Simulation of Many-Body Fermi Systems on a Universal Quantum Computer

Daniel S. Abrams
Department of Physics, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139

Seth Lloyd
Department of Mechanical Engineering, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139
(Received 7 November 1996)

We provide fast algorithms for simulating many-body Fermi systems on a universal quantum computer. Both first and second quantized descriptions are considered, and the relative computational complexities are determined in each case. In order to accommodate fermions using a first quantized Hamiltonian, an efficient quantum algorithm for antisymmetrization is given. Finally, a simulation of the Hubbard model is discussed in detail. [S0031-9007(97)04120-3]

PACS numbers: 89.80.+h, 03.65.–w, 71.10.Fd, 89.70.+c

Since the discovery by Shor of a quantum algorithm for factoring in polynomial time [1], there has been tremendous activity in the field of quantum computation [2–24]. Recent results include the first experimental demonstrations of working quantum logic gates [2–4], quantum error-correcting codes [5], and many novel proposals for the design of actual quantum computers [3,6–9,25]. For a review of progress in quantum computation, see Refs. [10] or [11]. Despite these advances, the technical hurdles that stand in the way of factoring a large number on a quantum computer remain daunting [12–15]. But the problem of simulation—that is, the problem of modeling the full time evolution of an arbitrary quantum system—is less technologically demanding. While thousands of qubits and billions of quantum logic operations are needed to solve classically difficult factoring problems [16], it would be possible to use a quantum computer with only a few tens of qubits and a few thousand operations to perform simulations that would be classically intractable [17]. A quantum computer of this scale appears to be a realistic possibility.

Because the size of the Hilbert space grows exponentially with the number of particles, a full quantum simulation demands exponential resources on a classical computer. A system of only 100 spin $\frac{1}{2}$ particles, for example, requires 2^{100} complex numbers to merely describe a general spin state. It is clear that on a classical computer, a simulation of this system is in general intractable. The idea that a quantum computer might be more efficient than a classical computer at simulating real quantum systems was first proposed by Feynman [26], but he speculated that the problem of Fermi statistics might prevent the design of a universal quantum simulator. More recently, Lloyd has shown how a quantum computer is in fact an efficient quantum simulator [17]; other work on simulations can be found in [18–20]. In this Letter, we deal explicitly with the problem of fermions, in part by describing a quantum algorithm for antisymmetrization which executes in polynomial time. We also describe algorithms for quantum simulation of the Hubbard model in both first and second quantized formalisms. We show how the computer can be prepared in a state analogous to the initial state of a many-body Fermi system, how it can be programmed to simulate the system's time evolution, and how measurements can then be made on the computer to extract relevant information. Thus this paper provides for the first time a complete quantum algorithm for simulating a system of physical interest, and describes the only known algorithm, other than Shor's, that gains an exponential speed increase by exploiting quantum computation (excepting certain artificial problems constructed explicitly for this purpose [27–29]).

The quantum computer used to perform the simulation could rely on a variety of possible physical systems to store and process quantum information: for example, photons interacting via small cavity QED effects, electron spins, nuclear spins (NMR), or trapped ions [2,3,6,7]. The actual implementation of the quantum computer is not relevant, as long as it supports universal quantum computation [8,21–24,30–33] (although different physical implementations may be better or worse suited for different problems). We consider a simulation of n particles, each of which can be in any of m single particle states, labeled $1, \ldots, m$. These states might be sites in a lattice, or atomic orbitals, or plane waves, etc. The mapping of the system onto the qubits of the computer depends on whether we choose a first or second quantized description. In many respects, the second quantized form appears naturally well suited for quantum computation of Fermi systems: The occupation of each state must be either 0 or 1, which maps directly to the state of a quantum bit, or qubit [34]. In this case, the memory needed to map the state of the entire n particle system is m qubits (independent of n) [35]. To treat a first quantized Hamiltonian, we imagine a quantum word, or qu-word, as a string of qubits of length $\log_2 m$; one qu-word represents any integer in the range $1, \ldots, m$ and, consequently, the state of one particle. The state of the entire physical system being simulated can therefore be represented by n qu-words, or $n \log_2 m$ qubits. If $n \ll m$, a first quantized representation is more efficient. In either representation, if the system is in a superposition of many direct product states (as it is in general), then the quantum computer

 0031-9007/97/79(13)/2586(4)$10.00

will be in a corresponding superposition of states in order to represent the correct physical state of the system.

The problem of fermions is handled more easily in the second quantized form. The calculation begins with all qubits in the $|0\rangle$ state. We can prepare the system in any state that can be reached from the zero state using a relatively small number of quantum logic operations (that is, polynomial in m)—including those in which the n particles are localized in individual lattice sites, momentum eigenstates, thermal states of noninteracting particles, and states in which particles obey k-particle correlations or entanglements (for small k). Thus Fermi statistics do not pose any additional complications for system preparation in the second quantized formalism. Because the statistics are incorporated into the raising and lowering operators, the additional complications occur during time evolution. As a concrete example, we consider the Hubbard model, in which electrons move about a lattice of sites. Each site may be empty, or contain one electron (of either spin) or two electrons. Two qubits are therefore required to represent the four possible states of each site. The Hamiltonian for the system is

$$H = \sum_{i=1}^{m} V_0 n_{i\uparrow} n_{i\downarrow} + \sum_{\langle i,j\rangle\sigma} t_0 c_{i\sigma}^{*} c_{j\sigma} \,. \tag{1}$$

In the first term (potential energy), V_0 is the potential strength and $n_{i\sigma}$ is the number operator for fermions of spin σ at site i. In the second term (kinetic energy), the sum $\langle i,j\rangle$ indicates all (physically) neighboring pairs of sites, t_0 is the "hopping" strength, and $c_{i\sigma}$, $c_{i\sigma}^{*}$ are annihilation and creation operators, respectively, for fermions at site i and spin σ. The computer simulates the Hubbard model by performing the unitary operation $U = \exp[(-i/\hbar)(t)H]$ on suitably encoded states. This can be accomplished by splitting the Hamiltonian into a sum of local terms H_i and repeatedly applying the operators $U_i = \exp[(-i/\hbar)(t/n)H_i]$, to evolve local parts of the system over small time slices t/n, in series. (See Ref. [17] for a detailed discussion of this technique.) Thus it suffices to describe algorithms which perform the time evolution corresponding to each local term in the Hamiltonian. To effect the time evolution resulting from the potential energy terms $V = \sum_{i=1}^{m} V_0 n_{i\uparrow} n_{i\downarrow}$, consider each site one at a time; for each site, if it is occupied by two electrons (of opposite spin), advance the phase of the state by $[(-i/\hbar)(t/n)V_0]$. This subroutine requires $O(m)$ operations.

To calculate the effect of the hopping terms $\sum_{\langle i,j\rangle\sigma} \times t_0 c_{i\sigma}^{*} c_{j\sigma}$ requires a slightly more complicated algorithm. For each σ and i,j pair, count the number of occupied states which fall between i and j when the system is written in second quantized form. A flag is set to the parity of this number, which indicates whether or not a change of sign is introduced when hopping between the two sites, as required by the definitions of the raising and lowering operators. It is now easy to perform the time evolution U_i corresponding to the Hermitian piece of the Hamiltonian $T_i = t_0(c_{j\sigma}^{*} c_{k\sigma} + c_{k\sigma}^{*} c_{j\sigma})$ by simply diagonalizing the Hamiltonian in the two qubit space i,j, and advancing the phase of the eigenstates. Assuming that the number of neighbors is a constant, the loops execute $O(m)$ times. It takes $O(m)$ operations to count the occupancy of the intervening states, and it follows that the entire algorithm for time evolving the second quantized Hubbard model executes in $O(m^2)$ quantum logic operations.

Fermi statistics are more difficult to handle in the usual first quantized description, because the initial state of the quantum computer must be antisymmetrized. As there are $n!$ states in the superposition, one needs a fast quantum algorithm for generating this superposition if the approach is to be tractable. The algorithm we describe takes an "unsymmetrized" state and generates an antisymmetrized superposition of $n!$ states in $O(n^2(\ln m)^2)$ time. Note that without further restriction, antisymmetrization is an irreversible process and cannot be performed by a reversible quantum computer: There are $n!$ input states which correspond to the same antisymmetrized state (modulo an overall phase). We therefore add the requirement that the input state must be ordered; i.e., the number representing the state of particle i is less than that of particle $i+1$, for all $i < n-1$. The correspondence between an ordered n-tuple of qu-words and an antisymmetrized superposition is one to one. Thus, system preparation in the first quantized formalism begins by first initializing the computer into an unsymmetrized state and then antisymmetrizing that state. The system can be easily prepared in any (unsymmetrized) direct product state by merely placing each particle in the appropriate single particle state. These single particle states include those which are localized in position space, momentum space [obtained with a quantum fast Fourier transform (FFT)], and thermal states. The system can also be initialized into states with arbitrary k-particle correlations or entanglements by performing quantum logic operations in the appropriate k-particle space, requiring only $O(m^{2k})$ operations in the general case, and often far fewer.

Antisymmetrization is accomplished in four main steps, summarized below. A more detailed description will be published elsewhere [36].

Step I: Initialization of the input state.—We define three registers A, B, and C, each consisting of n qu-words ($n \log_2 m$ qubits). The qubits in register A are initialized to the unsymmetrized input state $|\Psi\rangle$. The algorithm is unaffected if this state is a superposition of several ordered n-tuples.

Step II: Generating $n!$ states.—We create the following state in register B:

$$\frac{1}{n!}\left(\sum_{1}^{n}|i\rangle\right)\otimes\left(\sum_{1}^{n-1}|i\rangle\right)\otimes\left(\sum_{1}^{n-2}|i\rangle\right)\otimes\cdots \otimes(|2\rangle+|1\rangle)\otimes|1\rangle\,. \tag{2}$$

This is accomplished with $O(n(\ln m)^2)$ steps by performing appropriate rotations on each qubit, one at a time [37].

Step III: Transform into permutations of natural numbers.—The goal of this third step is to transform register B into the state $(1/n!)\sum_{\sigma\in S_n}|\sigma(1,\ldots,n)\rangle$, where S_n is the symmetric group of permutations on n objects. This is an equal superposition of the states representing all the permutations of the first n natural numbers. The basic idea is as follows: Let $B[i]$ indicate the ith qu-word in register B; map $B[i]$ into a qu-word $B'[i]$ by setting $B'[1] = B[1]$, and $B'[i]$ equal to the $B[i]$th natural number which is not contained in the set $\{B'[1],\ldots,B'[i-1]\}$, for $i > 1$. This transformation is effected with $O(n^2 \ln m)$ operations.

To prepare for the last step of the algorithm the n-tuple $1,2,3,\ldots,n$ is then assigned to register C, leaving the computer in the state

$$\frac{1}{n!}|\Psi\rangle \otimes \left(\sum_{\sigma\in S_n}|\sigma(1,\ldots,n)\rangle\right) \otimes |1,\ldots,n\rangle . \quad (3)$$

Step IV: Sorting and unsorting.—The algorithm proceeds with a series of sorting and unsorting operations. A string of "scratch" qubits is required so that the sorting operations are reversible. Any sorting algorithm can be used; we suggest using a Heap sort, because it requires $O(n \ln n)$ operations in all cases and only $n \log_2 n$ scratch qubits. The first sort orders register B with a series of exchanges and scrambles A and C with the same series of exchanges. At this point, one has already obtained a symmetrized superposition of the input states, but it is entangled with many other qubits. One can antisymmetrize by counting the number of exchanges made during the sorting operation and advancing the phase of that component of the superposition by π if this number is odd [38]. The algorithm continues by reversing the sort on register B, but leaving registers A and C unchanged. The qubits contained in B and C are then redundant: In each component of the superposition, if $B[i] = n$, then $C[n] = i$. This redundancy allows B to be set to zero reversibly. By then sorting A and C together, eliminating C, and unsorting, one obtains the desired antisymmetrized state. Note that in the final unsorting operation, the algorithm relies upon the fact that the ordering of the input state $|\Psi\rangle$ was stipulated to be the same as the ordering of the integers $1,\ldots,n$ in register C (so that antisymmetrization would be reversible); if this were not the case, the algorithm would fail. The entire process is completed in $O(n^2(\ln m)^2)$ operations.

Because the input state is now antisymmetrized, time evolution is in principle straightforward. Using the same technique as before, the Hamiltonian is split into a sum of terms H_i and the corresponding time evolution operators $U_i = \exp[(-i/\hbar)(t/n)H_i]$ are applied to the state in series. (The antisymmetry of the state will not be affected by truncation errors that occur during this process; although each individual U_i does not preserve antisymmetry, their products do exactly. For a more detailed discussion of errors that occur during time evolution, see [17].) Each U_i can be performed by an appropriate series of quantum logic operations; the actual sequence of gates required can be determined by inverting the Campbell-Baker-Hausdorff formula. Using this procedure, $O(m^2)$ steps are required to perform an arbitrary one particle operator U_i, and $O(m^4)$ operations are required to perform an arbitrary two particle operator. It is therefore possible to simulate in polynomial time any system of fermions (as long as the Hamiltonian does not include terms which involve k-particle interactions, where k is a large number of order n).

For the special case of the Hubbard model, the simplicity of the Hamiltonian allows one to perform each U_i in only $O((\ln m)^2)$ steps. To begin, consider the Hubbard model Hamiltonian in its first quantized form:

$$H = \sum_{i=1}^{n} T_i + \frac{1}{2}\sum_{k\neq 1} V_{kl} , \quad (4)$$

where $\langle i\sigma|T|j\sigma\rangle = t_0\delta_{\langle i,j\rangle}$ and $\langle i\uparrow, i\downarrow|V|i\uparrow, i\downarrow\rangle = V_0$. As before, the potential energy terms are easier because they are diagonal. For a given pair of particles, perform a controlled rotation if they are at the same site. In order to perform the time evolution resulting from the kinetic energy terms, we focus on one particle at a time. For each particle, the idea is to decompose the kinetic energy terms into a sum of block diagonal matrices and then diagonalize the subblocks in each matrix in parallel [39]. For simplicity of explanation, we describe here only a 1D Hubbard model and ignore spin. In this case the kinetic energy part of the Hamiltonian can be written

$$T = h(1,2) + h(2,3) + h(3,4) + \cdots + h(m-1,m) , \quad (5)$$

where $h(i,j)$ is the piece of the Hamiltonian that corresponds to hopping between sites i and j. Writing $T = T_1 + T_2$, we define

$$T_1 = h(1,2) + h(3,4) + h(5,6) + \cdots , \quad (6a)$$

$$T_2 = h(2,3) + h(4,5) + h(6,7) + \cdots . \quad (6b)$$

The operators T_1 and T_2 are in block diagonal form. To diagonalize each matrix (separately), perform quantum logic operations on each state to transform the state number into two quantum numbers labeling the block and the location within the block (0 or 1). For example, to diagonalize T_1, map the state $|n\rangle$ into $|(n+1) \text{ div } 2, n \bmod 2\rangle$. Because T_1 is block diagonal—and because all states within the same block have their first quantum number in common—the action of T_1 takes place entirely within the space of the second quantum number. In this one qubit space it is simply the matrix $t_0\left(\begin{smallmatrix}0 & 1\\ 1 & 0\end{smallmatrix}\right)$. Thus all the blocks can be diagonalized in parallel by diagonalizing this trivial 2×2 matrix in the one qubit space of the second quantum number. Each state in the superposition is then advanced by the appropriate phase, and all the previous steps are reversed. This algorithm requires only $O((\ln m)^2)$ quantum logic operations.

Finally, we consider what information can be extracted from a quantum many-body simulation. It is obviously impossible to obtain the entire wave function: Rather,

the "answer" is obtained by performing a series of measurements on the qubits, one at a time. Each such measurement will yield either $|0\rangle$ or $|1\rangle$. It is thus possible to measure any physical quantity that can be expressed in terms of such local variables. To obtain useful information about the physics of the simulated system, one must initialize the quantum computer, simulate time evolution, make a measurement, and then repeat this process a sufficient number of times to acquire a statistically significant result. For example, the electronic charge density distribution can be obtained in the second quantized representation by performing measurements at each site to determine the probability of occupancy. The number of such measurements required to obtain some desired accuracy ε varies as ε^{-2} (i.e., the accuracy grows as a polynomial function of the number of trials). In the first quantized representation, the same result is obtained by measuring the location of a given particle and generating a histogram of locations from repeated trials. It is straightforward to obtain two-particle correlation functions and even k-particle correlations using a similar approach (requiring roughly $O(\varepsilon^{-2}\delta^k)$ trials, where δ is the density of points in the histogram and ε is the desired accuracy). The momentum distribution function can be obtained by performing a quantum FFT before sampling the wave function. From the one- and two-particle densities and momentum distribution, one can obtain the expected energy. A variety of techniques can be used to obtain other information: For example, one can obtain scattering amplitudes by simulating the motion of an electron through a charged medium and measuring the probability of its emerging with different momenta. Or, one can perform a quantum simulated annealing by time evolving the system in contact with a simulated heat bath and then using the previous techniques to obtain information about the system's ground state.

In summary, we have shown how a universal quantum computer can be used to efficiently simulate systems consisting of many fermions. Depending on the particular problem, it may be preferable to employ first or second quantized notation ($n \log_2 m$ vs m qubits). A general algorithm for creating an antisymmetrized superposition of states has been described. We have also demonstrated algorithms which will simulate the Hubbard model, requiring $O(n^2(\ln m)^2)$ quantum logic operations in first quantized form, and $O(m^2)$ operations in second.

D. S. A. acknowledges support from an NDSEG fellowship and thanks J. D. Joannopoulos, Tomas Arias, and Steve Simon for helpful discussions. Portions of this research were supported by the ONR and through the DARPA initiative on Quantum Information and Computation (QUIC) administered by ARO.

[1] P. Shor, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science,* edited by S. Goldwasser (IEEE Computer Society, Los Alamos, CA, 1994), p. 124.
[2] C. Monroe *et al.,* Phys. Rev. Lett. **75**, 4714 (1995).
[3] Q. A. Turchette *et al.,* Phys. Rev. Lett. **75**, 4710 (1995).
[4] P. Domokos *et al.,* Phys. Rev. A **52**, 3554 (1995).
[5] P. W. Shor, Phys. Rev. A **52**, 2493 (1995).
[6] J. I. Cirac and P. Zoller, Phys. Rev. Lett. **74**, 4091 (1995).
[7] S. Lloyd, Science **261**, 1569 (1993).
[8] T. Sleator and H. Weinfurter, Phys. Rev. Lett. **74**, 4087 (1995).
[9] A. Barenco *et al.,* Phys. Rev. Lett. **74**, 4083 (1995).
[10] D. P. DiVincenzo, Science **270**, 255 (1995).
[11] A. Ekert and R. Jozsa, Rev. Mod. Phys. **68**, 733 (1996).
[12] W. G. Unruh, Phys. Rev. A **51**, 992 (1995).
[13] G. M. Palma, K. A. Suominen, and A. K. Ekert, Proc. R. Soc. London A **452**, 567 (1996).
[14] R. Landauer, Philos. Trans. R. Soc. London A **353**, 367 (1995).
[15] I. L. Chuang *et al.,* Science **270**, 1633 (1995).
[16] D. Beckman *et al.,* Phys. Rev. A **54**, 1034 (1996).
[17] S. Lloyd, Science **273**, 1073 (1996).
[18] S. Wiesner (to be published).
[19] C. Zalka (to be published).
[20] B. Boghosian and W. Taylor, Phys. Rev. E (to be published).
[21] S. Lloyd, Phys. Rev. Lett. **75**, 346 (1995).
[22] D. Deutsch, A. Barenco, and A. Ekert, Proc. R. Soc. London A **449**, 669 (1995).
[23] A. Yao, in *Proceedings of the 36th Annual Symposium on Foundations of Computer Science,* edited by S. Goldwasser (IEEE Computer Society, Los Alamos, CA, 1995), p. 352.
[24] A. Barenco *et al.,* Phys. Rev. A **52**, 3457 (1995).
[25] K. Obermayer, W. G. Teich, and G. Mahler, Phys. Rev. B **37**, 8096 (1988).
[26] R. P. Feynman, Int. J. Theor. Phys. **21**, 467 (1982).
[27] E. Bernstein and U. Vazirani, in *Proceedings of the 25th Annual ACM Symposium on Theory of Computing* (ACM, New York, 1993), p. 11.
[28] D. Simon, in *Proceedings of the 35th Annual Symposium on Foundations of Computer Science* (Ref. [1]), p. 116.
[29] D. Deutsch and R. Jozsa, Proc. R. Soc. London A **439**, 553 (1992).
[30] D. Deutsch, Proc. R. Soc. London A **400**, 97 (1985).
[31] D. Deutsch, Proc. R. Soc. London A **425**, 75 (1989).
[32] N. Margolus, Ann. N.Y. Acad. Sci. **480**, 487 (1986).
[33] P. Benioff, Phys. Rev. Lett. **48**, 1581 (1982).
[34] B. Schumacher, Phys. Rev. A **51**, 2738 (1995).
[35] For Bose particles, where the occupation of each state can be any integer in the range $0, \ldots, n$, $\log n$ qubits are needed to represent each state, yielding a total of $m \log n$ qubits.
[36] D. S. Abrams and S. Lloyd (to be published).
[37] For example, if we represent 8 with $|000\rangle$, the state $(\mathrm{Sum}|1\rangle \cdots |8\rangle)$ is generated by rotating each qubit into the state $|0\rangle + |1\rangle$. The states $(\mathrm{Sum}|1| \cdots |j\rangle)$ for $j \neq 2^i$ can be easily generated in a similar manner by using controlled rotations that are conditioned on previous qubits.
[38] For Bosons, simply leave the symmetrized state as is and proceed.
[39] Because one often encounters block diagonal operators in many different types of problems, it is likely that the technique described here may have a much wider range of utility than in simulations of the Hubbard model.