Can a Universal Quantum Computer Be Fully Quantum?

John M. Myers

Gordon McKay Laboratory, Harvard University, Cambridge, Massachusetts 02138-2901

(Received 3 December 1996)

A conflict is pointed out in the definition of a universal quantum computer between the need for a halt qubit and the need for operating on superpositions of states of a computational basis. [S0031-9007(97)02532-5]

PACS numbers: 89.70.+c, 03.65.–w

Computation as defined by Turing has been discussed from the standpoint of quantum mechanics for some years, and is under experimental as well as theoretical investigation. Early on, Deutsch discussed two rather different theoretical approaches: (1) a universal quantum computer [1] and (2) a quantum computational network [2]. Both forms compute by transforming an input state into an output state (in the Schrödinger picture), and both are claimed to operate on superpositions of states of a computational basis, thus exhibiting quantum parallelism. The capacity to operate on an input state that is a superposition of computational basis states was emphasized by Deutsch as necessary for a computer to be "fully quantum."

Although the computational network connects more directly to experiments and to such applications as factoring, the universal quantum computer mirrors more closely the Turing machine, sharing with it the theoretical capacity to compute any recursive function. For this reason and perhaps others, it is an interesting theoretical object.

For either a computational network or a universal quantum computer, reading the result of a computation requires measuring the state of the computer when the state is the output state. Because measuring an unknown state generally changes the state, this measurement must be made after the computation has been completed, and not before; otherwise the computation is spoiled.

Because of the halting problem [3], the issue of when a computation of a recursive function is complete cannot be sidestepped. In discussing the universal quantum computer, Deutsch partitioned the computational basis states, making each basis state a tensor product of two factors, the first an (unbounded) string of qubits, the second a single halt qubit n_0. Every input state is some superposition of basis states having n_0 initialized to 0. Starting from any basis state, any executing program sets n_0 to 1 when the calculation is complete but does not interact with n_0 otherwise. Regardless of whether the halt qubit is 0 or 1, a measurement to decide between 0 and 1 as possible states of the halt qubit finds the halt qubit in an eigenstate. This leaves the state of the computer unaffected, so the halt qubit can be measured repeatedly during a computation that starts from any basis state without spoiling the computation; once a value of 1 is obtained for the outcome, the state of the computer can be measured to register a result.

But what happens if the computation starts not from a computational basis state but from a superposition of basis states? In this case, a measurement of the halt qubit can spoil the computation, as follows. Suppose a calculation that starts from a basis state $|A\rangle|0\rangle$ as input sets the halt bit to 1 after N_A steps (where the second factor is the halt qubit set to 0). Similarly, suppose a calculation starting from basis state $|B\rangle|0\rangle$ sets the halt bit to 1 after N_B steps. Cases exist for which $N_B \gg N_A$. Consider starting from an input state which is the superposition of basis states $c_A|A\rangle|0\rangle + c_B|B\rangle|0\rangle$. The halt qubit must then be 0 for steps $N < N_A$, and 1 for steps $N > N_B$. For $N_A < N < N_B$, the state is a superposition of two basis states, one of which has the halt qubit set to 0, and the other of which has a qubit set to 1. In other words, in this range of steps the state entangles the non-halt qubits with the halt qubit. A measurement made of the halt qubit when it is entangled with the other qubits changes the state and spoils the computation.

Thus it follows: If paths of a quantum computation starting from two different input basis states halt at different counts, then a superposition of the two input states entangles the halt qubit for the steps between the one count and the other, so the measurement of the halt qubit is incompatible with the unrestricted superposition of input basis states.

For many special cases, including the factoring of large numbers [4], this difficulty need not occur, because in these cases the computation involves a number of steps which (for any computational basis state as input) is independent of the input. To be a universal quantum computer, however, the computer must calculate arbitrary recursive functions, and for this class of function the number of steps cannot always be held independent of the input. Hence, while some computational networks can be fully quantum, there is a conflict between being universal—capable of computing arbitrary recursive functions—and being fully quantum (capable of computing values from inputs which are superpositions of computational basis states).

This limitation has been neither noticed nor circumvented in later work with which I am familiar. For example, Bernstein and Vazirani focus on special cases in

which completion need not be interrogated [5]. A recent review by Ekert and Jozsa mentions the need to detect completion, but invokes Deutsch's halting qubit, the limitation of which is the subject of this note [6]. Spiller's more recent review makes use of Deutsch's definition of a universal quantum Turing machine, without discussing completion [7].

The question can be raised whether the limitation can be circumvented; perhaps some clever scheme, analogous to schemes for quantum error correction, might help. But while quantum error-correcting schemes involve measurements made in the course of a computation, they cannot and do not measure the state or the part of the state that carries the outcome. They exhibit great care and cunning in avoiding this, by measuring something much more restricted. Getting a result requires measuring the output part of the state. Thus, no direct carryover of techniques of error correction can circumvent the limitation: The design for a quantum computer by Deutsch cannot be both universal and fully quantum. Whether some other model of quantum computation can be invented is open to the future. Indeed a purpose of this note is to call attention to the present lack of such a model, partly in hopes of spurring its invention.

The conflict between universality and being fully quantum is a conflict internal to Deutsch's model of "universal quantum computer." Within the scope of that model, there is no hope for relief, unlike some of the more practical troubles that have been cited in the extensive literature on decoherence in quantum computers.

How much of a restriction is the limitation that has been pointed out? As noted, it is no threat to the factoring of large numbers, because for this task the running time can be known prior to the computation. For many other computations, one can make statistical predictions for their execution time. Then, if the prediction is borne out, a correct answer can usually be produced without the use of a halt qubit. However, a universal computing machine (quantum or not) is supposed to compute any function of the class of recursive functions, most of which do not have this property. So, whether the limitation is restrictive or not depends on whether one wants to consider the class of recursive functions or only some small subset. Sticking to a small subset is reasonable in many practical situations, but destroys the possibility of discourse that carries the subject of computability into a quantum context.

I thank Amr Fahmy for a discussion of Shor's algorithm for factoring, and Howard Brandt for several discussions on quantum computers. This work was supported in part by the Army Research Laboratory and Berkeley Research Associates under Subcontract No. BRA-96-W195PO.

[1] D. Deutsch, Proc. R. Soc. London A **400**, 97 (1985).
[2] D. Deutsch, Proc. R. Soc. London A **425**, 73 (1989).
[3] G. S. Boolos and R. C. Jeffrey, *Computability and Logic* (Cambridge University Press, Cambridge, England, 1989), 3rd ed.
[4] P. Shor, in *Proceedings of the 35th Symposium on Foundations of Computer Science*, edited by S. Goldwasser, (IEEE Computer Society Press, Los Alamitos, 1994), pp. 124–134.
[5] E. Bernstein and U. Vazirani, in *Proceedings of the 25th ACM Symposium on Theory of Computing* (ACM, New York, 1993), pp. 11–20.
[6] A. Ekert and R. Jozsa, Rev. Mod. Phys. **68**, 733 (1996).
[7] T. P. Spiller, Proc. IEEE **84**, 1719 (1996).