ELSEVIER

Mathematics and Computers in Simulation 47 (1998) 143–152

MATHEMATICS AND COMPUTERS IN SIMULATION

Monte Carlo simulation of quantum computation[1]

N.J. Cerf*, S.E. Koonin

W.K. Kellogg Radiation Laboratory, 106-38, California Institute of Technology, Pasadena, CA 91125, USA

Abstract

The many-body dynamics of a quantum computer can be reduced to the time evolution of non-interacting quantum bits in auxiliary fields using the Hubbard–Stratonovich representation of two-bit quantum gates in terms of one-bit gates. This makes it possible to perform the stochastic simulation of a quantum algorithm based on the Monte Carlo evaluation of an integral of dimension polynomial in the number of quantum bits. As an example, the simulation of the quantum circuit for the fast Fourier transform is discussed. © 1998 IMACS/Elsevier Science B.V.

Keywords: Quantum Monte Carlo methods; Quantum computers

1. Introduction

The potential use of quantum computers for solving certain classes of problems has recently received a considerable amount of attention (see, e.g., [1–3] for a comprehensive review). Several quantum algorithms have been developed, such as quantum factoring [4], having the potential for revolutionizing computer science. The purpose of this paper is to explore the application of a Monte Carlo method that has been developed in the context of quantum many-body systems to the simulation of quantum computers [5]. Quantum computers can be seen as peculiar quantum many-body systems that evolve according to a non-local time-dependent interaction so as to carry out a ''computation''. The component quantum bits (qubits) interact via a sequence of quantum gates, each performing a prescribed unitary transformation (rotation, Hadamard transformation, controlled NOT, controlled phase, etc.) [6]. Two-bit (or n-bit) gates therefore effect non-local interactions between qubits, and the ''quantum algorithm'' (characterized by a network of quantum gates) corresponds to a specific sequence of unitary transformations, i.e., a time-dependent interaction. Numerous methods have been developed for years in order to treat general quantum many-particle systems (see, e.g., [7]). It is therefore intriguing to examine whether the application of the same methods to quantum computers might be similarly successful. We focus here on a stochastic approach based on the Hubbard–Stratonovich transformation [8] which has been shown to be suitable for the

* Corresponding author. E-mail: cerf@krl.caltech.edu

[1]Presented at the IMACS Seminar on Monte Carlo Methods, Brussels, 1–3 April 1997.

PII S0378-4754(98)00099-8

description of quantum many-body systems (see, e.g., [9,10]). The central idea of this approach is to replace the many-body propagator for the entire quantum computer (of say L "interacting" qubits) with L one-bit propagators in fluctuating auxiliary fields, thereby "decoupling" the qubits. More specifically, solving the quantum dynamics of the L-bit computer in a very high-dimensional Hilbert space ($d = 2^L$) reduces to evaluating a high-dimensional – but polynomial in L – integral over auxiliary fields. The latter is then approximated by a stochastic method.

2. Quantum computer as a many-particle system

Consider a quantum computer consisting of a register of L qubits supplemented with a quantum algorithm, defined as a sequence of G quantum gates. The total unitary transformation characterizing the quantum computation is thus expressed as an ordered product of the operators (note the product from right to left):

$$U \equiv \prod_{g=1}^{G} U_g = U_G \cdots U_1 \tag{1}$$

where U_g is the unitary transformation performed by the gth gate and G is the total number of gates. It has been shown that *two-bit* gates are universal, i.e., quantum gates operating on one and two qubits are sufficient to construct a general quantum circuit [6,11–13]. Therefore, we restrict ourselves to the simulation of quantum circuits made of G two-bit gates (U_g being a two-bit gate acting on qubits a_g and b_g), keeping in mind that an arbitrary quantum computation can be achieved with an appropriate sequence of such gates. (Obviously, one-bit gates can always be incorporated into two-bit gates.) Note that an efficient quantum algorithm must have a G polynomial in L. For example, the quantum fast Fourier transform (FFT) circuit [14] used in quantum factoring [4] requires $G = L(L-1)/2$ two-bit gates. A two-bit quantum gate that effects a unitary transformation on qubits a_g and b_g can be written generically as the two-bit operator

$$U_g = \mathrm{e}^{-\mathrm{i}\alpha_g A_g B_g} \tag{2}$$

where α_g is a real number, and A_g and B_g are two *commuting* one-bit Hermitian operators referring to the qubits involved in the quantum gate (i.e., the operator A_g (B_g) affects qubit a_g (b_g)). For example, the controlled-NOT gate [6] acting on qubit a (as a control) and qubit b (as a target) has $\alpha = \pi/4$, $A = 1 - \sigma_z$, and $B = 1 - \sigma_x$, with σ_x and σ_z being Pauli matrices. The Hubbard–Stratonovich representation of U_g is obtained by writing the identity

$$\mathrm{i}\alpha_g A_g B_g = \mathrm{i}\alpha_g (A_g - \tau_g)(B_g - \sigma_g) + \mathrm{i}\alpha_g \sigma_g A_g + \mathrm{i}\alpha_g \tau_g B_g - \mathrm{i}\alpha_g \sigma_g \tau_g \tag{3}$$

where σ_g and τ_g are *real* auxiliary fields corresponding to the gth gate, and then integrating the exponential of Eq. (3) over σ_g and τ_g, resulting in

$$U_g = \frac{|\alpha_g|}{2\pi} \int_{-\infty}^{\infty} \mathrm{d}\sigma_g \, \mathrm{d}\tau_g \, \mathrm{e}^{\mathrm{i}\alpha_g \sigma_g \tau_g} \, \mathrm{e}^{-\mathrm{i}\alpha_g \sigma_g A_g} \, \mathrm{e}^{-\mathrm{i}\alpha_g \tau_g B_g} \tag{4}$$

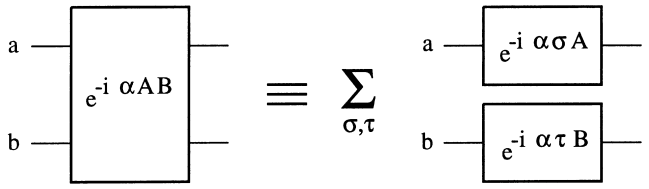

Fig. 1. Hubbard–Stratonovich representation of a two-bit quantum gate in terms of two one-bit gates in fluctuating auxiliary fields σ and τ.

This expression is most important because the *two-bit* gate U_g is represented as an infinite sum of the products of (field-dependent) *one-bit* gates, $e^{-i\alpha_g\sigma_g A_g}$ and $e^{-i\alpha_g\tau_g B_g}$. For a given value of the fields σ_g and τ_g, the two qubits a_g and b_g act as non-interacting particles and evolve independently (they do not become entangled when initially prepared in a product state).[2] Only the sum over fields creates a "coupling" between them as pictured in Fig. 1. As a consequence, for a given set of σ_g's and τ_g's, the time-evolution effected by the whole quantum circuit can be computed separately for each qubit: one calculates the time-evolution of L qubits in L two-dimensional Hilbert spaces rather than the time-evolution of a single quantum state (the state of the entire computer) in the full 2^L-dimensional Hilbert space. This exponential reduction of the size of the Hilbert space appears clearly when writing the total unitary transformation for the quantum circuit

$$U = \int D\sigma \exp\left(i \sum_g \alpha_g \sigma_g \tau_g\right) \underbrace{\prod_g V_g(\sigma_g) W_g(\tau_g)}_{U[\sigma]} \tag{5}$$

where $D\sigma = \prod_g (|\alpha_g|/2\pi)\, d\sigma_g\, d\tau_g$ is the measure over the auxiliary fields $\sigma_1, \ldots, \sigma_G$ and $\tau_1, \ldots, \tau_G$, and $U[\sigma]$ is the total unitary transformation for a given "path" σ in the auxiliary-field space. Here, $V_g(\sigma_g) \equiv e^{-i\alpha_g\sigma_g A_g}$ and $W_g(\tau_g) \equiv e^{-i\alpha_g\tau_g B_g}$ stand for the unitary transformation performed by the one-bit gate acting separately on qubit a_g and b_g, respectively. (The one-bit gates V_g and W_g replacing the gth two-bit gate U_g depend on the auxiliary fields σ_g and τ_g.) Eq. (5) involves only *one-bit* operators, and therefore describes the time-evolution of L *non-interacting* qubits (averaged over auxiliary fields). The operator $U[\sigma]$ is more conveniently written as a product of one-bit operators over the L qubits,

$$U[\sigma] = \prod_{l=1}^{L} U^{[l]}[\sigma], \tag{6}$$

where the one-bit operator $U^{[l]}$ describing the overall evolution of the lth qubit is expressed as the ordered product

$$U^{[l]}[\sigma] = \prod_g U_g^{[l]}(\sigma_g, \tau_g) \tag{7}$$

[2] Several alternative Hubbard–Stratonovich representations of a two-bit quantum gate requiring only one auxiliary field per gate can be written, the drawback being that the involved one-bit transformations are in general non-unitary.

with

$$U_g^{[l]}(\sigma_g, \tau_g) = \begin{cases} V_g(\sigma_g) & \text{if } l = a_g, \\ W_g(\tau_g) & \text{if } l = b_g, \\ 1 & \text{otherwise.} \end{cases} \tag{8}$$

The drawback of this exponential reduction in the Hilbert space is obviously the $2G$-dimensional integral over fields in Eq. (5), which can only be approximated by a numerical method in general. The underlying idea of a stochastic method is to compute only the dominant terms in this integral, i.e., to consider the paths in the auxiliary-field space that contribute the most to it, assuming that this yields a good estimate of the exact integral. Several (more or less efficient) Monte Carlo techniques can be thought of for sampling these paths, but a generic ''sign'' problem is likely to occur to the complex weight in Eq. (5). This will be discussed later on. However, the central point here is that the dimension $2G$ of this integral is polynomial in the dimension of the problem, i.e., a polynomial in the number of qubits L, at the condition that $G = \text{poly}(L)$. The latter condition is fulfilled for any efficient quantum algorithm, suggesting that the Monte Carlo simulation of a quantum computer is promising provided that the ''sign'' problem is circumvented.

3. Stochastic simulation of a quantum computer

Consider the stochastic calculation of the quantities of interest in a general quantum computation. In the context of quantum many-body systems, stochastic methods are especially appropriate for calculating quantum expectation values, so that our goal is to express the output of the quantum computation as an observable. Assume that the quantum computer is initially in a product state $|0_1 0_2 \cdots 0_L\rangle$. (If this is not the case, the first step of the computation should simply be the preparation of the correct initial state from $|0_1 0_2 \cdots 0_L\rangle$.) The quantum computation (i.e., the unitary transformation U) is implemented by a quantum circuit acting on this initial state. The final step of the quantum algorithm is then to measure a set of ''output'' qubits (not necessarily all the L qubits). We restrict ourselves here to quantum algorithms that provide a *deterministic* result (unlike quantum factoring). We assume therefore that the output bits are in a product state so that they can be measured individually (i.e., one can perform an *inclusive* measurement for each of them separately). The most general observable O with vanishing variance (deterministic output) consists then in a product of one-bit observables, and several such O's can be measured simultaneously. (If the output bits are not in a product state, one should extend the quantum computation with a unitary transformation mapping the entangled final state into a product state.[3]) More generally, the quantum many-particle simulation approach allows us to prescribe the value of certain qubits in the output register. We separate the L output qubits into L_m measured qubits, L_p prescribed qubits, and $L_\text{t}{=}L{-}L_\text{m}{-}L_\text{p}$ traced over qubits (i.e., ''scratch'' qubits that are necessary to make the overall computation unitary, but are not observed

[3]It is not clear whether this requirement makes the extended quantum computation much harder in a general case. At least, some quantum algorithms are known to provide a deterministic result, such as Grover's quantum search algorithm [15], so that the output bits are then in a product state. Note that the same requirement must be met for the recently suggested realization of quantum computers using NMR experiments [16].

in the final measurement). The observable can then be written as

$$O = \prod_{\{m\}}^{L_m} O_m \tag{9}$$

where O_m is a one-bit observable acting on a qubit m. Consequently, the result of a quantum computation can be written as the expectation value of the observable O:

$$\langle O \rangle = \frac{\langle 0_1 \cdots 0_L | U^\dagger O P U | 0_1 \cdots 0_L \rangle}{\langle 0_1 \cdots 0_L | U^\dagger P U | 0_1 \cdots 0_L \rangle} \tag{10}$$

where we define the projector P as

$$P = \prod_{\{p\}}^{L_p} P_p \tag{11}$$

and $P_p \equiv |\pi_p\rangle\langle\pi_p|$ is a projector on the prescribed value π_p for qubit p. Note that it is crucial to consider a quantum algorithm such that the variance of O vanishes when the prescribed qubits have the correct value, so that Eq. (10) yields the *deterministic* output of the quantum computation. The only variance in the simulated output will be the statistical noise resulting from the stochastic evaluation of Eq. (5).

The central point now is that, using Eqs. (5) and (6), the numerator and the denominator in Eq. (10) can be expressed in terms of an infinite sum of products of L one-bit matrix elements for each qubit,

$$\langle O \rangle = \frac{\int D\sigma D\sigma' \, \mathrm{e}^{\mathrm{i} \sum_g \alpha_g (\sigma_g \tau_g - \sigma'_g \tau'_g)} \prod_{l=1}^{L} \langle 0_l | U^{\dagger[l]}[\sigma'] O^{[l]} P^{[l]} U^{[l]}[\sigma] | 0_l \rangle}{\int D\sigma D\sigma' \, \mathrm{e}^{\mathrm{i} \sum_g \alpha_g (\sigma_g \tau_g - \sigma'_g \tau'_g)} \prod_{l=1}^{L} \langle 0_l | U^{\dagger[l]}[\sigma'] P^{[l]} U^{[l]}[\sigma] | 0_l \rangle} \tag{12}$$

where σ' represents the set of auxiliary fields (σ'_g and τ'_g) used in the Hubbard–Stratonovich expression of $U^\dagger$. This can be written more concisely as

$$\langle O \rangle = \frac{\int D\sigma D\sigma' \exp(-\mathrm{i} S[\sigma, \sigma']) O[\sigma, \sigma']}{\int D\sigma D\sigma' \exp(-\mathrm{i} S[\sigma, \sigma'])} \tag{13}$$

where the (complex) action is defined as

$$S[\sigma, \sigma'] = -\sum_g \alpha_g (\sigma_g \tau_g - \sigma'_g \tau'_g) + \mathrm{i} \sum_{l=1}^{L} \ln \langle 0_l | U^{\dagger[l]}[\sigma'] P^{[l]} U^{[l]}[\sigma] | 0_l \rangle \tag{14}$$

The operator $P^{[l]}$ is a one-bit projector if the lth qubit is prescribed, and the unit operator otherwise. The estimator of O is

$$O[\sigma, \sigma'] = \prod_{l=1}^{L} \frac{\langle 0_l | U^{\dagger[l]}[\sigma'] O^{[l]} P^{[l]} U^{[l]}[\sigma] | 0_l \rangle}{\langle 0_l | U^{\dagger[l]}[\sigma'] P^{[l]} U^{[l]}[\sigma] | 0_l \rangle}, \tag{15}$$

where $O^{[l]}$ is the l-th one-bit component of the observable O if the lth qubit is measured, and the unit operator otherwise. Note that the L matrix elements in the right-hand side of Eq. (14) are for single

qubits, so that the calculation of the action involves $\sim 4G$ products of non-unit 2×2-matrices. (There are two fields per gate, and the Hermitian conjugate $U^\dagger$ must be considered together with U.) The calculation of Eq. (15) requires essentially the same operations.

4. Sampling of the auxiliary-field paths

Let us now consider the stochastic evaluation of Eq. (13) based on a sampling of the paths (set of σ_g's and τ_g's) that contribute the most to the integral. The simplest possibility is to perform an importance sampling of the paths according to the weight $|e^{-iS}|$. (Note that this weight is not equal to 1 since S is generally complex.) This can be done for example by using the Metropolis method [17]. A random walk in the auxiliary-field space is simulated such that the limit distribution of sampled paths is proportional to $|e^{-iS}|$. This makes it possible to write $\langle O\rangle$ as a ratio of Monte Carlo averages:

$$\langle O\rangle \sim \frac{\langle e^{-i\,\mathrm{Re}\,S[\sigma,\sigma']}O[\sigma,\sigma']\rangle_{\sigma,\sigma'}}{\langle e^{-i\,\mathrm{Re}\,S[\sigma,\sigma']}\rangle_{\sigma,\sigma'}} \tag{16}$$

where $\langle\cdot\rangle_{\sigma,\sigma'}$ stands for the simulation average over auxiliary-field paths. A test of this approach has been carried out, showing that the term $e^{-i\ \mathrm{Re}\ S}$ generally makes the (averaged) numerator and the denominator of Eq. (16) exceedingly small. Unless this "sign" problem can be overcome, the standard Metropolis method seems therefore to be inefficient in this context.[4] Since the weight of the paths in Eq. (13) is complex (this is at the heart of the sign problem) a more promising possibility is the recourse to a simulation based on the complex Langevin equation [18,19]. In the Langevin algorithm (see, e.g., [20,21]), paths distributed according to the "complex probability distribution" $\sim e^{-iS}$ can be generated, allowing the computation of Eq. (13) as a time-average over a guided random walk for the fields in the *complex* plane. In the case of interest here, the random walk for a field σ_g is the solution of the stochastic differential equation

$$\frac{d\sigma_g}{dt} = -\frac{i}{2}\frac{\partial S}{\partial \sigma_g} + \eta_g(t) \tag{17}$$

where t is a fictitious time (simulation time) and η_g is a (real) Gaussian white noise satisfying $\langle\eta_g(t)\rangle = 0$ and $\langle\eta_g(t)\eta_g(t')\rangle = \delta(t-t')$. The first term in the right-hand side of Eq. (17) can be seen as a "string" force which keeps σ_g close to the value for which the action S is extremum, while the "noise" term is responsible for the sampling of a region in auxiliary-field space around this extremum. Although a general proof of the convergence of the complex Langevin simulation does not exist [20], it turns out to work very nicely for a number of systems (the convergence is related to the location of the repulsive points of the Langevin dynamics). The Langevin simulation yields then a stochastic estimate of the output of the quantum computer,

$$\langle O\rangle \sim \frac{1}{T}\int_t^{t+T} dt\, O[\sigma(t),\sigma'(t)] \tag{18}$$

[4] This numerical test has been performed on a small quantum circuit ($L = 3$, $G = 4$) using a one-field per gate Hubbard–Stratonovich transformation, but the "sign" problem for longer circuits most probably remains.

which is calculated by averaging $O[\sigma, \sigma']$ for a sufficiently long random walk. Using Eq. (14), the time-derivative of the field σ_g can be written explicitly as

$$\frac{\mathrm{d}\sigma_g}{\mathrm{d}t} = \frac{\mathrm{i}}{2}\alpha_g\tau_g + \frac{1}{2}R_g[\sigma, \sigma'] + \eta_g(t) \tag{19}$$

with

$$R_g[\sigma, \sigma'] = \prod_{l=1}^{L} \frac{\langle 0_l | U^{\dagger[l]}[\sigma'] P^{[l]}\, \mathrm{d}U^{[l]}[\sigma]/\mathrm{d}\sigma_g | 0_l \rangle}{\langle 0_l | U^{\dagger[l]}[\sigma'] P^{[l]} U^{[l]}[\sigma] | 0_l \rangle} \tag{20}$$

One single term ($l = a_g$) differs from one in this product as only the one-bit gate acting on qubit a_g depends on σ_g (see Eqs. (7) and (8)). One has

$$\frac{\mathrm{d}U^{[a_g]}[\sigma]}{\mathrm{d}\sigma_g} = \prod_{g'} \tilde{U}_{g'}^{[\alpha_g]}(\sigma_{g'}, \tau_{g'}) \tag{21}$$

with

$$\tilde{U}_{g'}^{[a_g]}(\sigma_{g'}, \tau_{g'}) = \begin{cases} -\mathrm{i}\alpha_g A_g U_g^{[a_g]}(\sigma_g, \tau_g) & \text{if } g' = g, \\ U_{g'}^{[a_g]}(\sigma_{g'}, \tau_{g'}) & \text{otherwise.} \end{cases} \tag{22}$$

The calculation of the derivative $\mathrm{d}\sigma_g/\mathrm{d}t$ (necessary to increment the fields along the random walk) thus relies on an estimate of $R_g[\sigma, \sigma']$ which is of the same kind as expression (15) for the observable O: rather than inserting the observable O, one inserts the operator A_g (conjugate to the field σ_g) at a specific point in the ordered product of propagators. The coupling in the time-evolution of the fields is obvious from Eq. (19). In particular, each pair of fields (σ_g, τ_g) is strongly coupled through the first term in the right-hand side of Eq. (19). Indeed, combining Eq. (19) and its counterpart for τ_g, it is easy to see that the time-evolution of the field σ_g (or, equivalently, τ_g) is governed by a second-order differential equation of the type $\mathrm{d}^2\sigma_g/\mathrm{d}t^2 \simeq -\alpha_g^2\sigma_g/4$, supplemented with a drift term (R_g) and a noise term (η_g) in both the field σ_g and its velocity $\mathrm{d}\sigma_g/\mathrm{d}t$.

The detail of the Monte Carlo algorithm for implementing the complex Langevin simulation will be reported elsewhere. In short, the Langevin algorithm proceeds essentially in two alternating steps: (i) for the current value of the fields, calculate O and store it; (ii) update the fields by calculating all their time-derivatives $\mathrm{d}\sigma_g/\mathrm{d}t$, using expression (20) for estimating the R_g's. The time-average of O then yields the output of the quantum computation, the statistics being controlled by adjusting the length of the random walk.

5. Example: The quantum FFT circuit

Let us now discuss the scaling of the computational effort required to simulate a quantum algorithm, focusing on the quantum FFT algorithm [14] used in Shor's factoring algorithm as an illustration.[5]

[5]Note that the FFT is not the most computationally demanding task in Shor's algorithm, but this is unimportant for our illustrative purpose here.

Since Shor's algorithm has been described in detail in the literature (see, e.g., [3] for a review), it will be sufficient to note that after a certain number of computational steps, the quantum register is in a *periodic* superposition of states labeled by an integer between 0 and 2^L-1, the period being related to the sought factor of the composite number. The register is then subjected to a quantum FFT, resulting in a probabilistic estimate of the period (the probability of success can be made arbitrarily close to one by repeating the computation). The time-demanding task in the Monte Carlo simulation of the quantum FFT is the update of the $4G$ auxiliary fields. Performing one step of the random walk in the auxiliary-field space needs the computation of $4G$ time-derivatives, each requiring the calculation of a single (cf. Eqs. (21) and (22)) one-bit matrix element (involving a product of about G/L non-unit 2×2 matrices). Thus, since G scales as $L^2/2$ for the quantum FFT circuit, of the order of L^3 computation steps (2×2 matrix multiplication) are necessary to perform one step of the random walk. Assuming that the number of steps necessary to achieve a given statistical error in the estimate of $\langle O\rangle$ does not grow exponentially with G (the sign problem should be overcome and the auto-correlation time of the random walk should not be exponential in G), the total number of computation steps would be polynomial in L. This does not rule out the possibility that, for a general quantum algorithm, the simulation effort might be *polynomial* in L whenever the number of gates G required in the quantum circuit is polynomial in L. This is an open question.

As an example, we consider here a two-bit quantum FFT, i.e., the quantum computation of the discrete Fourier transform of a 4-point function (see Fig. 2). The input qubits of the quantum register ($L=2$) are labeled 0 (and 1) for the least (and most) significant qubit. The two-bit quantum FFT circuit [14] requires a single two-bit gate, a controlled-phase operator $C_{01}=\mathrm{e}^{\mathrm{i}\omega AB}$ acting on qubits a and b, with $\omega=\pi/2$ and $A=B=(1-\sigma_z)/2$, and two additional one-bit gates H_0 and H_1 with H being the Hadamard transformation,

$$H|0\rangle \to (|0\rangle+|1\rangle)/\sqrt{2}, \qquad H|1\rangle \to (|0\rangle-|1\rangle)/\sqrt{2} \tag{23}$$

The total unitary transformation is the ordered product $U=H_0C_{01}H_1$. The two one-bit gates H_0 and H_1 can be incorporated into the two-bit gate, which can in turn be written in terms of field-dependent one-bit gates using the Hubbard–Stratonovich representation, yielding $U[\sigma]=U_0(\sigma)U_1(\tau)$, with

$$U_0(\sigma)=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & \mathrm{e}^{\mathrm{i}\omega\sigma}\\ 1 & -\mathrm{e}^{\mathrm{i}\omega\sigma}\end{pmatrix}, \qquad U_1(\tau)=\frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ \mathrm{e}^{\mathrm{i}\omega\tau} & -\mathrm{e}^{\mathrm{i}\omega\tau}\end{pmatrix} \tag{24}$$

For a simple test of the Langevin algorithm, we consider here the Fourier transform of a constant function, i.e., the initial state is the product state $2^{-1/2}(|0\rangle+|1\rangle)\otimes 2^{-1/2}(|0\rangle+|1\rangle)$. The complex

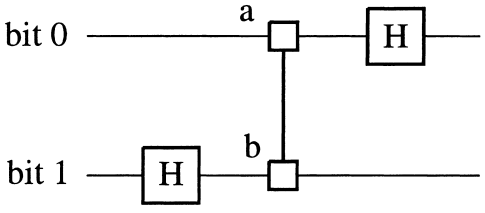

Fig. 2. Two-bit quantum fast Fourier transform circuit. It requires two one-bit Hadamard gates and one two-bit controlled-phase gate.

action can then be simply expressed as

$$S = \omega(\sigma\tau - \sigma'\tau') + \mathrm{i}\ln\left(\frac{1+\mathrm{e}^{\mathrm{i}\omega(\sigma-\sigma')}}{2}\right) \tag{25}$$

depending on the four auxiliary fields σ, τ, σ', and τ'. A straightforward calculation shows that the stochastic differential equations obeyed by the fields are

$$\frac{\mathrm{d}\sigma}{\mathrm{d}t} = -\mathrm{i}\frac{\omega}{2}\left(\tau - \frac{1}{2}\right) - \frac{\omega}{4}\tan\left(\frac{\omega}{2}(\sigma-\sigma')\right) + \eta_\sigma \tag{26}$$

$$\frac{\mathrm{d}\sigma'}{\mathrm{d}t} = \mathrm{i}\frac{\omega}{2}\left(\tau' - \frac{1}{2}\right) + \frac{\omega}{4}\tan\left(\frac{\omega}{2}(\sigma-\sigma')\right) + \eta_{\sigma'}$$

$$\frac{\mathrm{d}\tau}{\mathrm{d}t} = -\mathrm{i}\frac{\omega}{2}\sigma + \eta_\tau, \qquad \frac{\mathrm{d}\tau'}{\mathrm{d}t} = \mathrm{i}\frac{\omega}{2}\sigma' + \eta_{\tau'}$$

The Monte Carlo simulation of these equations is easy to perform. (Note that the fixed point of the Langevin dynamics, i.e., the path of minimum action, is $\sigma = \sigma' = 0$, $\tau = \tau' = 1/2$.) The resulting Monte Carlo averages for the one-bit observables $O_0 = |0\rangle\langle 0|$ and $O_1 = |0\rangle\langle 0|$ converge to 1, implying that the expectation value for the output register is $|00\rangle$ as expected (the spectrum has a continuous component only). The simulation of larger quantum circuits using this technique is the subject of further work to be reported elsewhere.

6. Conclusion

We have shown that a quantum computer can be treated as a genuine quantum many-particle system, and that such an approach sheds new light on quantum computation. More specifically, the use of a quantum Monte Carlo method might be interesting for sinulating ''large'' quantum computers because of the polynomial scaling of the auxiliary-field space in the dimension of the problem. This advantage, however, hinges on an appropriate circumvention of the Monte Carlo ''sign'' problem. In this respect, the use of a Langevin algorithm as a possibly efficient simulation technique is discussed. The stochastic simulation of quantum computation proposed here could be useful for at least two reasons: (i) it could help in devising actual quantum computers by avoiding the need for an explicit experimental realization to test a quantum algorithm; (ii) it could give rise to a new class of ''quantum-inspired'' algorithms that could be implemented on an ordinary classical computer for solving certain computationally hard problems.

Acknowledgements

We acknowledge C. Adami for many helpful discussions. This work has been funded in part by the NSF under grant nos. PHY 94-12818 and PHY 94-20470, and by a grant from DARPA/ARO through the QUIC Program (no. DAAH04-96-1-3086).

References

[1] S. Lloyd, Science 261 (1993) 1569; Science 273 (1996) 1073.
[2] D.P. DiVincenzo, Science 270 (1995) 255.
[3] A. Ekert, R. Jozsa, Rev. Mod. Phys. 68 (1996) 733.
[4] P.W. Shor, Proceedings of the 35th Annual Symposium on Foundations of Computer Science, IEEE Press, New York, 1994.
[5] S.E. Koonin, JASON Report (JSR-95-115).
[6] A. Barenco, C.H. Bennett, R. Cleve, D.P. DiVincenzo, N. Margolus, P. Shor, T. Sleator, J.A. Smolin, H. Weinfurter, Phys. Rev. A 52 (1995) 3457.
[7] J.W. Negele, H. Orland, Quantum Many-particle Systems, Addison-Wesley, Reading, MA, 1988.
[8] J. Hubbard, Phys. Lett. 3 (1959) 77; R.D. Stratonovich, Dokl. Akad. Nauk. SSSR 115 (1957) 1907 [Sov. Phys. Dokl. 2 (1958) 416].
[9] Y. Alhassid, S.E. Koonin, Phy. Rev. C 23 (1981) 1590; Y. Alhassid, B. Muller, S.E. Koonin, Phys. Rev. C 23 (1981) 487.
[10] C.W. Johnson, S.E. Koonin, G.H. Lang, W.E. Ormand, Phys. Rev. Lett. 69 (1992) 3157; S.E. Koonin, D.J. Dean, K. Langanke, Phys. Rep. 278 (1997) 2.
[11] D.P. DiVincenzo, Phys. Rev. A 51 (1995) 1015.
[12] D. Deutsch, A. Barenco, A. Ekert, Proc. R. Soc. London, Ser. A 449 (1995) 669.
[13] S. Lloyd, Phys. Rev. Lett. 75 (1995) 346.
[14] D. Coppersmith, IBM Res. Rep. RC19642, 1994; D. Deutsch, unpublished.
[15] L.K. Grover, Proceedings of the 28th Annual ACM Symposium on Theory of Computing, 1996.
[16] N.A. Gershenfeld, I.L. Chuang, Science 275 (1997) 350.
[17] N. Metropolis, A.W. Rosenbluth, M.N. Rosenbluth, A.H. Teller, E. Teller, J. Chem. Phys. 21 (1953) 1087.
[18] G. Parisi, Y.S. Wu, Sci. Sin. 24 (1981) 483; G. Parisi, Phys. Lett. B 131 (1983) 393.
[19] J.R. Klauder, Phys. Rev. A 29 (1984) 2036.
[20] K. Okano, L. Schulke, B. Zheng, Prog. Theor. Phys. Suppl. 111 (1993) 313.
[21] C. Adami, S.E. Koonin, unpublished.