ICM 1998, Berlin, Aug. 18–27

Abstracts of Plenary and Invited Lectures

Section:

0. Plenary Lectures

1991 MS Classification: 11Y05, 68Q05, 81P99
Shor, Peter W., AT&T Labs, Florham Park, USA
Quantum Computing

Church's thesis says that any computable function is computable on a Turing machine. This thesis arose in the 1930's, and was motivated by the realization that three quite distinct definitions of computable functions were all equivalent. It is well-known that this is not a theorem, as Church's thesis does not specify a rigorous mathematical definition of "computable"; specifying such a definition would lead to a provable theorem (and in many cases has), but it detracts from the generality of the thesis. What is somewhat less commonly realized is that this thesis can be viewed as a statement about the laws of physics, by interpreting computable as meaning computable in the physical world. For this interpretation, if the laws of physics are computable by a Turing machine, then Church's thesis is true.
The development of digital computers rendered the distinction between computable and uncomputable functions too coarse in practice, as it does not take into account the time required for computation. What was needed was some characterization of efficiently computable functions. Theoretical computer scientists reached a good compromise between theory and practice with the definition of polynomial-time computable functions as those functions whose value can be computed in a number of steps polynomial in the input size. This definition is abstract enough to have good theoretical properties for use in proofs, while for functions arising in practice it includes most of the efficiently computable ones and excludes most of those not efficiently computable. This definition naturally gave rise to a "folk thesis, " the polynomial Church's thesis, which says that any function physically computable in time t can be computed on a Turing machine in time p(t), where p is some polynomial.
Is this thesis valid? One good place to start looking for counterexamples would be physical systems which require large amounts of computer time to simulate. Two obvious such candidates are turbulence and quantum mechanics. In 1982, Feynman considered the case of quantum mechanics, and concluded that quantum mechanical systems seem to inherently require an exponential overhead to simulate on digital computers. In a "side remark, " he proposed using quantum computers, operating on quantum mechanical principles, to circumvent this problem. Deutsch (1985) gave a definition of a quantum Turing machine and suggested that quantum computers might solve classical problems more quickly than digital computers. It currently

appears that this is indeed the case. One of the pieces of evidence for this is that quantum computers can factor integers in polynomial time, something which it is not known how to do on classical computers despite many years of study.

A nice abstract model of a quantum computer is the quantum circuit. This machine computes using n 2-state quantum systems; each of these 2-state systems is called a qubit. The quantum state space of the joint system is a complex vector space of dimension 2^n, formed from the tensor product of n two-dimensional complex vector spaces, one associated with each qubit. The state of this system is described by unit vector in this 2^n-dimensional space. Because of the Heisenberg uncertainty principle, the output state cannot be obtained exactly. Rather, if the end result of the computation is $\sum_{i=0}^{2^n-1} \alpha_i v_i$, the observed output will be some integer $0 \leq i < 2^n$, and i will be observed with probability $|\alpha_i|^2$ (here the v_i are basis vectors of the quantum state space). Possible physical transitions are unitary transformations of this vector space. We cannot assume that we can perform any unitary transformation in one step, as this is not only physically unrealistic, but also would result in any function being computable in unit time. Rather, we must assume that in one step we can make a unitary transformation on one or two of the qubits, which via the tensor product naturally induces a transformation on the whole vector space. In fact, it suffices to be able to perform a small number of different one- and two-qubit unitary transformations to perform any quantum computation (Barenco *et al* 1995).

For factoring an L-bit number N, the best classical algorithm known is the number field sieve, which asymptotically takes time $O(\exp(cL^{1/3}\log^{2/3} L))$. On a quantum computer, the quantum factoring algorithm takes asymptotically $O(L^2 \log L \log\log L)$ steps. The key idea is to use a Fourier transform to find the period of the sequence $a_i = x^i (\mathrm{mod\ N})$, from which period a factorization of N can be obtained. The period of this sequence is exponential in L, so this approach is not practical on a digital computer. On a quantum computer, however, we can find the period in polynomial time by exploiting the 2^{6L}-dimensional state space of $6L$ qubits, and taking the Fourier transform over this space.

Quantum computers will be very hard to realize physically. We need quantum systems which are relatively stable, and which interact strongly with each other but weakly with everything else. Without error correction, it would probably be an impossible engineering task to build quantum computers large enough to factor 100-digit numbers—factoring such a number requires billions of steps on a quantum computer, so each step would need to be accurate to better than one part in a billion. Fortunately, it is possible to design fault-tolerant circuits for quantum computers, which allow computations of arbitrary length to be performed with gates accurate only to some constant c. Current estimates put this constant larger than 10^{-4} (Preskill 1997).

References:

A. Barenco, C. H. Bennett, R. Cleve, D. P. DiVincenzo, N. Margolus, P.

Shor, T. Sleator, J. A. Smolin, and H. Weinfurter, Elementary gates for quantum computation, *Phys. Rev. A* **52** (1995), 3457–3467.
D. Deutsch, Quantum theory, the Church–Turing principle and the universal quantum computer, *Proc. Roy. Soc. London Ser. A* **400** (1985), 96–117.
R. Feynman, Simulating physics with computers, *Internat. J. Theoret. Phys.* **21** (1982), 467–488.
J. Preskill, Fault-tolerant quantum computation, LANL e-print quant-ph/9712048, available at URL `http://xxx.lanl.gov/` (1997).
P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, *SIAM J. Computing* **26** (1997), 1484–1509.