

Short Introduction to Quantum Computation

Jeffrey Yepez*

Phillips Laboratory, Hanscom Field, Massachusetts
yepez@plh.af.mil

26 July 1996

Abstract

Presented is quantum lattice gas method useful for nanoscale computing and quantum computing.

KEY WORDS: quantum computing, nano-scale computing, Moore's law

1 Introduction

It is likely that within this human generation nanometer scale computing will prevail as a standard computing technology. Figure 1 is a log-linear plot of the areal size of a bit over the last fifty years for a variety of commercial-grade device technologies. Consistent with Moore's law but also spanning technologies earlier than integrated circuits, it is clear there has been an exponential reduction in bit size where the characteristic linear dimension has been halving approximately every two years. It appears that a computational bit's size is heading towards the atomic scale, and if the trend indicated in Figure 1 continues, atomic computing densities will be achieved perhaps within two decades from now.

2 Nano-scale computing

There are several important issues that arise when one considers fabricating nanoscale computing devices, and these issues are different depending on the type of computing one expects to do at this scale.

The first type of computing, introduced by Ed Fredkin, Tom Toffoli, and Norm Margolus [1, 2], would be classical computing where Boolean bits that have a definite value of either 0 or 1 are still employed

*This work supported by the Air Force Office of Scientific Research under the Mathematical and Computational Sciences initiative No. 2304CP.

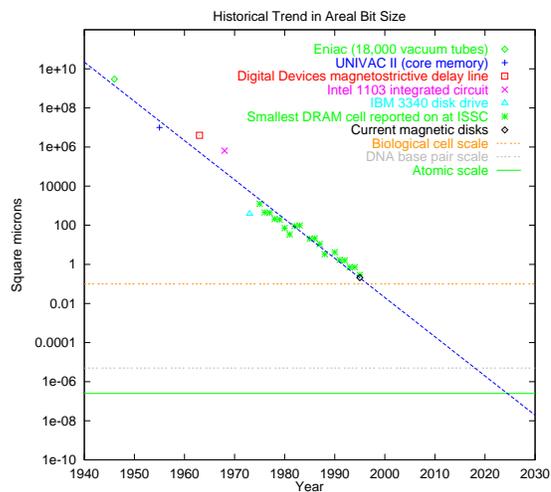


Figure 1: Exponential reduction in areal size of a bit for the last fifty years.

and where all logical gate operations are represented by unitary permutation operators causing neither any quantum mechanical superposition nor any quantum entanglement. This kind of computing may best be termed *nano-scale computing*. I'll describe the important issues of nano-scale computing below.

A second type of computing, introduced by Richard Feynman [3, 4], has been termed *quantum computing* where two-level quantum objects, such as spin- $\frac{1}{2}$ particles, are used to represent quantum bits or "qubits" and where quantum superposition and entanglement are integral to the logical gate operations and are in fact required for computational efficiency. The characteristic nature of a qubit is that it can be in a superposition of the Boolean states $|0\rangle$ and $|1\rangle$, prototypically the ground state and excited state of a two-level quantum system. That is, if one measures the value of the

qubit, binary values are observed corresponding to either the square of the probability amplitude of it being in the ground state, $|0\rangle$, or the square of the probability amplitude of it being in the excited state, $|1\rangle$. Of course, the probability of these classical outcomes add to unity: $\langle 0|0\rangle + \langle 1|1\rangle = 1$.

2.1 Nano-scale Computing

In nano-scale computing, such considered by Mike Bifare [5], one tries to implement reversible classical algorithms whose logical gate operations are represented by orthogonal permutation matrices, which are a special class of unitary matrices. There are several reasons why one is driven to develop reversible algorithms in this context.

Firstly, already with present-day micro-scale central processing units, uncontrolled heat production and dissipation is a menacing problem, albeit a manageable one to-date. In nano-scale devices, one would expect that uncontrolled heat dissipation would be a fatal problem because the nano-scale device components and wires might be so delicate and fragile that heat modes could in fact couple strongly with neighboring device components or wires causing them to melt or substantially deform. The obvious solution is to avoid producing any increment of heat, say dQ , at all costs. Charlie Bennett has argued that reversible logic operations can avoid this, in principle [6]. Since information is exactly preserved in a reversible unitary algorithm, the Gibbs entropy, S , is constant throughout the course of the calculation ($dS = 0$). Consequently, since

$$dQ = TdS = 0, \tag{1}$$

no heat is produced, where T is the operating temperature of the device. Here we are assuming of course that such a simplistic thermodynamics argument as (1) can be applied, at least in a heuristic way, to a nano-scale device which would be better characterized by a proper lower-level quantum mechanical description.

Secondly, since all nonrelativistic dynamics at the nano-scale are governed by the Schroedinger wave equation with a Hamiltonian, \hat{H} , represented by a hermitian matrix, the evolution operator

$$\hat{U} = e^{-i\hat{H}t/\hbar} \tag{2}$$

is unitary, so ideally the time-history of the device is invertible. The consequence to computing of building up algorithms out of sequences of unitary operators of the form (2) is that the underlying occupancy of quantum memory states in the nano-scale device itself

would undergo reversible transitions obeying detailed-balance conditions.

Conversely, for any reversible algorithm, one can cast that algorithm in terms of an effective Hamiltonian (approximately a sum of hermitian matrices, good to a pre-specified level of accuracy) that controls the temporal dynamics of the state data, for which there corresponds a unitary evolution operator. For example, lattice-gas algorithms of Frisch, Hasslacher, Pomeau [7], and Wolfram [8] for modeling viscous fluid dynamics are reversible algorithms where the effective Hamiltonian of the lattice-gas system is block diagonal over the particle configurations with a particular set of conserved quantities (equivalence classes) and where the evolution operator is simply a unitary permutation matrix. For any reversible algorithm chosen, the task is to map the computational ‘‘Hamiltonian’’ of the algorithm on to the Hamiltonian of the nano-scale device physics in question. Since microscopic physical dynamics is described by hermitian Hamiltonian operators, it is convenient to describe microscopic algorithms that way too. So I believe the first crucial issue in this regard is finding what are some useful reversible algorithms for physical modeling and casting them in terms of interaction Hamiltonians that are technologically accessible for nano-scale engineering. Once the particular quantum mechanical Hamiltonian representations of useful reversible algorithms are found, then the question of whether it is possible to actually construct a nano-scale device to implement those reversible algorithms becomes worth answering by a directed Air Force basic research effort.

Thirdly, another pressing issue in nano-scale computing has to do with classical parallelism. The notion of having bits stored at atomic scales allows us to contemplate densities so high that any computation would necessarily have to be local, involving only nearby neighbors, and consequently would be quite fine-grained. So the issue of classical parallelism here involves coming up with a reasonable strategy of reading and writing to such a large collection of bits in a reliably controlled and clocked fashion.

Reversible algorithms for useful computational physics application are already known today for modeling the dynamics of viscous fluids, as already mentioned above, liquid-gas phase transitions of Appert and Zaleski [9], liquid-solid phase transitions of Yepez [10], binary immiscible fluids of Rothman and Zaleski [11, 12], and microemulsions of Boghosian, Coveney, and Emerton [13]. In turn, our group has made progress in our exploration into how to translate these kinds of rule-based reversible algorithms into the quan-

tum mechanical language of effective Hamiltonians. Furthermore, Seth Lloyd has argued it is certainly plausible that engineering-grade nano-scale technologies for computing will exist in the foreseeable future [14].

2.2 Quantum Computing

In quantum computing, one tries to do quantum physics engineering to construct efficient alternative device solutions suited to running algorithms that would otherwise require exponentially large computing devices when engineered using only the principles of classical physics or would require an exponentially large amount of processing time. For example, Peter Shor's research seems to indicate that on a quantum computer prime number factoring can in principle be done in a time that grows polynomially in the size of the input composite number to be factored [15, 16]. In contrast, today on a classical computer, even using the best known algorithms, the time required to find all the factors of a composite number grows exponentially in the size of the input composite number. Shor's theoretical argument that prime factoring can be done efficiently is sending shock waves through the cryptography community because the security of their schemes is centrally based on the assumption that finding the prime factors of a large composition number, say 128 or 256 bits in length, is essentially an intractable numerical problem on conventional computers. There are two difficult issues I would like to address, the first is experimental and the second is theoretical in nature.

Firstly, quantum computing relies on having quantum interference and entanglement over a system of qubits occurring in a controlled fashion. These qubits will likely be spin- $\frac{1}{2}$ objects. For example, nuclear spins within a suitable molecule with carbon-13 isotopes and hydrogen atoms, or some other two energy-level state system such as those in a solid-state quantum well or perhaps the spin of an electron localized in a laser controlled trap containing an artificial quantum chain or in a long polymer. Light might be used to initialize the spin states of the qubits (writing), and then after the qubits have interfered in some computing cycle, they might also be clocked by a sequence of light pulses; light might then likewise be used to measure the resulting spin states of the qubits (reading). This kind of controlled light-and-matter interaction is well known in nuclear-magnetic-resonance experiments, for example like those carried out by Cory, Fahmy, and Havel [17], where π -pulses are used to tip nuclear spins timed in units of Planck's constant divided by the coupling strength of the scalar spin-spin

interaction Hamiltonian. The most difficult issue for quantum computing is sufficiently isolating the qubits from the surrounding environment to avoid coupling with spurious quantum modes or energy levels, yet while at the same time maintaining a high-degree of controlled quantum mechanical evolution. Since interference and entanglement effects are essential for the computation, any spontaneous coupling with the environment destroys such effects causing decoherence of the many-body quantum memory state of the quantum computer and thereby substantially ruining the computation.

Secondly, to date only one important algorithm is known whose efficiency is based on quantum interference and entanglement: Shor's prime number factoring algorithm. The reversible algorithms that I mentioned above for computational physics applications have a severe limitation: they have a very high amount of dissipation (strong shear and bulk viscosity modes, and strong sound damping modes). The high value of the transport coefficients limits the maximum attainable value of the Reynolds number (ratio of convection to dissipation) that can be modeled, forcing us to model fluid flow patterns well below the turbulent regime. Therefore only relatively simple flow conditions can be modeled in comparison to the turbulent flow conditions generally present during high-performance aircraft flights or within any running jet engine. It is possible that quantum generalization of the reversible lattice-gas algorithms will be found that allow us to reduce the amount of dissipation in our numerical fluid models, hopefully to the point where we can efficiently model complex fluid flows, even inviscid fluid flows, well within turbulent flow regimes. At this point in time however, this has not been proven to be possible, although it appears plausible. Therefore, it remains an open question as to whether the class of efficient quantum algorithms is broad enough to merit the likely tremendous nano-scale engineering costs of maintaining quantum coherence and long-range entanglement, or at least the basic research costs of attempting to figure out a way to do so.

3 Conclusion

I hope we are very realistic about this engineering feasibility issue. If we do build a prototype quantum computer in the near future, it will necessarily at first be quite a basic thing. It may allow for the interference of a few qubits (likely no more than two) to carry out a simple single quantum logical gate operation. It would not be running Shor's algorithm to factor any

number that could not be factored easily on a present-day hand-held or wristwatch-mounted classical computer. Yet the computational value of the quantum mechanical superposition of states, particularly entangled states, referred to as *quantum parallelism*, may be practically demonstrated at least at the experimental level as a proof-of-concept of possible engineering-grade quantum computing architectures of the future.

Quantum computing is a fascinating area where physics and computation merge into a new field of quantum physics engineering. For some time, particularly since the start of our new basic research initiative “Novel Parallel Computing Strategies” in 1992, I have believed it would be in the Air Force’s interest to help nurture the development of advanced computing technologies of this sort. The fulfillment of the promise of efficient quantum computing is a long way off in the distant future. But if indeed fulfilled one day, then the consequent harvest would be so abundant in reward that it would no doubt certainly make the very large number of required years of sowing new quantum technologies economically practical. It is worth while to pursue how quantum mechanics might be refashioned into the ultimate machine language.

4 Acknowledgement

I would like to thank Norman Margolus for many useful conversations about atomic-scale computing. He introduced me to the subjects of reversible logic, nanoscale computing, and quantum computing. Most of, if not all, the proverbs of physical computation listed below, which circumscribe the viewpoint presented in this article, come directly through those conversations with Norm.

References

- [1] Edward Fredkin and Tommaso Toffoli. Conservative logic. *International Journal of Theoretical Physics*, 21(3/4):219–253, 1982.
- [2] Norman Margolus, Tommaso Toffoli, and Gérard Vichniac. Cellular-automata supercomputers for fluid-dynamics modeling. *Physical Review Letters*, 56(16):1694–1696, 1986.
- [3] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21(6/7):467–488, 1982.
- [4] Richard P. Feynman. Quantum mechanical computers. *Optics News*, 11(2):11–20, 1985.
- [5] Michael Biafore. Cellular automata for nanometer-scale computation. *Physica D*, 70(4):415–433, 1994.
- [6] Charles H. Bennett. Thermodynamics of computation—a review. *International Journal of Theoretical Physics*, 21:219–253, 1982.
- [7] Uriel Frisch, Brosl Hasslacher, and Yves Pomeau. Lattice-gas automata for the navier-stokes equation. *Physical Review Letters*, 56(14):1505–1508, 1986.
- [8] Stephen Wolfram. Cellular automaton fluids 1: Basic theory. *Journal of Statistical Physics*, 45(3/4):471–526, 1986.
- [9] Cécile Appert and Stéphane Zaleski. Lattice gas with a liquid-gas transition. *Physical Review Letters*, 64:1–4, 1990.
- [10] Jeffrey Yepez. Lattice-gas crystallization. *Journal of Statistical Physics*, 81(1/2):255–294, 1994.
- [11] Daniel H. Rothman. From ordered bubbles to random stripes—pattern-formation in a hydrodynamic lattice-gas. *Journal of Statistical Physics*, 71(3-4):641–652, 1993.
- [12] Daniel H. Rothman and Stéphane Zaleski. Lattice-gas models of phase separation: interfaces, phase transitions, and multiphase flow. *Reviews of Modern Physics*, 1994.
- [13] Bruce M. Boghosian, Peter V. Coveney, and Andrew N. Emerton. A lattice-gas model of microemulsions. *Proceedings of the Royal Society, A*(452):1221–1250, 1996.
- [14] Seth Lloyd. Quantum-mechanical computers. *Physics Today*, Oct:140–145, 1995.
- [15] Peter W. Shor. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. In *Proceedings of the 35th Annual Symposium on Foundations of Computer Science*, pages 124–134. Santa Fe, NM, IEEE Computer Society Press, 1994.
- [16] Peter W. Shor. Scheme for reducing decoherence in quantum computer memory. *Physical Review A*, 52:R2493–R2496, 1995.
- [17] David G. Cory, Amr F. Fahmy, and Timothy F. Havel. Ensemble quantum computing by nuclear

magnetic resonance spectroscopy. Technical report tr-10-96, Harvard University Center for Research in Computing Technology, Aiken Computation Laboratory, 33 Oxford Street, Cambridge, MA 02138, December 1996.

- A finite physical system can only be used to store a finite amount of information: real numbers are not real.

A Proverbs of Physical Computation

- Quantum mechanics is the ultimate machine language.
- A heat engine is a special case of a computing engine.
- All computing engines are microscopically reversible: the good ones are also macroscopically reversible.
- Since microscopic physics is reversible, microscopic algorithms can be too.
- Computer scientist's version of 2nd Law of Thermodynamics: A reversible computing "engine" is the most efficient device for transforming information-bearing energy from one form to another.
- The job of the computer architects is to provide the programmer with a computational abstraction of reality; they shouldn't try to hide real physical constraints, since generic strategies are inevitably inefficient.
- In the future all processing should be local; today computer architects make all memory appear equally far away from the processor.
- In the future gates should be reversible and dissipate no heat; today computer architects make all gates dissipate at a maximum rate all the time.
- Our world is ultimately quantum mechanical; today computer architects put together macroscopic numbers of quantum elements to build their classical gates.
- Computer architects take discrete quantum objects, fighting against their discrete quantum nature, to buildup continuous classical objects, that in the end they use as discrete digital objects.
- Within a closed system all algorithms are reversible by the time they are physically implemented.