tion with O_2 and CO_2 and temperature control (Ruskinn Technologies, Leeds, UK). RCC4-VHL.HA, labeling conditions, and coimmunoprecipitation assays have been described previously (*11*). We used 12.5 μM MG132 for proteasomal inhibition. For hypoxic harvest, cells were lysed in the workstation with buffers deoxygenated in the chamber overnight. For standard harvest, the cells were removed from the chamber after hypoxic exposure, before cell lysis.
19. pcDNA3.VHL.HA and pcDNA3.HIF-1α were used to program TNT reticulocyte lysate (Promega). When programming in hypoxia, the reaction mix was preincubated in the workstation for 10 min before addition of the DNA template. An aliquot was removed from the workstation for transcription/translation under ambient oxygenation.
20. Proteins were expressed in rabbit reticulocyte or wheat germ IVTT systems (Promega), in insect cells with the use of a baculoviral system, and in bacteria. IVTTs were programmed with pcDNA3-based vectors encoding subdomains of HIF-1α or pVHL.HA as indicated. For recombinant baculoviral expression, Sf9 insect cells were infected with pFastBac1 vectors (GibcoBRL) encoding PK.HIF-1α(344-698) and HIF-1α(1-826).PK and harvested 60 hours after infection. In bacteria, pVHL was expressed as glutathione-*S*-transferase together with elongins B and C (GST-VBC complex) and HIF-1α as a maltose-binding protein fusion [pMAL.HIF-1α(344-698)].
21. pGal/HIF-1α549-582/VP16 was used to program reticulocyte lysate in the presence of unlabeled methionine. The fusion protein was immunoprecipitated with beads precoated with anti-Gal4 RK5C1 (Santa Cruz). Immunoprecipitated HIF-1α fusion was washed with NETN buffer [50 mM tris (pH 7.5), 150 mM NaCl, 0.5 mM EDTA, and 0.5% NP-40], and the beads were incubated with cell lysate in hypotonic extraction buffer [HEB: 20 mM tris (pH 7.5), 5 mM KCl, 1.5 mM $MgCl_2$, 1 mM dithiothreitol] for 60 min at 22°C. The beads were then washed with NETN containing DFO (100 μM) and incubated for 2 hours at 4°C with ^{35}S-methionine–labeled pVHL.HA.
22. HIF-1α(1-826).PK or PK.HIF-1α(344-698) produced from baculoviruses was immunoprecipitated with anti-PK (Serotec). Bead-bound immunoprecipitates were incubated under test conditions and assayed for pVHL.HA capture.
23. D. R. Mole, data not shown.
24. For peptide-blocking assays, peptides were preincubated in cell extract or other conditions for 60 min at 30°C and then added (final concentration, 1 μM) to NETN buffer containing a mixture of HIF-1α and pVHL.HA.
25. Samples for mass spectroscopic analyses were biotinylated synthetic peptides 19:WT (HIF-1α residues 556 to 574) or 34:WT (HIF-1α residues 549 to 582) or baculoviral PK-tagged HIF-1α. After modification by mammalian cell lysates, the material was purified by streptavidin/biotin capture (synthetic peptides) or anti-PK immunoprecipitation and SDS–polyacrylamide gel electrophoresis (SDS-PAGE) (baculoviral HIF). Proteolytic digestion was performed on the beads (synthetic peptides) or in-gel with trypsin and V8 protease at pH 7.8 or V8 protease at pH 4.5. Samples were lyophilized and dissolved in aqueous 0.1% trifluoroacetic acid. Peptides were concentrated, desalted on a 300-μm inside diameter/5-mm length C18 PepMap column (LC Packings, San Francisco, CA), and eluted with 80% acetonitrile. The high-performance liquid chromatography (CapLC, Waters, Milford, MA) was coupled through a Nano-LC inlet to a Q-TOF mass spectrometer (Micromass, Manchester, UK) equipped with a nanoelectrospray Z-spray source. The eluted peptide mixture was analyzed by tandem mass spectrometric sequencing with an automated MS-to-MS/MS switching protocol. Online determination of precursor-ion masses was performed over the *m/z* range from 300 to 1200 atomic mass units in the positive charge detection mode with a cone voltage of 30 V. The collision-induced dissociation for peptide sequencing by MS/MS was performed with argon gas at 20 to 40 eV and a three-dimensional quadrupole resolution.
26. K. I. Kivirikko, R. Myllylä, in *The Enzymology of Post-translational Modification of Proteins*, R. B. Freeman, H. C. Hawkins, Eds. (Academic Press, London, 1980), pp. 53–102.
27. In pVHL capture assays with biotinylated peptides, the peptide was preincubated with cell extract or buffer for 30 min at 30°C and then assayed for ability to bind ^{35}S-labeled pVHL.HA.
28. K. I. Kivirikko, J. Myllyharju, *Matrix Biol.* **16**, 357 (1998).
29. C. J. Schofield, Z. Zhang, *Curr. Opin. Struct. Biol.* **9**, 722 (1999).
30. Single-letter abbreviations for the amino acid residues are as follows: A, Ala; C, Cys; D, Asp; E, Glu; F, Phe; G, Gly; H, His; I, Ile; K, Lys; L, Leu; M, Met; N, Asn; P, Pro; Q, Gln; R, Arg; S, Ser; T, Thr; V, Val; W, Trp; X, any amino acid; and Y, Tyr.
31. C. J. Cunliffe, T. J. Franklin, N. J. Hales, G. B. Hill, *J. Med. Chem.* **35**, 2652 (1992).
32. G. A. Jansen *et al.*, *J. Lipid Res.* **40**, 2244 (1999).
33. M. Mukherji, M. D. Lloyd, unpublished results.
34. Y.-M. Tian, M. Mukherji, data not shown.
35. Prolyl 4-hydroxylase activity was assayed by a method based on the hydroxylation-coupled decarboxylation of 2-oxo[1-^{14}C]glutarate [K. I. Kivirikko, R. Myllylä, *Methods Enzymol.* **82**, 245 (1982)] with recombinant human type I and II prolyl 4-hydroxylases expressed in insect cells. Each reaction contained 0.5 or 1.0 mg of peptide.
36. W. Ehleben, T. Porwol, J. Fandrey, W. Kummer, H. Acker, *Kidney Int.* **51**, 483 (1997).
37. E. E. Patton, A. R. Willems, M. Tyers, *Trends Genet.* **14**, 236 (1998).
38. D. Skowyra, K. L. Craig, M. Tyers, S. J. Elledge, J. W. Harper, *Cell* **91**, 209 (1997).
39. P. W. Conrad, T. L. Freeman, D. Beitner-Johnson, D. E. Millhorn, *J. Biol. Chem.* **274**, 33709 (1999).
40. D. E. Richard, E. Berra, E. Gothie, D. Roux, J. Pouysségur, *J. Biol. Chem.* **274**, 32631 (1999).
41. N. C. Bacon *et al.*, *Biochem. Biophys. Res. Commun.* **249**, 811 (1998).
42. N. Masson, unpublished results.
43. M. Ivan *et al.*, *Science* **292**, 464 (2001).
44. M. Rechsteiner, S. W. Rogers, *Trends Biol. Sci.* **21**, 267 (1996).
45. P. Jaakkola *et al.*, data not shown.
46. Supported by grants from the Wellcome Trust and the Medical Research Council. P.J. is Junior Research Fellow of the Academy of Finland. We thank J. Myllyharju for performing the prolyl-4-hydroxylase assay; K. Brindle, R. Hider, K. Kivirikko, E. Maher, and R. Wolfe for advice; N. Pavletich for GST-VCB expression constructs; and E. Gibson for synthesis of 2-oxoglutarate analogs.

12 February 2001; accepted 19 March 2001
Published online 5 April 2001;
10.1126/science.1059796
Include this information when citing this paper.

REPORTS

A Quantum Adiabatic Evolution Algorithm Applied to Random Instances of an NP-Complete Problem

Edward Farhi,[1]* Jeffrey Goldstone,[1] Sam Gutmann,[2] Joshua Lapan,[3] Andrew Lundgren,[3] Daniel Preda[3]

A quantum system will stay near its instantaneous ground state if the Hamiltonian that governs its evolution varies slowly enough. This quantum adiabatic behavior is the basis of a new class of algorithms for quantum computing. We tested one such algorithm by applying it to randomly generated hard instances of an NP-complete problem. For the small examples that we could simulate, the quantum adiabatic algorithm worked well, providing evidence that quantum computers (if large ones can be built) may be able to outperform ordinary computers on hard sets of instances of NP-complete problems.

Although a large quantum computer has yet to be built, the rules for programming such a device, which are derived from the laws of quantum mechanics, are well established. It is already known that quantum computers could solve problems believed to be intractable on classical (i.e., nonquantum) computers. An intractable problem is one that necessarily takes too long to solve when the input gets too big. More precisely, a classically intractable problem is one that cannot be solved using any classical algorithm whose running time grows only polynomially as a function of the length of the input. For example, all known classical factoring algorithms require a time that grows faster than any polynomial as a function of the number of digits in the integer to be factored. Shor's quantum algorithm for the factoring problem (*1*) can factor an integer in a time that grows (roughly) as the square of the number of digits. This raises the question of whether quantum computers could solve other classically difficult prob-

[1]Center for Theoretical Physics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA. [2]Department of Mathematics, Northeastern University, Boston, MA 02115, USA. [3]Massachusetts Institute of Technology, Cambridge, MA 02139, USA.

*To whom correspondence should be addressed. E-mail: farhi@mit.edu

lems faster than classical computers.

Beyond factoring, there is a famous collection of problems called NP-complete [see, for example, (*2*)]. Hundreds of problems are known to be NP-complete—for example, (a variant of) the Traveling Salesman problem—and they are all related in the following sense: If someone finds a polynomial-time algorithm for one NP-complete problem, then this algorithm could be used as a subroutine in programs that would then solve all other NP-complete problems in polynomial time. That no one has succeeded in finding a classical polynomial-time algorithm for any of these problems is strong evidence for the intractability of all of them. On the other hand, no one has been able to prove that a polynomial-time algorithm cannot be constructed for any NP-complete problem. Settling the question of whether a polynomial-time algorithm does or does not exist for an NP-complete problem is one of the outstanding problems of classical computer science. It is also an open question whether an NP-complete problem could be solved in polynomial time on a quantum computer.

Here, we describe a recently developed approach to quantum computation based on quantum adiabatic evolution [(*3*); for related ideas, see (*4*)]. We apply the quantum adiabatic algorithm to a specific NP-complete problem, Exact Cover. A decisive mathematical analysis of this quantum adiabatic evolution algorithm has not been possible. Instead, we resort to numerical simulation of the running of the quantum algorithm (*5*). Each time we do the simulation, we use as input a randomly generated instance of Exact Cover. The lengths of these inputs are necessarily small because simulating a quantum computer on a classical computer requires memory that grows exponentially in the length of the input. On these small inputs our data look promising. For our randomly generated instances of Exact Cover, we find that the quantum algorithm succeeds in a time that grows only quadratically in the length of the input.

Saying that an algorithm solves a problem in polynomial time means that the algorithm succeeds in polynomial time on every possible input. On the other hand, an algorithm may succeed in polynomial time on a large set of inputs but not on all. This has led to efforts to identify sets of instances that are hard for particular classical algorithms. Researchers working on the NP-complete problem 3-SAT (Three-Satisfiability) have identified a set of instances that are hard for standard classical search algorithms [(*6*); see also (*7*) and references therein]. Although the quantum adiabatic evolution algorithm could be applied to 3-SAT, we find it more convenient to study Exact Cover. Our instances of Exact Cover are generated from a set that we believe to be classically intractable for sufficiently large inputs. Using a running time that grows only quadratically in the length of the input, the quantum adiabatic algorithm solves the Exact Cover instances we randomly generated. Again, because of the space requirements inherent in simulating a quantum computer, these instances are necessarily small. However, if classical algorithms indeed require exponential time on this set and the quantum quadratic behavior actually persists for large instances, then quantum computers could outperform classical computers on randomly generated hard instances, although not necessarily in the worst case.

We now define the NP-complete problem Exact Cover [see, for example, (*2*)]. Consider n bits $z_1, z_2, \ldots, z_n$ each of which can take the value 0 or 1. An n-bit instance of Exact Cover is built up from clauses, each of which is a constraint imposed on the values of three of the bits. If a given clause involves the three bits labeled i, j, and k, then the constraint is that one of the three bits must have the value 1 and the other two must have the value 0. An n-bit instance of Exact Cover is a list of triples (i, j, k) indicating which groups of three bits are involved in clauses. The problem is to determine whether there is some assignment of the n-bit values that satisfies all of the clauses. Given an assignment of values for $z_1, z_2, \ldots, z_n$, we can easily check whether the assignment satisfies all of the clauses. But determining whether at least one of the 2^n assignments of $z_1, z_2, \ldots, z_n$ satisfies all the clauses is in fact an NP-complete problem.

All quantum systems evolve in time according to the Schrödinger equation

$$i\frac{d}{dt}|\psi(t)\rangle = H(t)|\psi(t)\rangle \qquad (1)$$

where $|\psi(t)\rangle$ is the time-dependent state vector and $H(t)$ is the time-dependent Hamiltonian operator. A quantum computer algorithm can be viewed as a specification of a Hamiltonian $H(t)$ and an initial state $|\psi(0)\rangle$. These are chosen so that the state at time T, $|\psi(T)\rangle$, encodes the answer to the problem at hand.

In designing our quantum algorithm we take advantage of the quantum adiabatic theorem, which we now explain. At time t, the Hamiltonian $H(t)$ has an instantaneous ground state $|\psi_g(t)\rangle$, which is the eigenstate of $H(t)$ with the lowest energy. Adiabatic evolution refers to the situation where $H(t)$ is slowly varying. Suppose the quantum system starts at $t = 0$ in the ground state of $H(0)$, that is, $|\psi_g(0)\rangle$. The adiabatic theorem says that if $H(t)$ varies slowly enough, then the evolving state vector $|\psi(t)\rangle$ will remain close to the instantaneous ground state $|\psi_g(t)\rangle$. [For a more precise discussion of the adiabatic theorem, see (*3*).]

To specify our algorithm we must give $H(t)$ for $0 \leq t \leq T$, where T is the running time of the algorithm. We choose $H(t)$ so that the ground state of $H(0)$ is known in advance and is easy to construct. For any instance of the problem under study (Exact Cover in this case), there is a Hamiltonian, H_P, whose ground state encodes the solution. Although it is straightforward to construct H_P, finding its ground state is computationally difficult. We take $H(T) = H_P$, which means that $|\psi_g(T)\rangle$ encodes the solution. For intermediate times, $H(t)$ smoothly interpolates between $H(0)$ and $H(T) = H_P$, say by taking

$$H(t) = \left(1 - \frac{t}{T}\right) H(0) + \frac{t}{T} H_P \qquad (2)$$

We start with the quantum system in the known ground state of $H(0)$. If the running time T is large enough, $H(t)$ will indeed be slowly varying, and by the adiabatic theorem the final state reached, $|\psi(T)\rangle$, will be close to $|\psi_g(T)\rangle$.

The state vector of the quantum computer evolves in a Hilbert space of dimension 2^n. We take as a basis the 2^n vectors

$$|z_1\rangle|z_2\rangle \ldots |z_n\rangle \qquad (3)$$

where each $z_i = 0$ or 1. This n-qubit Hilbert space can be realized as a system of n spin-½ particles, where $|z_i = 0\rangle$ corresponds to the ith spin being up in the z-direction and $|z_i = 1\rangle$ corresponds to spin down in the z-direction. For $H(0)$ we couple a magnetic field in the x-direction to each quantum spin. [Specifically, the strength of the field at each site is equal to the number of clauses that contain the bit. Thus, $H(0)$ is instance-dependent; see (*3*).] The ground state of the ith qubit corresponding to spin aligned in the x-direction is

$$\frac{1}{\sqrt{2}}(|z_i = 0\rangle + |z_i = 1\rangle) \qquad (4)$$

The ground state of $H(0)$ for the n-qubit quantum system is therefore

$$|\psi_g(0)\rangle = \frac{1}{2^{n/2}} \sum |z_1\rangle |z_2\rangle \ldots |z_n\rangle \qquad (5)$$

where the sum is over all 2^n basis vectors. This means that $|\psi_g(0)\rangle$, which we take to be the starting state of our algorithm, is a uniform superposition of states corresponding to all possible assignments of bit values.

To define $H_P = H(T)$, we first define a classical energy function $h(z_1, z_2, \ldots, z_n)$ that is a sum of energy functions $h_C(z_i, z_j, z_k)$ where i, j, and k are the labels of the bits involved in clause C. It costs energy to violate clause C:

$$h_C(z_i, z_j, z_k) = \begin{cases} 0 & \text{clause } C \text{ satisfied} \\ 1 & \text{clause } C \text{ violated} \end{cases} \qquad (6)$$

Now let

$$h = \sum_C h_C \qquad (7)$$

which for any bit assignment $z_1, z_2, \ldots, z_n$ is

equal to the number of clauses that the assignment violates. We turn this classical energy function into a quantum operator, diagonal in the z-basis:

$$H_P|z_1\rangle|z_2\rangle \ldots |z_n\rangle = h(z_1, z_2, \ldots, z_n)|z_1\rangle|z_2\rangle \ldots |z_n\rangle \quad (8)$$

This means that the ground state of H_P corresponds to the bit assignment that violates the minimal number of clauses. (If more than one assignment minimizes the number of violations, then there will be more than one ground state of H_P.) The problem of Exact Cover is to determine whether a given instance (specified by a set of clauses) has an assignment that violates no clauses. If we can produce the ground state of H_P, we can solve the computational problem.

At time T, we measure the state $|\psi(T)\rangle$ in the basis of (Eq. 3). This will produce a string $z_1, z_2, \ldots, z_n$ of 0's and 1's. This string can be quickly checked to see whether it satisfies all of the clauses. Note that if we run the quantum algorithm again [with the same instance, starting state $|\psi(0)\rangle$ and running time T] we end up in the same quantum state $|\psi(T)\rangle$. The probability of obtaining a satisfying assignment depends only on $|\psi(T)\rangle$.

The adiabatic theorem ensures that the quantum adiabatic evolution algorithm will produce the desired state that encodes the solution to the instance of Exact Cover if the running time T is long enough. Determining how long is long enough to produce a reasonably large success probability is the key to determining the potential usefulness of the algorithm. For certain specialized examples, we know that the required running time grows only as a polynomial in the number of bits (*3*). But addressing the general case of all instances of Exact Cover is beyond our analytical abilities. Here we report on a numerical study of the running time needed to solve a randomly generated set of Exact Cover instances.

To simulate the quantum computer, we numerically integrate the Schrödinger equation (Eq. 1). For an n-qubit quantum system, the state vector $|\psi(t)\rangle$ has 2^n complex components. For $n = 20$, this means numerically integrating a differential equation with 2,097,152 real variables. This is as large a system as we could handle with our computer resources in a few months of running.

Because the number of bits in our instances of Exact Cover is never more than 20, we can always determine whether the instance has satisfying assignments and what they are. We do this by exhaustively checking all of the bit assignments, which takes virtually no time on a classical computer. The present report concerns randomly generated instances of Exact Cover with a unique satisfying assignment (USA) because we believe that these are the most difficult for the quantum algorithm. We have evidence that the quantum algorithm runs faster on instances with more than one satisfying assignment (*8*). We also have evidence that the quantum algorithm works well on instances with no satisfying assignment, in which case the algorithm produces an assignment that violates the minimal number of clauses. Thus, the restriction to a USA appears to restrict us to the most difficult cases for the quantum algorithm.

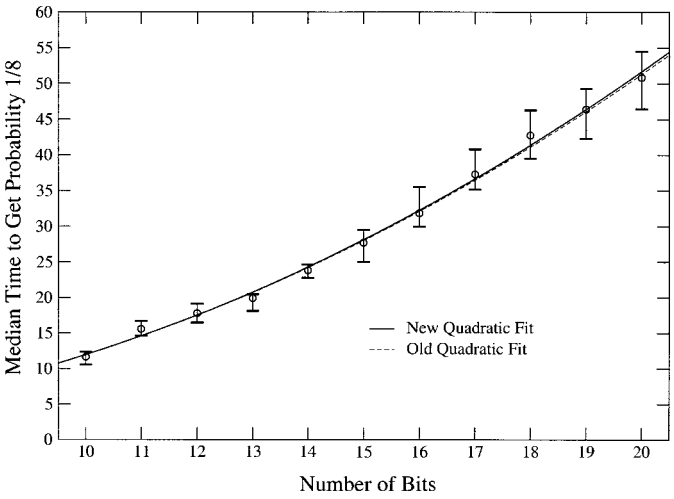

Fig. 1. Each circle is the median time to achieve a success probability of 1/8 for 75 USA instances. The error bars give 95% confidence limits for each median. The solid line is a quadratic fit to the data. The broken line, which lies just below the solid line, is the quadratic fit obtained in (*5*) for an independent data set up to 15 bits.

With the number of bits fixed to be n, we generate USA instances of Exact Cover as follows. We pick three distinct bits at random, uniformly over the integers from 1 to n. We then have a formula with one Exact Cover clause. We add clauses one at a time by picking new sets of three bits. After each clause is added we calculate the number of satisfying assignments, which always decreases (or stays the same). If the number of satisfying assignments is reduced to just 1, we stop and accept the instance. If the number of satisfying assignments drops from more than 1 to 0 without hitting 1, we reject the instance and start again. With this procedure, the number of clauses is not a fixed function of the number of bits, but rather varies from instance to instance. These instances, on average, have about as many clauses as bits.

By the adiabatic theorem, the probability of finding the satisfying assignment approaches 1 as the running time approaches infinity. We are, of course, forced to settle for a finite running time and a probability less than 1. We have (somewhat arbitrarily) picked a success probability of 1/8, which, for $n \geq 10$, is much larger than $1/2^n$, the probability that a random bit assignment is the satisfying assignment. For each number of bits n between 10 and 20, we generated 75 USA instances of Exact Cover. For each value of n, we determined the median time required to achieve a success probability of 1/8. (Because this is a numerical study, we actually hunted for a time that gives a probability between 0.12 and 0.13.)

In Fig. 1, the circles represent the median time to achieve probability 1/8 for $10 \leq n \leq 20$. The error bars give 95% confidence limits on the medians. The solid line is a quadratic fit to the data. In (*5*) we obtained corresponding data for $7 \leq n \leq 15$, and the dashed line is the quadratic fit to those data. The limited power of classical computers makes it impractical to go even a few bits beyond 20, so further numerical study will not decisively determine how the median running time grows with the number of bits. However, it is possible that the data up to 20 bits already reveal the asymptotic performance of the algorithm.

To specify the algorithm, we want a running time, set in advance, that depends only on the number of bits and not on the instance being considered. We propose running the quantum algorithm for a time $T = T(n)$ that is equal to the quadratic fit to the median time required to achieve probability 1/8, the solid curve shown in Fig. 1. To test the algorithm at the proposed running time, we generated 100 new USA instances of Exact Cover, at each value of n between 10 and 20, and ran the simulation on each instance with $T = T(n)$. In Fig. 2, the circles show the median probability of success at each n. Not surprisingly, these are close to 1/8. We also show the 10th-worst and worst probability for each n. The good news for the quantum algorithm is that these do not appear to decrease appreciably with n.

We also generated 1000 new USA instances of Exact Cover at both 16 and 17 bits. Figure 3 shows the histograms of the success probability when the instances are run at $T(16)$ and $T(17)$, respectively. The histograms indicate that a USA instance with a success probability below 0.04 is very unlikely to be generated.

If an algorithm (classical or quantum) succeeds with probability at least p, then running the algorithm k times gives a success probability of at least $1 - (1 - p)^k$. For example, if $p = 0.04$, then 200 repetitions of the algo-

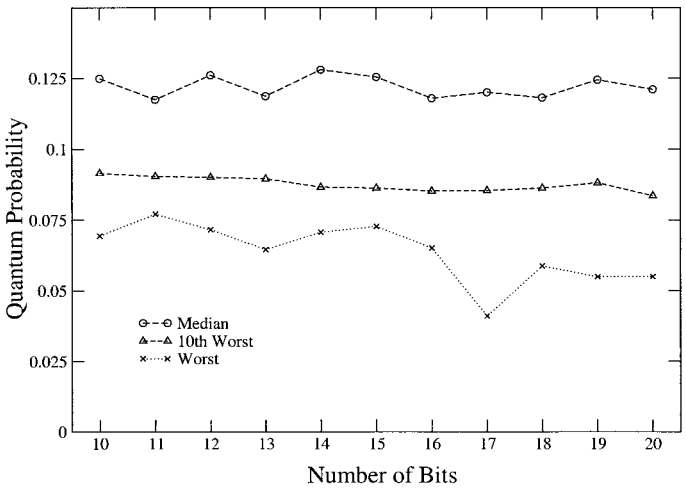

Fig. 2 (left). Each circle is the median probability of success for 100 USA instances with the algorithm run at the solid-line quadratic fit shown in Fig. 1. Each triangle is the 10th-lowest probability, and each x is the lowest probability. Note that the lowest probabilities do not decrease much as the number of bits n increases.

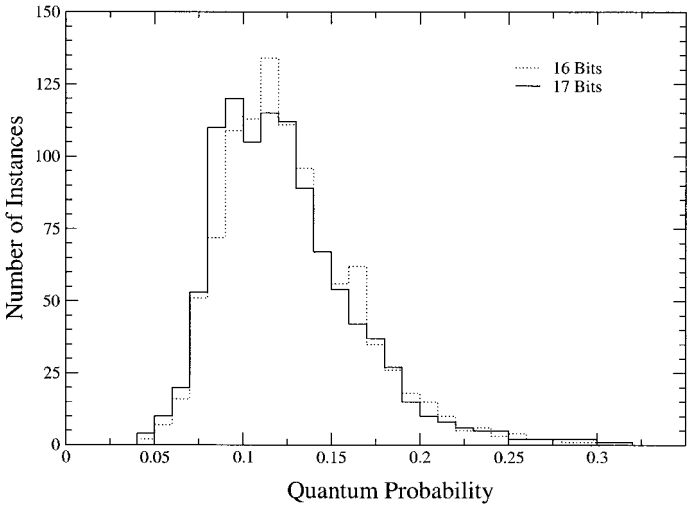

Fig. 3 (right). Histograms of success probability for 1000 USA instances at 16 bits and another 1000 instances at 17 bits. The running times are given by the solid-line quadratic fit in Fig. 1 at $n = 16$ and $n = 17$.

rithm gives a success probability of better than 0.9997. Suppose that as the number of bits increases, it remains true that almost all USA instances (generated as described above) have a success probability of at least 0.04 at the quadratic running time $T(n)$. Then any n-independent desired probability of success can be achieved with a fixed number of repetitions.

If the behavior we have seen up to 20 bits persists for all values of n, then we have identified a set of instances on which the adiabatic algorithm performs well. There is also evidence that this set is hard for classical algorithms. Most of the instances we have generated lie near the "phase transition" for Exact Cover. The phase transition region consists of instances with the number of clauses chosen so that half of the instances have one or more satisfying assignments. For 3-SAT, an NP-complete problem closely related to Exact Cover, there is evidence that the hard instances for classical algorithms are located at the phase transition (*7*).

In previous work (*3*), quantum adiabatic evolution was studied analytically on certain sequences of instances of Satisfiability where the clauses involve at most two bits. Each of these sequences has enough structure to make it possible to determine the required running time for any number of bits. For each case considered, the quantum adiabatic algorithm succeeds in a time that grows only polynomially in the number of bits. Of course, the structure of those instances makes it possible to determine the satisfying assignment by inspection, so these instances are certainly easy for some classical algorithms.

In (*3*) and (*5*) the quantum adiabatic algorithm was also applied to the problem of unstructured search (*9*). This problem can be cast as a restricted form of Satisfiability where each instance has a single clause that involves all of the bits and determines a USA. In this case the required running time is provably exponential in the number of bits even in the quantum case (*10*). This exponential behavior is indeed clearly seen in the data out to 14 bits in the numerical simulation of quantum adiabatic evolution presented in (*5*).

We have also been looking for a structured sequence of instances of Exact Cover that may be difficult for the quantum adiabatic algorithm. We have a candidate sequence where the success probabilities drop sharply as a function of the number of bits when the algorithm is run at the quadratic fit shown in Fig. 1. We have experimented with ad hoc modifications of the quantum algorithm that increase the success probability for this sequence. In any case, sequences of structured instances have little bearing on the performance of the quantum algorithm on randomly generated sets, but are relevant to discussions of whether this algorithm (or a modified version) could solve an NP-complete problem outright.

The quantum adiabatic evolution algorithm operates in continuous time by evolving a quantum state according to the Schrödinger equation (Eq. 1). In the conventional quantum computing paradigm, an algorithm consists of a sequence of discrete unitary transformations. Although the adiabatic time evolution can be well approximated by a sequence of discrete unitary steps (*3*), we see no advantage in this reformulation. In fact, continuous time evolution may offer an alternative model for the design of a quantum computer.

Quantum computation by adiabatic evolution works by keeping the quantum state close to the instantaneous ground state of the Hamiltonian that governs the evolution. This suggests that a device running the quantum adiabatic algorithm should be kept at a low temperature to reduce unwanted transitions out of the ground state. Conventional quantum computing does not take place in the ground state, and decohering transitions caused by interactions with the environment are a major impediment to current efforts to build a large-scale quantum computer. The quantum adiabatic algorithm running on a cold device may be more fault tolerant than the implementations of discrete-step quantum computation usually envisioned.

Quantum computation by adiabatic evolution applied to a wide variety of combinatorial search problems [see, for example, (*11*)] will succeed if the running time is long enough. We have seen evidence that for our randomly generated small instances of Exact Cover, the required running time grows slowly as a function of the number of bits. It is possible that the slow growth we have already seen indicates the true asymptotic behavior of the algorithm when applied to randomly generated hard instances of Exact Cover. If this quantum adiabatic algorithm does actually outperform known classical algorithms, we would have reason to eagerly await the construction of a quantum computer capable of running it.

References and Notes

1. P. W. Shor, *SIAM J. Comput.* **26**, 1484 (1997).
2. D. S. Johnson, C. H. Papadimitriou, in *The Traveling Salesman Problem,* E. L. Lawler, J. K. Lenstra, A. H. G. Rinnooykan, D. B. Shmays, Eds. (Wiley, New York, 1985), p. 37.
3. E. Farhi, J. Goldstone, S. Gutmann, M. Sipser, http://xxx.lanl.gov/abs/quant-ph/0001106.
4. T. Kadowaki, H. Nishimori, *Phys. Rev. E* **58**, 5355 (1998).
5. E. Farhi, J. Goldstone, S. Gutmann, http://xxx.lanl.gov/abs/quant-ph/0007071.
6. D. G. Mitchell, B. Selman, H. J. Levesque, in *Proceedings of the 10th National Conference on Artificial Intelligence* (American Association for Artificial Intelligence, Menlo Park, CA, 1992), p. 459.

7. S. Kirkpatrick, B. Selman, *Science* **264**, 1297 (1994).
8. E. Farhi *et al.*, unpublished data.
9. L. K. Grover, *Phys. Rev. Lett.* **78**, 325 (1997).
10. E. Farhi, S. Gutmann, *Phys. Rev. A* **57**, 2403 (1998); C. H. Bennett, E. Bernstein, G. Brassard, U. V. Vazirani, *SIAM J. Comput.* **26**, 1510 (1997) (available at http://xxx.lanl.gov/abs/quant-ph/9701001).
11. A. M. Childs, E. Farhi, J. Goldstone, S. Gutmann, http://xxx.lanl.gov/abs/quant-ph/0012104.
12. Supported in part by the U.S. Department of Energy under cooperative agreement DE-FC02-94ER40818 and also by MIT's Undergraduate Research Opportunities Program. We thank the Computer Facility of the MIT Laboratory for Nuclear Science for extensive use of the computer Abacus. We benefited greatly from conversations with A. Childs, B. Selman, and L. Valiant. We also thank D. Fisher, B. Halperin, M. Kardar, and P. Lee for useful discussions about the connection of our work to statistical mechanics systems.

27 November 2000; accepted 19 March 2001

Observation of Vortex Lattices in Bose-Einstein Condensates

J. R. Abo-Shaeer, C. Raman, J. M. Vogels, W. Ketterle

Quantized vortices play a key role in superfluidity and superconductivity. We have observed the formation of highly ordered vortex lattices in a rotating Bose-condensed gas. These triangular lattices contained over 100 vortices with lifetimes of several seconds. Individual vortices persisted up to 40 seconds. The lattices could be generated over a wide range of rotation frequencies and trap geometries, shedding light on the formation process. Our observation of dislocations, irregular structure, and dynamics indicates that gaseous Bose-Einstein condensates may be a model system for the study of vortex matter.

The quantization of circulation has a profound effect on the behavior of macroscopic quantum systems. Magnetic fields can penetrate type-II superconductors only as quantized flux lines. Vorticity can enter rotating superfluids only in the form of discrete line defects with quantized circulation. These phenomena are direct consequences of the existence of a macroscopic wavefunction, the phase of which must change by integer multiples of 2π around magnetic flux or vortex lines. In superconductors, magnetic flux lines arrange themselves in regular lattices that have been directly imaged (*1*). In superfluids, direct observation of vortices has been limited to small arrays (up to 11 vortices), both in liquid ^{4}He (*2*) and, more recently, in rotating gaseous Bose-Einstein condensates (BECs) (*3*, *4*).

We report the observation of vortex lattices in a BEC. We are now able to explore the properties of bulk vortex matter, which includes local structure, defects, and long-range order. In contrast, the properties of small arrays are strongly affected by surface and finite size effects. The vortex lattices are highly excited collective states of BECs with an angular momentum of up to 60 $\hbar$ per particle. Our experiments show that such states can be prepared and are much more stable than predicted (*5*).

Vortices in BECs have been the subject of extensive theoretical study (*6*). Experimental progress began only recently with the observation of quantized circulation in a two-component condensate by a phase engineering technique (*7*) and of vortex arrays in a single-component BEC (*3*). A condensate can be subjected to a rotating perturbation by revolving laser beams around it. This technique was used to study surface waves in a trapped BEC (*8*), and subsequently for the creation of vortices (*3*). In 1997, we tried unsuccessfully to detect quantized circulation as a "centrifugal hole" in ballistic expansion of the gas (*9*, *10*). Theoretical calculations (*11*–*13*) and ultimately the pioneering experimental work (*3*) showed that vortices can indeed be detected through ballistic expansion, which magnifies the spatial structure of the trapped condensate.

BECs of up to 5×10^7 Na atoms with a negligible thermal component (condensate fraction $\geq$ 90%) were produced by a combination of laser and evaporative cooling techniques (*8*, *10*, *14*). A radio-frequency "shield" limited the magnetic trap depth to 50 kHz (2.3 μK), preventing high-energy atoms from heating the condensate. Experiments were performed in cylindrical traps with widely varying aspect ratios. Most of the results and all of the images were obtained in a weak trap, with radial and axial frequencies of $\nu_r = 84$ Hz and $\nu_z = 20$ Hz (aspect ratio 4.2), respectively. In this weak trap inelastic losses were suppressed, resulting in larger condensates of typically 5×10^7 atoms. Such clouds had a chemical potential (μ) of 310 nK (determined from time-of-flight imaging), a peak density of 4.3×10^{14} cm^{-3}, a Thomas-Fermi radius along the radial direction (R_r) of 29 μm, and a healing length (ξ) of about 0.2 μm.

Vortex lattices were produced by rotating the condensate around its long axis with the optical dipole force exerted by blue-detuned laser beams at a wavelength of 532 nm. A two-axis acousto-optic deflector generated a pattern of two laser beams rotating symmetrically around the condensate at variable drive frequency Ω (*8*). The two beams were separated by one Gaussian beam waist (w = 25 μm). The laser power of 0.7 mW in each beam corresponded to an optical dipole potential of 115 nK. This yielded a strong, anharmonic deformation of the condensate.

After the condensate was produced, the stirring beam power was ramped up over 20 ms, held constant for a variable stirring time, and then ramped down to zero over 20 ms. The condensate equilibrated in the magnetic trap for a variable hold time (typically 500 ms). The trap was then suddenly switched off, and the gas expanded for 35 ms to radial and axial sizes of $l_r \cong 1000$ μm and $l_z \cong 600$ μm, respectively. We probed the vortex cores using resonant absorption imaging. To avoid blurring of the images due to bending of the cores near the edges of the condensate, we pumped a thin, 50- to 100-μm slice of atoms in the center of the cloud from the $F = 1$ to the $F = 2$ hyperfine state (*15*). This section was then imaged along the axis of rotation with a probe pulse resonant with the cycling $F = 2 \rightarrow 3$ transition. The duration of the pump and probe pulses was chosen to be sufficiently short (50 and 5 μs, respectively) to avoid blurring due to the recoil-induced motion and free fall of the condensate.

We observed highly ordered triangular lattices of variable vortex density containing up to 130 vortices (Fig. 1). A striking feature is the extreme regularity of these lattices, free of any major distortions, even near the boundary. Such "Abrikosov" lattices were first predicted for quantized magnetic flux lines in type-II superconductors (*16*). Tkachenko showed that their lowest energy structure should be triangular for an infinite system (*17*). A slice through images shows the high visibility of the vortex cores (Fig. 2), which was as high as 80%. For a trapped condensate with maximum vortex density, we infer that the distance between the vortices was $\cong$ 5 μm. The radial size of the condensate in the time-of-flight images was over 10% larger when it was filled with the maximum number of vortices, probably due to centrifugal forces.

When a quantum fluid is rotated at a frequency Ω, it attempts to distribute the vorticity as uniformly as possible. This is similar to a rigid body, for which the vorticity

Department of Physics, Center for Ultracold Atoms at Massachusetts Institute of Technology (MIT) and Harvard University, and Research Laboratory of Electronics, MIT, Cambridge, MA 02139, USA.

*To whom correspondence should be addressed. E-mail: jamil@mit.edu