Cover Story

David Keil

Assistant Professor, Computer Science Department, Framingham State University, Framingham, Massachusetts, USA

Can Quantum Computing Provide Exponential Speedup?

Some New Developments

A recent development in quantum computing is the assembling of a team of twenty researchers by Google to build quantum computing devices. University of California Santa Barbara physicist John Martinis will lead the team. Work will be based in part on previous research with machines built by D-Wave, a Canadian company. D-Wave claimed to have created "the world's first commercially available quantum computer" in 2011.

Quantum computing has been a theoretical and even speculative field. It is based on the notion of qubits (pronounced "Q-bits"), whose values may be 0, 1, or a probability distribution over that set. Qubits are fragile in that tiny perturbations in a system may destroy them. Researchers have been able to extend their lives from nanoseconds, decades ago, to minutes, today. Martinis reports a 10,000-fold rise in the lifetimes of qubits that can be regularly maintained, up to 50 to 100 microseconds.

The progress of quantum computing to recognition by significant forces in industry and academia is striking. Twenty years ago, an expert noted that capabilities "permit only the most rudimentary implementations of quantum computing"[7]. Has this changed?

The progress of quantum computing to recognition by significant forces in industry and academia is striking

Also in 2014, a test of the $10 million D-Wave computer, posted online at *Science* magazine in June, produced results that some researchers described as not supporting claims of exponential speedup[3, 14]. The machine used from 8 to 512 qubits. The summer 2014 testing was done by a team that included Martinis, as well as Matthias Troyer of the Swiss Federal Institute in Zurich. The co-founder of D-Wave has been quoted disparaging the test in a communication to *Wired* magazine.

The test team, which included researchers from the University of Southern California, University of California Santa Barbara, Google, and Microsoft, defined quantum speedup, S(*n*), for a problem with input of size *n*, as C(*n*) / Q(*n*), where C(*n*) is the time required by a classical device and Q(*n*) is the time required by a quantum one.

The D-Wave machine is not a universal programmable computer, like classical devices, but rather what is called a quantum annealer. It solves certain problems by having a two-dimensional array of qubits interact to reach a "ground state." Annealing, a metaphor for which is used in some artificial-intelligence algorithms, is a physical process of "cooling down" or lowering energy level or disorder in a system.

A Fast Quantum Algorithm for Factoring

Independent of claims for the performance of particular machines, an efficient quantum-computing algorithm for finding the factors of large integers has been published and studied since the mid-1990s[14]. Peter Shor's algorithm executes in time that is a polynomial function of the size of the input, but is not guaranteed always to provide the correct answer to the factoring problem. However, the answer it provides may be easily checked in polynomial time[7]. No known algorithm for classical computer architectures factors numbers in polynomial time. Hence Shor's algorithm provides exponential speedup for this task. But we find no indication that it has been used on a D-Wave or other quantum device.

The prospect that quantum computing might enable exponential speedup was explored by David Deutsch in the 1980s[6]. Research in the 1990s identified problems that quantum computers can theoretically solve quickly and exactly, that classical devices cannot. It induced discussion about a possible quantum-computing challenge to the principle, known as the *quantitative Church's Thesis*, that any physical computing device can be simulated by a Turing machine in a number of steps that is polynomial in the resources used by the computing device. The Turing machine is an imaginary mathematical device that, according to Alan Turing and Alonzo Church, can simulate any algorithmic computation. Turing and Church were logicians working in the 1930s.

. . . a possible quantum-computing challenge to the principle, known as the quantitative Church's Thesis, that any physical computing device can be simulated by a Turing machine in a number of steps that is polynomial in the resources used by the computing device

Shor wrote, however, that "quantum computers will likely not become widely useful unless they can solve NP-complete problems.... There are some weak indications that quantum computers are not powerful enough to solve NP-complete problems"[14]. Moreover, the only practical application of the quantum algorithm described by Shor appears to be in breaking public-key cryptography.

Satisfiability and Other Hard Problems

Many practical problems faced and solved by life forms require exponentially large amounts of time to solve perfectly. These include planning, because threats and opportunities explode exponentially as we look father into the future to weigh them. They also include resolving ambiguities in communication, since alternative possibilities also explode with the length of utterances. A large set of such problems exists, called NP-complete (NPC), none of which has a known polynomial-time solution but all of which would have such solutions if any one of them did. We don't solve them exactly; rather we obtain satisfactory approximate solutions.

Let us consider a case where the exponential speed-up, claimed by some for quantum computing, would open up entirely new practical computing horizons, if it were applicable to general-purpose computing. A classical NPC problem is the *satisfiability* problem (SAT) in propositional logic.

We may approach satisfiability by starting with the easier problem of evaluation of logic formulas. Suppose we are offered a formula, such as $(p \vee q)$

$\wedge$ ($\neg p \vee r$) $\wedge \neg r$, and a set of *assigned truth values* of the variables *p*, *q*, and *r*, such as (T, F, T); that is, *p* = true, *q* = false, and *r* = true. Then what is the value of the formula? It turns out to be false, as we can determine with a truth table or by substituting the values *true* and *false* for the variable names and performing the specified operations. This is the *evaluation* problem in propositional logic. It's easy to solve quickly, in time proportional to the number of variables or the size of the formula.

A different problem, about formulas alone, is to tell whether *some* set of variable assignments exists that makes the formula true. One way to solve this is to write a truth table and see if any row has a rightmost value *true*. For the formula ($p \vee q$) $\wedge$ ($\neg p \vee r$) $\wedge \neg r$, the truth table is as follows:

p	q	r	$p \vee q$	$\neg p \vee r$	$\neg r$	$(p \vee q) \wedge (\neg p \vee r) \wedge \neg r$
0	0	0	0	1	1	0
0	0	1	0	1	0	0
0	1	0	1	1	1	1
0	1	1	1	1	0	0
1	0	0	1	0	1	0
1	0	1	1	1	0	0
1	1	0	1	0	1	0
1	1	1	1	1	0	0

The formula is *satisfiable* because it evaluates to *true* in at least one case: the third row of the truth table (F, T, F). Unlike the evaluation problem in propositional logic, the satisfiability problem is believed to be very time consuming. This is because the size of a truth table with *n* variables is 2^n, and no faster method of solving SAT in the general case is known, other than checking every row of such a truth table. If *n* is 200, then the truth table will fill the universe.

A 2006 paper by S. Bravyi describes a quantum algorithm for quantum 2-SAT, a version of the satisfiability problem that replaces propositional-logic formulas with two variables by 2-qubit states, and replaces the assertion that a set of variable assignments satisfies a formula with the quantum *n*-qubit state[2]. Bravyi describes BQP, the set of problems that a quantum computer can solve in polynomial time with a bounded probability of error. Similarly, QMA is a set of problems whose solutions may be verified efficiently on a quantum computer, given a possible solution, analogous to the problem set *NP* in classical computing.

On the other hand, a 2010 paper by researchers at Princeton and the Max Planck Institute indicates pessimistic prospects for quantum solutions to the SAT problem[12]. For problems like satisfiability or other NP-complete problems, "the quantum benefits appear to be inherently more restricted" than for factoring.

Quantum Theory and Computing

Classical computing, with microprocessors, bits, and random-access memory, could not have become practical without quantum theory, which made possible the invention of the transistor[3]. The transistor replaced the slow, unreliable relay and vacuum tube as a switch that switches another switch. Likewise, lasers, essential for devices such as DVDs, are generated by the quantum effect of excitation of electrons.

For problems like satisfiability or other NP-complete problems, "the quantum benefits appear to be inherently more restricted" than for factoring

Quantum mechanics is a development in theoretical physics that occurred in the twentieth century. It supplemented at the *subatomic* level the dynamics that Newtonian and Einsteinian physics had explained at the level of large bodies and waves in motion. Whereas Newton explained that forces accelerate large bodies, Einstein's theory of relativity explained that this acceleration has a limit – the speed of light – and that matter and energy, space and time, particles and waves may be seen accurately as pairs that are one.

Quantum theory addressed the properties of wave/particles and discovered the discreteness of behavior at the particle level. Whereas we think of the size of a planetary orbit as a quantity along a continuum, for example, the possible orbits of an electron around the nucleus of an atom are limited to certain discrete quantities.

In a quantum computer, symbols can be either 0, 1, or a *superposition* of probabilities of 0 and 1 at the same time. The superposed state collapses when it is measured. *Entanglement* in quantum theory is the case where two particles may even at a distance have states that depend on each other. These are weird-sounding and counterintuitive notions for people who work in fields like classical computing.

Quantum theory has from the beginning challenged common-sense intuition. A century ago (1913), Niels Bohr published a description of atoms as electron particles orbiting clusters of other particles. Unlike planetary orbits in the solar system, electron orbits are discretely constrained, and changes in orbit are accompanied by emission or absorption of energy. We might notice that in this sense, atoms have a digital character that solar systems lack at the macro scale.

. . . atoms have a digital character that solar systems lack at the macro scale

Quantum State, Gates, and Registers

Researchers in quantum computing have described notions of quantum *state*, analogous to the state of a computation on a classical machine, and quantum *gates*, analogous to logic gates in classical computing. On an ordinary computer, *state* is the set of values of all bits in the machine, or equivalently, the values of all variables in a computation. The state of a quantum computer is described by a vector that is a linear superposition of all bits in the vector.

State transitions are described by a unary operator on the vector space[1]. A 3-qubit register, for example, may have the state $|010\rangle$, also written $|2\rangle$, also written (0, 0, 1, 0, 0, 0, 0, 0), where the 1 is the probability that the register has the value 2 and the 0s are the probabilities that is has the values 0, 1, or 3 to 7.

All the allowed operations on qubit registers are rotations. Since rotations may be done forward or backward, all quantum computations are reversible. Hence only reversible classical algorithms may be executed on quantum machines. This is not an insurmountable obstacle[14].

Quantum unary transformations and quantum XOR gates enable the building of circuits of arbitrary complexity in quantum computers[7]. Multiple computational pathways may, by superposition, evolve in parallel. For example, a 1000-qubit register may take all 2^{1000} pathways. State is indeterminate in a quantum computer at any intermediate stage, but may be definite, hence useful, at the completion of a computation.

Errors and *decoherence* (collapse of superposed states) are two concerns of implementers of quantum computing. Errors grow exponentially as the quantity of qubits in a system rises. Interaction of the extremely sensitive quantum systems with their environments may destroy qubits.

Copying qubits is not possible, because their quantum state is destroyed in the copying. Almost all digital computing operations involve copying. But theorists have worked on detection schemes that involve comparing rather than copying qubits.

Copying qubits is not possible, because their quantum state is destroyed in the copying

Determinism: A Foundation of Computer Science

The notions that observable entities, such as particles, are in *indeterminate*, *undefined*, or *probabilistic* states seems to collide with the necessary foundations of computer science. This discipline is concerned with the execution of algorithms and algorithm-based interactive processes in the real world. Algorithms compute mathematical functions on representations of parts of the real world. A function on natural numbers or strings of symbols is, *by definition*, a mapping from one discrete value *x* to another, *y*, such that for any *x* a unique *y* exists. Here *x* is called input and *y* is output.

In computer science, a notion of *nondeterminism* exists; namely, that for a given *x, y* may be a set. Thus we could make *x* a city and *y* the set of cities with direct airline connections to *x*. If we consider where we could be after taking some plane on one flight from city *x*, then the set *y* would contain all the possibilities. If we imagine boarding a random plane, the one city where we will land is constrained but could be considered nondeterministic. Nondeterministic automata (simple abstract mathematical machines) are of great theoretical value in computer science.

On the other hand, to *compute* such a set *y* of one-stop destination cities would require a representation of *y*, a representation of sets. Such digital representations exist, and they are by no means random, nor probability distributions.

Thus quantum computing challenges those of us who are trained in computer science to stretch our imaginations and our notions of what computing is. In the same way, quantum theory challenged physicists trained in the Newtonian assumptions about physics and in the much more recent theory of relativity.

Fundamental Obstacles to Some Speedup Prospects?

Possibly a final negative answer to the question posed by this article is a result reported in 2003[10]. According to a theorem by Jozsa and Linden, the quantum entanglement of multiple elements, with a number unbounded with respect to input size, is needed for exponential speedup. Shor's algorithm, for example, entails entanglement of an unbounded number of particles. A widely respected 2008 paper affirms this result in passing[5].

Thus, the cost of exponential speedup may be exponential quantum processing hardware. This is similar to the requirement, for some parallel computation, that an exponential number of processors be available in order to provide exponential speedup. The superposition of 2^{1000} states and pathways may be physically possible under quantum theory. The entanglement of 2^{1000} particles, more than can exist in the universe, is not. If the result by Jozsa and Linden holds, then despite the power of quantum computing, it will not break the performance-to-resources barrier established tentatively by complexity theorists forty years ago.

Thus the cost of exponential speedup may be exponentially large quantum processing hardware

Is Quantum Computing Better Adapted to Fuzzy Problems?

Whereas the formal descriptions of the hard problems encountered by humans and all other life forms are discrete, and these problems (called NP-complete) give evidence of being intractable and not worth solving precisely, these life forms nevertheless survive by finding satisfactory *approximate* and *probabilistic* solutions. (In fact, since these problems are in continuous physical space and time, they are in that form *unsolvable* by any digital computer, anyway[6].)

Some researchers have noted that these *fuzzy* problems are well suited to quantum computing[13]. A sequence of *fuzzy* numbers may correspond to our uncertainty about the inputs to a computation. A fuzzy number *x* is a generalization of a real number, in that it is a set of values v that may be *x*, each with an extent to which it *is* *x*. Thus it is similar to a fuzzy set, such as *Tall*, the set of all possibly tall people together with the degrees to which each person is deemed tall.

Two ideas of these researchers for quantum algorithms that speed up fuzzy computing are as follows:

1. Since fuzzy numbers may be represented as classes of real intervals, fast quantum interval algorithms may solve problems involving fuzzy numbers. (Whereas classically computing the range of an interval is NP-hard, quantum algorithms offer tractable solutions.)
2. Computing a function on a sequence of fuzzy numbers on a quantum computer may reduce the classical running time to its square root for the most time-consuming step, which is the minimum of all possible combinations of inputs.

. . . the work on quantum algorithms for fuzzy computing may point the way toward fruitful avenues of research leading to practical contributions

Readers of this article may wish to consider looking into these optimistic claims. Though a speedup of $\sqrt{n}$ is not exponential, or even linear, the work on quantum algorithms for fuzzy computing may point the way toward fruitful avenues of research leading to practical contributions.

References

[1] Adriano Barenco et al. Elementary gates for quantum computation. *Physical Review* A 52:5 (1995): 3457.

[2] Sergey Bravyi. Efficient algorithm for a quantum analogue of 2-SAT. No. quant-ph/0602108. Feb, 2006.

[3] Adrian Cho. Quantum or not, controversial computer yields no speedup. *Science* 344:6190 (June 20, 2014), pp. 1330-1331.

[4] Brian Cox and Jeff Forshaw. *The Quantum Universe*. Da Capo Press, 2011.

[5] Animesh Datta, Anil Shaji, and Carlton M Caves. Quantum Discord and the Power of One Qubit. *Phys. Rev. Lett*. 100, 050502 (2008).

[6] David Deutsch. Quantum theory, the Church-Turing principle and the universal quantum computer. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences*, 400:1818 (1985), pp. 97-117.

[7] DiVincenzo, David P. Quantum computation. *Science* 270:5234 (1995): 255-261.

[8] Richard P Feynman. Simulating physics with computers. *International Journal of Theoretical Physics* 21:6/7 (1982), pp. 467-488.

[9] Elizabeth Gibney. Quest for Quantum Computers Heats Up. *Scientific American*, Dec. 4, 2014.

[10] Jeremy Hsu. Google hires quantum computing expert John Martinis to build new hardware. *IEEE Spectrum*, Sept. 8, 2014.

[11] Richard Jozsa and Noah Linden. On the role of entanglement in quantum-computational speed-up. Proceedings of the Royal Society of London. *Series A: Mathematical, Physical and Engineering Sciences* 459.2036 (2003): 2011-2032.

[12] Christopher R Laumann et al. On product, generic and random generic quantum satisfiability. *Phys. Rev. A* 81:6 (2010).

[13] Mark Martinez, Luc Longpre, Vladik Kreinovich, Scott A Starks, and Hung T Nguyen. Fast quantum algorithms for handling probabilistic, interval, and fuzzy uncertainty. Department of Computer Science Technical Report UTEP-CS-03-16, University of Texas El Paso, 5/1/2003.

[14] Peter W Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. *SIAM Journal on computing* 26:5 (1997): 1484-1509.

[15] Troels F Ronnow, Matthais Troyer, et al. Defining and detecting quantum speedup. *Science* 345:6195 (July 25, 2014), pp. 420-424.

About the Author

Prof. David M Keil has taught at Framingham State University, USA, since 1997. He has acted as Director of Assessment for computer science since 2009. Additional responsibilities include as Search Committee chair in Spring 2001 and member of search committee in 2004 and 2013-2014. He was Acting chair in Spring 2000-Fall 2000 and 2003-2004 and Faculty coordinator for computer-science lab during 1997-1998.

He has presented at workshops on theory and practice of open computational systems, evolutionary computation, environments for multi-agent systems, foundations of interactive computing, and teaching and assessment in computer science. His research interests include interactive models of computation, evolutionary computation, artificial intelligence, database theory and Kolmogorov complexity.