden Research Center and a consulting professor in electrical engineering at Stanford University. Chuang received a PhD in electrical engineering from Stanford where he was a Hertz Foundation Fellow. He has been a visiting researcher at the Fujitsu Parallel Computing Research Center, Nippon Telephone and Telegraph Basic Research Laboratory, and the University of California at Santa Barbara's Institute for Theoretical Physics. He was also a postdoctoral fellow at Los Alamos National Laboratory and the University of California at Berkeley. In 1999, *MIT Technology Review* named Chuang one of the top 100 young innovators of the year. His research interests are quantum computation and quantum information.

Direct questions or comments about this article to Matthias Steffen at msteffen@snowmass.stanford.edu.