Quantum Computing Classical Physics

David A. Meyer

Department of Mathematics
University of California, San Diego
La Jolla, CA 92039-0112

Air Force Office of Scientific Research
4015 Wilson Blvd
Mail Room 713
Arlington, VA 22203

Distribution Statement A. Approved for public release; distribution is unlimited.

In the past decade quantum algorithms have been found which out perform the best classical solutions known for certain classical problems as well as the best classical methods known for simulation of certain quantum systems. This suggests that they may also speed up the simulation of sonic classical systems. I describe one class of discrete quantum algorithms which do so-quantum lattice gas automata-and show how to implement them efficiently on standard quantum computers.

19a. NAME OF RESPONSIBLE PERSON
David A. Meyer

19b. TELEPHONE NUMBER (Include area code)
QUANTUM COMPUTING CLASSICAL PHYSICS

David A. Meyer

Project in Geometry and Physics
Department of Mathematics
University of California/San Diego
La Jolla, CA 92093-0112 USA
dmeyer@chonji.ucsd.edu

ABSTRACT

In the past decade quantum algorithms have been found which outperform the best classical solutions known for certain classical problems as well as the best classical methods known for simulation of certain quantum systems. This suggests that they may also speed up the simulation of some classical systems. I describe one class of discrete quantum algorithms which do so—quantum lattice gas automata—and show how to implement them efficiently on standard quantum computers.

2001 Physics and Astronomy Classification Scheme: 03.67.Lx. 05.10.-a.
2000 American Mathematical Society Subject Classification: 81P68, 65Z05.
Key Words: quantum lattice gas automata, quantum simulation, quantum Fourier transform.

1. Introduction

Quantum computing originated with Feynman's observation that quantum systems are hard to simulate on classical computers, but that they might be easier to simulate if one had a computer which could be operated quantum mechanically [1]. Developments during the subsequent two decades have not only supported this observation [2–8], but have also demonstrated that quantum computers would—if they existed—solve certain classical problems like factoring more efficiently than is possible on classical computers running the best classical algorithms known [9]. This raises a natural question [10–12]: Might quantum computers help with the simulation of classical systems? Or more specifically, given the focus of this workshop, could quantum computers efficiently simulate fluid dynamics? Without answering the specific question, in this paper I try to explain why the answer to the general question may be 'yes', and along the way explain some of the quantum algorithmic tricks which seem likely to be useful in future investigations of these questions.

I begin by (very) rapidly introducing the ideas of quantum computation in §2. For a complete presentation, see [13]. In §3 I describe one approach to simulating quantum systems—quantum lattice gas automata (QLGA) [14]—and then in §4 explain a connection with simulation of classical systems. The crucial issue is the relative complexity of quantum and classical algorithms; so §5 contains a detailed analysis for one specific problem, including some new results on implementing QLGA algorithms on 'standard' quantum computers. I conclude with a brief discussion in §6.

2. Quantum computers

The possible states of a classical computer are (very long) bit strings \( b_1 \ldots b_n \in \{0,1\}^n \). A particular computation proceeds via a sequence of maps to new bit strings: \( b'_1 \ldots b'_n \ldots \). A fundamental result, which contributed directly to the conceptual development of quantum computation, is that any classical computation can be made to be reversible [15], i.e., these maps can be chosen to be permutations on the space of states. Quantum computation can then be understood as a generalization of classical computation: The possible states of a quantum computer are superpositions of bit strings \( \sum a_{b_1 \ldots b_n} |b_1 \ldots b_n\rangle \in (\mathbb{C}^2)^\otimes n \) (each \( \mathbb{C}^2 \) tensor factor is called a quantum bit or qubit), where \( \sum |a_{b_1 \ldots b_n}|^2 = 1 \) so that the norm-squared of each amplitude \( a_{b_1 \ldots b_n} \in \mathbb{C} \) is the probability that the state \( |b_1 \ldots b_n\rangle \) is observed if the quantum system is measured in this 'computational basis'. A particular quantum computation proceeds via a sequence of unitary maps to new states \( \sum a'_{b_1 \ldots b_n} |b_1 \ldots b_n\rangle \), \ldots This much is a generalization of classical reversible computation since permutations are unitary maps, and each classical state is an allowed quantum state. The difference is that the final state is not directly available; it can only be sampled according to the probabilities given by the norm-squared of the amplitudes.

To evaluate the computational complexity of an algorithm, either classical or quantum, we must specify a set of elementary operations, the number of which used during the computation quantifies the complexity. If we allow arbitrary permutations classically, or arbitrary unitary transformations quantum mechanically, any state can be reached from
any other state in a single step—these are clearly not reasonable models of computation. Instead, consider the ‘gate operations’ shown in Fig. 1: NOT, C-NOT and C-C-NOT (‘C’ abbreviates ‘CONTROLLED’). Each of these is a permutation on the set of bit strings, and Toffoli [16] has shown that this set of three gates is universal for classical computation, in the sense that any (reversible) boolean operation can be decomposed as a sequence of these gates—which by Bennett’s result [15] suffices for universality. Each of these gates can be extended by linearity to a unitary map on \((\mathbb{C}^2)^\otimes n\), acting non-trivially only on a subset of 1, 2, or 3 qubits, and thus can also be used as a quantum gate. (Traditionally the quantum NOT gate is denoted by \(\sigma_x\) or \(X\).) Two other quantum gates which are particularly useful act on single qubits:

\[
H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \text{and} \quad R_\omega = \begin{pmatrix} 1 & 0 \\ 0 & \omega \end{pmatrix},
\]

the ‘Hadamard transform’ and phase \(\omega\) rotation, respectively. These matrices have been expressed in the computational basis; thus

\[
H|0\rangle \mapsto \frac{1}{\sqrt{2}} (|0\rangle + |1\rangle)
\]

\[
H|1\rangle \mapsto \frac{1}{\sqrt{2}} (|0\rangle - |1\rangle).
\]

C-NOT and \(R_\omega\) for \(\omega = e^{i\theta}\) with \(\theta = \cos^{-1} \frac{3}{8}\) form a universal set of gates for quantum computation [17].

As we noted earlier, any array of (reversible) classical gates can be simulated by some array of quantum gates. The remarkable fact is that in some cases fewer gates are required quantum mechanically. The following example is the smallest version of the Deutsch-Jozsa [18] and Simon [19] problems:

**EXAMPLE.** Given a function \(f : \{0,1\} \rightarrow \{0,1\}\), we would like to evaluate \(f(0)\) XOR \(f(1)\). The function is accessed by calls which have the effect of taking classical states \((x,b)\) to \((x, b \oplus f(x))\) where \(\oplus\) denotes addition mod 2. This is a reversible operation and thus also defines a unitary transformation on a pair of qubits: \(f\)-C-NOT. Classically, this gate must be applied at least twice (once with \(x = 0\) and once with \(x = 1\)) in any algorithm which outputs \(f(0)\) XOR \(f(1)\) correctly with probability greater than \(\frac{1}{2}\) (assuming a uniform
distribution on the possible functions). Quantum mechanically we can exploit interference to do better, applying the operation only once. Suppose the system is initialized in the state $|0\rangle \otimes |0\rangle$. Then apply the following sequence of unitary operations:

$$|0\rangle \otimes |0\rangle \xrightarrow{H \otimes H X} \frac{1}{2} \sum_{x=0}^{1} |x\rangle \otimes (|0\rangle - |1\rangle)$$

$$\xrightarrow{f_{\text{C-NOT}}} \frac{1}{2} \sum_{x=0}^{1} (-1)^{f(x)} |x\rangle \otimes (|0\rangle - |1\rangle)$$

$$\xrightarrow{H \otimes I_2} \sum_{x,y=0}^{1} \frac{1}{2\sqrt{2}} (-1)^{f(x)} (-1)^{y} |y\rangle \otimes (|0\rangle - |1\rangle)$$

$$= \frac{1}{\sqrt{2}} |f(0) \text{ XOR } f(1)\rangle \otimes (|0\rangle - |1\rangle).$$

The first step Fourier transforms the ‘query’ qubit into an equal superposition of $|0\rangle$ and $|1\rangle$, and initializes the ‘response’ qubit into a state which will create the phase $(-1)^{f(x)}$ in the second step. The third step Fourier transforms the query qubit again, creating the interference which ensures that subsequently measuring the query qubit outputs $f(0)$ XOR $f(1)$ correctly with probability 1. The gate array implementing this quantum algorithm is shown in Fig. 2; it includes only one $f$-C-NOT gate.

![Gate Array Diagram](image)

**Figure 2.** The gate array implementing the quantum algorithm which solves the XOR problem with a single call to the $f$-C-NOT gate. The states are not normalized.

This example displays perhaps the simplest quantum improvement over classical computation. Other superior quantum solutions to classical problems include Deutsch and Jozsa’s balanced function algorithm [18], Bernstein and Vazirani’s parity search algorithm [20], Simon’s period finding algorithm [19], Shor’s factoring algorithm [9], Grover’s unstructured search algorithm [21], van Dam’s algorithms for weighing matrices and quadratic residues [22], and Hunziker and Meyer’s highly structured quantum search algorithms [23].

### 3. Quantum lattice gas automata

Although the example in §2 demonstrates the superior computational power of quantum systems for certain problems, it seems to have little to do with confirming Feynman’s original idea that quantum systems could efficiently simulate other quantum systems [1]. For the community attending this workshop, a natural place to look for such confirmation is lattice models: The possible configurations for each particle on a one dimensional lattice $L$ are labelled by pairs $(x, \alpha) \in L \times \{\pm 1\}$, where $x$ is the position and $\alpha$ the velocity.
A classical lattice gas evolution rule consists of an advection stage \((x, \alpha) \mapsto (x + \alpha, \alpha)\), followed by a scattering stage. Each particle in a quantum lattice gas automaton (QLGA) [14] exists in states which are superpositions of the classical states: \(|\psi\rangle = \sum \psi_{x,\alpha} |x, \alpha\rangle\), where \(1 = \langle \psi | \psi \rangle = \sum \bar{\psi}_{x,\alpha} \psi_{x,\alpha}\). The evolution rule must be unitary; the most general with the same form as the classical rule is:

\[
\sum \psi_{x,\alpha} |x, \alpha\rangle \xrightarrow{\text{advect}} \sum \psi_{x,\alpha} |x + \alpha, \alpha\rangle \xrightarrow{\text{scatter}} \sum \psi_{x,\alpha} S_{\alpha\alpha'} |x + \alpha, \alpha'\rangle,
\]

where the scattering matrix is

\[
S = \begin{pmatrix} \cos s & i \sin s \\ i \sin s & \cos s \end{pmatrix}.
\]

Let \(U\) denote the complete single timestep evolution operator for a single particle, the composition of advection and scattering. Fig. 3 illustrates this quantum evolution: at \(s = 0\) it specializes to the classical deterministic lattice gas rule. The \(\Delta x = \Delta t \to 0\) limit of this discrete time evolution is the Dirac equation [14]; the \(\Delta x^2 = \Delta t \to 0\) limit is the Schrödinger equation [24].

QLGA models can be generalized to higher dimensions [24], and to include more particles [14,25,24], potentials [26,27] and various boundary conditions [28]. These are quantum models which we might try to simulate classically or quantum mechanically. Figures 4–6 show the results of classical simulations of plane waves, wave packets, and scattering off potential steps, respectively. These support the claim in the previous paragraph that QLGA are discrete models for quantum particles. In the next section we will see how they are also relevant to the question of simulating classical physical systems quantum mechanically.
Figure 5. Evolution of a wave packet in the general one dimensional QLGA [26].

Figure 6. Scattering of a wave packet from a potential step in the general one dimensional QLGA [26].
4. Diffusion

The evolution rule for a single particle QLGA bears some resemblance to a random walk. More precisely, it is the unitary version of a correlated random walk [29,30]—with \( \theta = \pi/4 \) the analogue of uncorrelated. In Fig. 5, for example, we can see evolution like that of a biased random walk, with a spreading Gaussian distribution. And, in fact, this is as close as possible—there is no quantum random walk in the sense of local, solely \( x \) dependent unitary evolution; the \( \alpha \) dependence must be included [31]. (This is not true for quantum processes continuous in time [32], which can also be used for computational purposes like the ones we are considering here [33,34].)

Consequently, there are differences. Diffusion approaches an equilibrium state, independently of the initial condition (on connected spaces). Unitary evolution of a single particle QLGA cannot: the distance \( \delta \) between successive states, defined by \( \cos \delta = \langle U\psi|\psi \rangle \) is constant, so the evolution of any state which is not an eigenstate of \( U \) with eigenvalue 1 does not converge. Each state implies a probability distribution on the lattice, given by

\[
P_i(x) = \text{prob}(x; t) = |\psi_{x,-1}(t)|^2 + |\psi_{x,+1}(t)|^2,
\]

so we can also ask if this converges. In fact, this probability distribution is constant for each eigenstate \( |\phi_i \rangle \) of \( U \) [26]. But since there exists \( T \in \mathbb{Z}_{>0} \) such that \( \lambda_i^T \sim 1 \) for all eigenvalues \( \lambda_i \) of \( U \), and hence \( U^T \psi \sim \psi \), for any initial state such that \( P_i \neq P_0 \), the probability distribution cannot converge either [35].

Aharonov, Ambainis, Kempe and Vazirani have shown, however, that the time average of the probability distribution does converge [35]: Expand the initial state \( |\psi \rangle = \sum_i a_i |\phi_i \rangle \) in terms of the eigenvectors of \( U \). Then \( U^t |\psi \rangle = \sum_i a_i \lambda_i^t |\phi_i \rangle \), so

\[
\text{prob}(x, \alpha; t) = \left| \sum_i a_i \lambda_i^t \langle x, \alpha | \phi_i \rangle \right|^2 = \sum_{i,j} a_i \bar{a}_j (\lambda_i \lambda_j)^t \langle x, \alpha | \phi_i \rangle \langle \phi_j | x, \alpha \rangle.
\]

Then the time average of the probability is

\[
\frac{1}{T} \sum_{t=0}^{T-1} \text{prob}(x, \alpha; t) = \sum_{i,j} a_i \bar{a}_j \langle x, \alpha | \phi_i \rangle \langle \phi_j | x, \alpha \rangle \frac{1}{T} \sum_{t=0}^{T-1} (\lambda_i \lambda_j)^t.
\]

For \( \lambda_i \neq \lambda_j \), the interior sum goes to 0 as \( T \to \infty \). This leaves only the terms in the sum for which \( \lambda_i = \lambda_j \), which are independent of \( T \). Thus the time average converges. In particular, for the one dimensional single particle QLGA, it converges to the uniform distribution which is the equilibrium distribution for diffusion on one dimensional lattices. That is, by measuring the position at random times we can simulate sampling from the equilibrium distribution of classical diffusion. Although this is not true for all graphs,
e.g., the Cayley graph of the nonabelian group $S_3$ [35], we have analyzed one example of discrete quantum simulation of a classical physical process.

5. Computational complexity

To be truly useful, the quantum computation should be more efficient than the corresponding classical computation. Classically, $O(N^2)$ steps of a random walk are required to approximate the equilibrium distribution on a lattice of size $N$. Aharonov et al. have shown that only $O(N \log N)$ steps of the single particle QLGA are required for equally accurate sampling [35], and more detailed calculations by Ambainis, Bach, Nayak, Vishwanath and Watrous [36] show that $O(N)$ steps suffice. The proofs of these results depend on careful estimates about, for example, the distribution of eigenvalues of $U$ and are somewhat involved. Simple simulations, however, provide a heuristic explanation for this quantum improvement. Fig. 7 shows the evolution of a QLGA particle initialized at $|\psi\rangle = (|0, -1\rangle + |0, +1\rangle)/\sqrt{2}$. Notice that the peaks of the probability distribution—indicated by the darkest squares in the plot—spread approximately linearly in time. This is the origin of $O(N)$ number of steps required for the QLGA to sample the equilibrium distribution.

Thus the QLGA provides a quadratic improvement in the number of steps required compared to a classical random walk. To verify a computational improvement, however, the QLGA must not require much more computation per step. Each step of the random walk requires a coin-flip, i.e., generation of a random number. Inasmuch as this is possible classically, it requires constant time, independent of $N$. Each step also requires the addition of the $\pm 1$ result of the coin flip to the current position. Since the latter is a $\log N$ bit integer, this requires $O(\log N)$ elementary operations. Thus the total number of computational steps to simulate one run of the random walk is $O(N^2 \log N)$. We can compare this with the computation required to compute the whole probability distribution by evaluating the Markov process. Although this could be computed by matrix multiplication at each step, the locality of the process means that we need only compute

$$\text{prob}(x; t+1) = \frac{1}{2}\text{prob}(x - 1; t) + \frac{1}{2}\text{prob}(x + 1; t)$$

for each lattice point. Since this has a constant computational cost per lattice point, evaluating the whole probability distribution for $O(N^2)$ steps takes $O(N^3)$ elementary operations.

For the QLGA we cannot run single trajectories since that would miss any interference between trajectories. Thus classical simulation of the QLGA must be like the Markov process calculation of the whole probability distribution of the random walk. Again, by locality, each step requires constant computation per lattice point. Thus evolution of the whole state for $O(N)$ steps takes $O(N^2)$ elementary operations. Taking the time average to approximate the diffusive equilibrium distribution requires another $O(N)$ factor, hence the same $O(N^3)$ elementary operations we found in the previous paragraph. Unsurprisingly,
Figure 7. Evolution of a single particle QLGA initialized at $x = 0$ in an equal superposition of left and right moving states [14]. Each square corresponds to one of the classical states; thus each lattice site is represented by two squares, one for each allowed velocity. The boundary conditions are periodic and time runs upward.
therefore, we have not discovered a faster classical algorithm for simulating the equilibrium distribution of diffusion.

Thus, to realize a quantum mechanical improvement we must be able to compute a single step $U$ of the QLGA evolution with fewer than $O(N)$ elementary operations on a quantum computer. Schematically, $U$ is implemented as shown in Fig. 8, acting on log $N$ qubits which encode the position on the lattice and a single qubit which encodes the velocity. The scattering operation $S$ acts on the last qubit and thus has constant cost per step. Advection is a shift operation, i.e., multiplication by a matrix which is diagonalized by the discrete Fourier transform $F_N$. The left shift $|x\rangle \mapsto |x-1\rangle$ is

$$
\begin{pmatrix}
0 & 1 & 0 & \cdots & 0 \\
0 & 0 & 1 & \cdots & 0 \\
\vdots & \ddots & \ddots & \ddots & \vdots \\
0 & 0 & 0 & \cdots & 1 \\
1 & 0 & 0 & \cdots & 0
\end{pmatrix} = F_N
\begin{pmatrix}
1 & \omega & \omega^2 & \cdots & \omega^{N-1} \\
\omega & \omega^2 & \cdots & \omega^{N-1} & 1 \\
\omega^2 & \cdots & \omega & \cdots & \omega^{N-1} \\
\cdots & \ddots & \ddots & \ddots & \ddots \\
\omega^{N-1} & \cdots & \cdots & \cdots & 1
\end{pmatrix}
F_N^\dagger.
$$

(1)

Here $\omega = e^{2\pi i/N}$ and the right shift diagonalizes to the same matrix using $F_N^\dagger$ rather than $F_N$. Assuming $N = 2^n$, the diagonal matrix can be implemented by $n = \log N$ single qubit phase rotations: $R_\omega \otimes R_{\omega^2} \otimes R_{\omega^4} \otimes \cdots \otimes R_{\omega^{N/2}}$.

Naïvely, the Fourier transform $F_N$ requires $O(N^2)$ operations. The classical FFT improves this to $O(N \log N)$. And perhaps the most fundamental result in quantum computation is that $F_N$ can be implemented with $O(\log^2 N)$ elementary quantum operations [9,37]. Thus we can implement the advection step with an $\alpha$-controlled-shift operation $|x, \alpha\rangle \mapsto |x + \alpha, \alpha\rangle$, where the shift (in each direction) is decomposed as in (1). Slightly less efficiently, but with the same $O(\log^2 N)$ complexity, we can implement it by acting by a left shift, and then acting by a two step right shift, conditioned on $\alpha = 1$; this circuit is shown in Fig. 9. Thus on a quantum computer we can sample the diffusion equilibrium with $O(N \log^2 N)$ elementary operations. This is an improvement by $O(N/\log N)$ over the classical random walk algorithm.
6. Discussion

We have seen that a classical physics problem—sampling from the equilibrium distribution of a diffusion process—is solvable more efficiently with a quantum computer than with a classical one, in the sense that the QLGA algorithm outperforms the random walk algorithm. The solution utilizes two of the fundamental quantum speedups: First, the quadratic improvement in the number of steps necessary for a QLGA simulation compared to a random walk simulation is reminiscent of the quadratic improvement of Grover’s quantum search algorithm—which he, in fact, describes as diffusion [21]. Second, the exponential improvement of the quantum Fourier transform over the FFT [9,37] provides a speedup in the advection step. Perhaps the simple problem considered here will inspire application of these techniques—or new ones—to speed up computation of other classical systems with quantum algorithms. For example, a natural generalization would be to analyze diffusion in a potential, for which a QLGA simulation has already been shown to evolve the mean of the distribution quadratically faster than does a classical (biased) random walk, in certain cases [30]. Successful development of such quantum algorithms should provide additional impetus to efforts towards building a quantum computer [38].

Acknowledgements

I thank Jean-Pierre Boon for his interest in these results and for inviting me to present them. I also thank Bruce Boghosian, Peter Coveney, Lov Grover, Manfred Krafczyk, Daniel Lidar, Seth Lloyd, Peter Love and Li-Shi Luo for useful conversations, before and during the workshop. This work was supported in part by the National Security Agency (NSA) and Advanced Research and Development Activity (ARDA) under Army Research Office (ARO) contract numbers DAAG55-98-1-0376 and DAAD19-01-1-0520, and by the Air Force Office of Scientific Research (AFOSR) under grant number F49620-01-1-0494.

References


