A quantum Monte Carlo calculation of the ground state energy of the hydrogen molecule

Carol A. Traynor and James B. Anderson
Department of Chemistry, The Pennsylvania State University, University Park, Pennsylvania 16802
Bruce M. Boghosian
Thinking Machines Corporation, Cambridge, Massachusetts 02142-1264

(Received 20 August 1990; accepted 6 November 1990)

We have calculated the ground state energy of the hydrogen molecule using the quantum Monte Carlo (QMC) method of solving the Schrödinger equation, without the use of the Born–Oppenheimer or any other adiabatic approximations. The wave function sampling was carried out in the full 12-dimensional configuration space of the four particles (two electrons and two protons). Two different methods were employed: the diffusion quantum Monte Carlo (DQMC) method and the Green's function quantum Monte Carlo (GFQMC) method. This computation is very demanding because the configurations must be evolved on the time scale of the electronic motion, whereas the finite nuclear mass effects are resolved accurately only after equilibration on the much slower time scale of the nuclear motion. Thus, a very large number of iterations is required. The calculations were performed on the CM-2 Connection Machine computer, a massively parallel supercomputer. The enormous speedup afforded by the massive parallelism allowed us to complete the computation in a reasonable amount of time. The total energy from the DQMC calculations is $-1.163\,97 \pm 0.000\,05$ a.u. A more accurate result was obtained from the GFQMC calculations of $-1.164\,024 \pm 0.000\,009$ a.u. Expressed as a dissociation energy, the GFQMC result is $36\,117.9 \pm 2.0\ \mathrm{cm}^{-1}$, including the corrections for relativistic and radiative effects. This result is in close agreement with accurate nonadiabatic-relativistic dissociation energies from variational calculations (corrected for radiative effects) in the range of 36 117.9–36 118.1 cm^{-1} and with the best experimentally determined dissociation energy of McCormack and Eyler $36\,118.1 \pm 0.2\ \mathrm{cm}^{-1}$.

I. INTRODUCTION

The determination of energies for molecular systems is a problem of general interest in chemistry and physics. We report here calculations of the ground state energy for the relatively simple system of the hydrogen molecule using quantum Monte Carlo (QMC) methods without the fixed-nuclei restriction, the Born–Oppenheimer, or any other adiabatic approximations. We have chosen this case since it is the simplest molecular system for which there is an extensive history of accurate theoretical predictions and high quality experimental measurements of the dissociation energy. These QMC calculations also provide us with an example of a computationally demanding problem which is well suited for massively parallel computing.

The history of accurate calculations of energies for H_2 begins with the 1933 work of James and Coolidge.[1] Their work represented one of the first successes in solving the Schrödinger equation for molecules. In the 1960's, more accurate results for the hydrogen molecule were obtained by Kolos and Roothaan[2] and by Kolos and Wolniewicz,[3] who established the foundation for future calculations. They implemented a variational approach in which the wave function is expressed in elliptic coordinates, and using a method of Born,[4] the Hamiltonian is separated into two parts $H = H^0 + H'$, where H^0 is the electronic Hamiltonian including nuclear repulsion, and H' is the Hamiltonian for the nuclear motion including coupling between the electrons and the nuclei. The adiabatic approximation is made by neglecting the off-diagonal contributions of H'. Variational calculations by Kolos and Wolniewicz, using a 54-term wave function, yielded a dissociation energy in the adiabatic approximation of 36 118.1 cm^{-1}.[3] Incorporation of the corrections for relativistic[3] and radiative effects[5] using perturbation theory (-0.54 and $-0.22\ \mathrm{cm}^{-1}$, respectively) resulted in a dissociation energy for the hydrogen molecule of 36 117.4 cm^{-1}. Their calculations have been outlined in detail by Fischer.[6] Improvements by Wolniewicz,[7] including a more flexible wave function, a variational-perturbation method to include the off-diagonal contributions to the exact nonrelativistic Hamiltonian, and the relativistic and radiative corrections gave a dissociation energy of 36 118.01 cm^{-1}. Kolos *et al.*[8] made improvements to the Born–Oppenheimer potential curve and incorporated the same corrections for nonadiabatic, relativistic, and radiative effects as Wolniewicz to obtain an energy of 36 118.088 cm^{-1}. In addition, Bishop and Cheung[9] calculated the energy of H_2 by treating the full four-body problem as a nonadiabatic variational problem using a 1070-term wave function to arrive at a dissociation energy for H_2 of 36 117.92 cm^{-1}.

In 1960, experimental measurements by Herzberg and Monfils[10] using far-UV spectroscopy gave a dissociation energy for H_2 of 36 113.6 cm^{-1}, which was in agreement with the theoretical predictions at that time. By 1968, however, that experimental energy no longer matched the best theoretical predictions when Kolos and Wolniewicz[3] reported

an energy of 36 117.4 cm^{-1}, revealing a discrepancy of approximately 4 cm^{-1}. Subsequent reinterpretation of the spectra showed that the original evaluation of the dissociation limit had been influenced by the presence of overlapping lines. A more careful analysis of the spectra led to an improved estimate of the dissociation energy with an upper bound of 36 118.3 cm^{-1} and a lower bound of 36 116.3 cm^{-1}.[11] Stwalley[12] also analyzed the high resolution spectra available from the work of Herzberg *et al.*[10,11] and of Namioka.[13] Stwalley examined the effect of long-range forces on the vibrational levels near the dissociation limit, made an assignment of unassigned lines, and concluded that the dissociation energy of H_2 was 36 118.6 $\pm$ 0.5 cm^{-1}. The most recent experimental determination of the dissociation energy is that of McCormack and Eyler,[14] which gives a value of 36 118.1 $\pm$ 0.2 cm^{-1}. In addition, McCormack *et al.*[15] as well as Jungen *et al.*[16] measured transitions of high Rydberg states and have obtained an ionization potential for H_2 (from which the dissociation energy can be extracted) with an uncertainty of $\pm$ 0.015 cm^{-1}.

The idea of simulating the Schrödinger equation using Monte Carlo methods originated with the work of Fermi. In describing that work, Metropolis and Ulam[17] noted that the Schrödinger equation could be expressed as a diffusion equation and simulated by a system of particles undergoing a random walk in which there is a probability for multiplication of particles (hereafter, referred to as psips). With the subsequent advances in computer technology, Monte Carlo methods have become more practical for calculating properties of atomic and molecular systems. The random walk method has been applied to polyatomic ions[18] and molecules[19] using the importance sampling technique of Grimm and Storer.[20] Importance sampling has also been applied to the Green's function quantum Monte Carlo (GFQMC) method used by Kalos *et al.*[21,22]

Whereas the diffusion quantum Monte Carlo (DQMC) method simulates the time-dependent Schrödinger equation, the GFQMC method simulates the time-independent Schrödinger equation. It thus eliminates the problem of finite time step error, but replaces it by a cutoff of the repulsive potential at small distances necessary for the stability of the algorithm. Just as the DQMC method converges to the exact answer only in the limit of small time step, the GFQMC method as used here converges to the exact answer only in the limit of small cutoff; we have used a small cutoff in this work. The DQMC method has been used for treating several excitonic systems[23,24] involving coupled nuclear and electronic motion without the Born–Oppenheimer approximation. Ceperley and Alder[25] have used GFQMC methods for coupled systems of protons and electrons to study bulk hydrogen at high pressures.

II. THEORY

A. The diffusion QMC method

The DQMC method uses the time-dependent Schrödinger equation

$$i\hbar\frac{\partial\Psi}{\partial t}=-\sum_{j}^{n}\frac{\hbar^2}{2m_j}\nabla_j^2\Psi+V(\mathbf{R})\Psi\equiv\hat{H}\Psi \tag{1}$$

in $3n$-dimensional space, where m_j is the mass of particle j and $\mathbf{R}$ represents the configuration space of the n particles. The potential energy V is expressed as

$$V(\mathbf{R})=\sum_{i>j}^{n}\frac{Z_iZ_je^2}{r_{ij}}. \tag{2}$$

The Schrödinger equation can then be represented in imaginary time

$$\hbar\frac{\partial\Psi}{\partial\tau}=\sum_{j}^{n}\frac{\hbar^2}{2m_j}\nabla_j^2\Psi-[V(\mathbf{R})-E_T]\Psi, \tag{3}$$

where $\tau\equiv it$ and we have introduced a constant energy offset E_T. This energy offset is introduced for convenience to alter the zero of energy without affecting the properties calculated from the solution to the Schrödinger equation. Equation (3) has real solutions of the form

$$\Psi(\mathbf{R},\tau)=\sum_{\alpha}c_\alpha\phi_\alpha(\mathbf{R})e^{-(E_\alpha-E_T)\tau/\hbar}. \tag{4}$$

So for positive real τ, the decaying exponential causes the states with the larger eigenvalues to decay away, leaving only the state with the smallest eigenvalue, after long τ.

Importance sampling is a technique used to improve the statistical accuracy of the simulation. To implement importance sampling[20,21] the exact wave function Ψ is multiplied by a trial wave function ϕ_T to obtain a new function f:

$$f(\mathbf{R},\tau)=\phi_T(\mathbf{R})\Psi(\mathbf{R},\tau). \tag{5}$$

Substituting f/ϕ_T for Ψ in Eq. (3), we obtain the following Fokker–Planck equation for $f(\mathbf{R},t)$:

$$\hbar\frac{\partial f}{\partial\tau}=\sum_{j}^{n}\frac{\hbar^2}{2m_j}\nabla_j\cdot(\nabla_j f-f\nabla_j\ln|\phi_T|^2)-\left(\frac{\hat{H}\phi_T}{\phi_T}-E_T\right)f. \tag{6}$$

The terms on the right-hand side may be identified as a diffusion term, an advection term, and a source/sink term, respectively.

The important result of adding an advection term is to force the psips away from regions where ϕ_T is small. This advective effect was given a nice intuitive interpretation by Reynolds *et al.*[26] Suppose that ϕ_T is the canonical ensemble probability distribution corresponding to an "effective potential energy" v; i.e., write $\phi_T\propto\exp(-\beta v)$. Then the advection term in Eq. (6) is proportional to the gradient of v. This is an effective force that pushes the particles toward regions of higher importance (higher ϕ_T). The source/sink term contains the growth rate $(\hat{H}\phi_T/\phi_T-E_T)$ which is referred to as the local energy. The growth rate can thus be controlled for a good choice of ϕ_T by adjusting the value of E_T.

As usual, the Monte Carlo simulation of the Fokker–Planck equation [Eq. (6)] is carried out by representing the density f by particles that take random steps to simulate the diffusion, take directed steps to simulate the advection, and are multiplied or eliminated to model sources and sinks, respectively. Since the particles take finite time steps in a computer simulation, their evolution may be described by an integral transport equation

$$f(\mathbf{x},\tau+\Delta\tau)=\int d^{N}s f(\mathbf{x}-\mathbf{s},\tau)p(\mathbf{s})(1+S\Delta\tau), \tag{7}$$

where p is the probability of taking step $\mathbf{s}$ in time $\Delta\tau$ and S is the growth rate (positive for multiplication of particles and negative for elimination of particles); these may depend weakly on $\mathbf{x}$. Given that the moments of p are

$$1 = \int d^N s p(\mathbf{s}), \tag{8}$$

$$\mathbf{c} \equiv \frac{1}{\Delta\tau} \int d^N s p(\mathbf{s}) \mathbf{s}, \tag{9}$$

$$\mathbf{D} \equiv \frac{1}{2\Delta\tau} \int d^N s p(\mathbf{s}) \mathbf{s}\mathbf{s} \tag{10}$$

and by performing a Taylor expansion for $\mathbf{s}$, we recover the desired Fokker–Planck equation

$$\frac{\partial f}{\partial \tau} = \nabla \cdot (\mathbf{D} \cdot \nabla f - \mathbf{c} f) + S f. \tag{11}$$

Comparing Eqs. (6) and (11), we identify the diffusivity

$$D_{ij} = \frac{m_e}{2m_i} \delta_{ij}, \tag{12}$$

where m_i is the mass associated with the ith degree of freedom and there is no sum on i, the advection velocity

$$c_i = \frac{m_e}{2m_i} \nabla_i \ln|\phi_T|^2, \tag{13}$$

and the growth rate

$$S = E_T - \frac{\hat{H}\phi_T}{\phi_T}. \tag{14}$$

B. The Green's function QMC method

The GFQMC method is well described in Ref. 22 for the case of particles of equal masses. In this section, we give the straightforward generalization to the situation of particles of different masses, as is appropriate for our problem.

We begin with the time-independent Schrödinger equation which may be expressed in the form

$$-\sum_j^n \frac{\hbar^2}{2m_j E} \nabla_j^2 \Psi(\mathbf{R}) + \frac{V(\mathbf{R})}{E} \Psi(\mathbf{R}) = \Psi(\mathbf{R}). \tag{15}$$

Defining $k^2 \equiv -2m_e E/\hbar^2$ and scaled coordinates $\mathbf{Q}_j \equiv \sqrt{m_j/m_e}\,\mathbf{R}_j$, Eq. (15) becomes

$$\left(\frac{1}{k^2} \nabla_Q^2 - 1\right)\Psi = -\frac{\Upsilon(\mathbf{Q})}{E} \Psi, \tag{16}$$

where the function Υ is defined by $\Upsilon(\mathbf{Q}) = V(\mathbf{R})$.

Now, the left-hand side of Eq. (16) is simply a Helmholtz operator. The Green's function for the N-dimensional Helmholtz equation is defined by

$$\left(\frac{1}{k^2} \nabla_Q^2 - 1\right) G(\mathbf{Q},\mathbf{Q}') = -k^{-N} \delta(\mathbf{Q} - \mathbf{Q}'). \tag{17}$$

The solution of Eq. (17) for G is elementary (see, e.g. Ref. 22). The result is

$$G(\mathbf{Q},\mathbf{Q}') = (2\pi)^{-N/2} (kq)^{1-N/2} K_{N/2-1}(kq), \tag{18}$$

where $q \equiv |\mathbf{Q} - \mathbf{Q}'|$, K is the modified Bessel function that vanishes at infinity, and $N \equiv 3n$, where n is the number of particles. A simple rescaling of the independent variable gives $G(\mathbf{R},\mathbf{R}')$.

Using this Green's function to solve Eq. (16) as though the right-hand side of that equation were simply an inhomogeneous term yields an integral equation for $\Psi(\mathbf{R})$:

$$\Psi(\mathbf{R}) = k^N \int d^N R'\, G(\mathbf{R},\mathbf{R}') \frac{V(\mathbf{R}')}{E} \Psi(\mathbf{R}'). \tag{19}$$

With the inclusion of the importance sampling function ϕ_T, Eq. (19) may be expressed in terms of $\Phi \equiv \phi_T \Psi$. Thus

$$\Phi(\mathbf{R}) = k^N \int d^N R' \frac{\phi_T(\mathbf{R})}{\phi_T(\mathbf{R}')} G(\mathbf{R},\mathbf{R}') \frac{V(\mathbf{R}')}{E} \Phi(\mathbf{R}'). \tag{20}$$

The goal is then to evaluate this integral by Monte Carlo. To see how to sample for this purpose, let us momentarily return to using $\mathbf{Q}$ for our independent variable. It is clear that $\mathbf{X} \equiv k\mathbf{Q}$ must be sampled from the distribution

$$p(\mathbf{X}) d^N X = (2\pi)^{-N/2} X^{1-N/2} K_{N/2-1}(X) d^N X. \tag{21}$$

Writing $d^N X = X^{N-1} d\Omega\, dX$, we see that we must sample the magnitude kQ from

$$r(X) dX = (2\pi)^{-N/2} X^{N/2} K_{N/2-1}(X) dX. \tag{22}$$

This problem is solved in the appendix of Ref. 22. The result is

$$Q = -\frac{1}{k} (1 - \xi_0^{2/(N-1)})^{1/2} \ln(\xi_1 \xi_2 \cdots \xi_N), \tag{23}$$

where the ξ_j's are uniformly distributed random numbers (see Appendix A for details of the random number generator used in this work). Reference 22 also shows clearly how to sample solid angle in $3n \equiv N$ dimensions, so that each of the separate components Q_i are obtained. Finally, we can simply rescale to get $\mathbf{R}$,

$$R_i = \sqrt{\frac{m_e}{m_i}} Q_i. \tag{24}$$

This is the sampling procedure referred to in Sec. III of this paper.

C. The trial wave function

The trial wave function ϕ_T used for the implementation of the importance sampling scheme was a product of four terms

$$\phi_T = \phi_1 \phi_2 \phi_3 \phi_4. \tag{25}$$

The four terms are

$$\phi_1 = \exp(-a r_{13}) + \exp(-a r_{14}), \tag{26}$$

$$\phi_2 = \exp(-a r_{23}) + \exp(-a r_{24}), \tag{27}$$

$$\phi_3 = \exp\left(\frac{b r_{12}}{1 + b r_{12}}\right), \tag{28}$$

$$\phi_4 = \exp[-d(r_{34} - c)^2], \tag{29}$$

where the labels 1 and 2 refer to the electrons and 3 and 4 refer to the protons. The trial function contains a Jastrow function[27] of the interelectronic distance (ϕ_3) and a har-

monic oscillator term (ϕ_4) to include the nuclear interaction. For the parameters a, b, c, and d, we used (in a.u.) 1.1750, 0.500, 1.401, and 10.0, respectively.

III. THE ALGORITHM

A. The DQMC code

The general algorithm involved in the DQMC calculation involves a number of substeps within each time step executed. Figure 1 is a flowchart of the computations necessary for one time step. The algorithm begins with the initialization of the psips in the 12-dimensional configuration space with weights initialized to 1.0. After initialization, the particles undergo advection and diffusion according to Eq. (6). The predictor–corrector method used in previous calculations by Anderson[28] is implemented to ameliorate the error due to finite time step. The weights of each of the psips are adjusted and used in the calculation of the local energy. Finally, the psips are killed or multipied according to probabilities determined by their weights.

B. The GFQMC code

The Green's function QMC algorithm is outlined in Fig. 2. As with the DQMC method, it begins with an initial distribution of psips, with weights. The value of the wave function is calculated and the weights are adjusted by $1/\Psi(R')$. Next, the sampling of the Green's function is performed, as described above, and all psips take a step. The wave function is calculated and the weights are then multiplied by $\Psi(R)$. The killing and multiplication is performed based on these weights, followed by calculation of the local energy. New weights are calculated from the ratio V/E, where V is the potential energy given by Eq. (2) and E is the trial energy minus an energy offset; we used 20.0 a.u. This energy offset

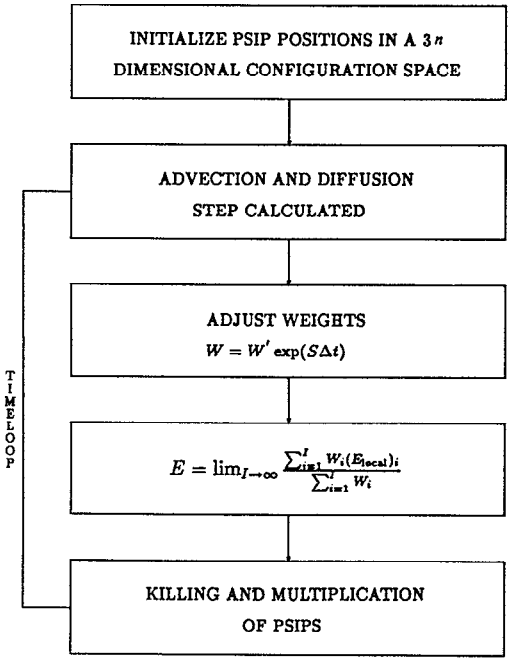

FIG. 1. Flowchart of the diffusion quantum Monte Carlo code.

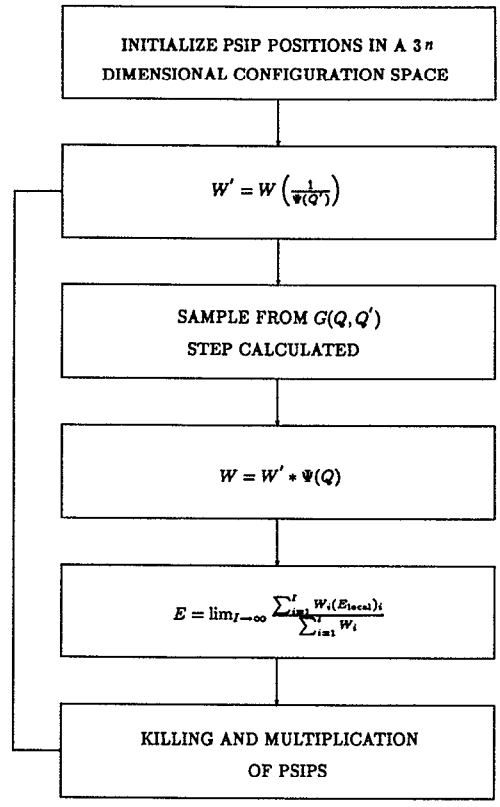

FIG. 2. Flowchart of the Green's function quantum Monte Carlo code

(potential cutoff) of the zero of the potential energy is used to ensure that the weights remain positive. In addition, the energy E can be associated with an effective time step through the diffusion equation which relates time step and the mean-square step size. In our case, this effective time step is 0.05 a.u. As a result of this large time step (relative to that of the DQMC calculation), the GFQMC method allows us to compute the energy with a much higher accuracy for a given amount of computation time. Also, an average potential energy is used for the cases in which the interelectronic distance r_{ij} is small, as described by Anderson.[29] Thus, the interelectronic potential is given by

$$V_{ij} = \begin{cases} \dfrac{1}{r_{ij}}, & \text{for } r_{ij} \geqslant r^* \\ \dfrac{3}{2r^*}, & \text{for } r_{ij} < r^*, \end{cases} \tag{30}$$

where we used $r^* = 0.090$ a.u.

C. Implementation on the Connection Machine

1. General architectural description

The programs were developed and implemented on the CM-2 Connection Machine® computer (Connection Machine is a registered trademark of Thinking Machines Corp.), a massively parallel computer with up to $2^{16} = 65\,536$ processors. Each processor has up to $2^{18} = 262\,144$ bits of local memory, so the machine has an overall memory capacity of 2 gigabytes. Thanks to floating point accelerator chips that support each group of 32 proces-

sors in the CM-2, the peak performance figure for the full machine is roughly 2.8×10^{10} floating-point operations per second (28 gigaflops). Our code achieved roughly 1.5 gigaflops on a full CM-2 using 64 bit IEEE format floating point representation.

The CM-2 operates primarily in Single Instruction Multiple Data (SIMD) mode; i.e., an identical instruction set is broadcast to each processor. Code is edited, compiled, and executed on a front-end computer (in our case a Sun 4 work station). When instructions that involve the manipulation of parallel data are executed, the appropriate commands are sent to the CM-2 processor array.

In order to make the granularity of the machine somewhat flexible, the Connection Machine system allows the use of *virtual processors* (VPs). This is software that causes each physical processor to simulate some larger number of "virtual" processors. It thus effectively makes a CM-2 simulate another CM-2 with more processors.

The CM-2 is furthermore characterized by a very sophisticated communications network linking all of its processors. For a CM-2 with n processors, the topology of the interconnections is an $(-5+\log_2 n)$-dimensional Boolean hypercube linking the floating-point units. This connectivity gives rise to certain primitive communications operations that are very efficient on the machine. These include nearest-neighbor communication on a Cartesian grid and a family of cumulative combining operations along grid dimensions, known as *scans*. These operations warrant a bit of explanation since they are indispensible to the algorithm that we use to kill and split particles in both the DQMC and GFQMC codes.

Scan operations may be thought of as a cumulative combination of data along one dimension of a Cartesian grid. Thus, given a parallel variable whose value in processor j is a_j, a $\oplus$ scan yields a new parallel variable whose value in processor j is $a_1 \oplus a_2 \oplus \cdots \oplus a_j$.

Here $\oplus$ can be any associative binary operation such as max, min, $+$, etc. Note that $a \oplus b$ could be $a+b$ (sum scan), $\max(a,b)$ (max scan), $\min(a,b)$ (min scan), a (copy scan), etc. On a CM-2 with n processors arranged in a hypercube architecture, scans can be accomplished in a time proportional to $\log_2 n$ [$O(n)$ on a serial machine].

To give some simple examples of functionality, consider a hypothetical eight-processor machine with a parallel variable denoted by **A**. The following chart shows the values of **A** and its $\oplus$ scan, denoted by Sc(**A**), in each processor P:

P	0	1	2	3	4	5	6	7
A	a	b	c	d	e	f	g	h
Sc(**A**)	a	$a\oplus b$	$a\oplus b\oplus c$	$a\oplus b\oplus c\oplus d$	$a\oplus b\oplus c\oplus d\oplus e$	$a\oplus b\oplus c\oplus d\oplus e\oplus f$	$a\oplus b\oplus c\oplus d\oplus e\oplus f\oplus g$	$a\oplus b\oplus c\oplus d\oplus e\oplus f\oplus g\oplus h$

An important class of scans, known as segmented scans, allow the user to specify segments along the axis to be scanned such that no data can cross segment boundaries. The boundaries may be specified by a one-bit parallel variable called start bit. To pursue the above example, we might have the following:

P	0	1	2	3	4	5	6	7
A	a	b	c	d	e	f	g	h
Start bit	1	0	0	0	0	1	0	0
Sc(**A**)	a	$a\oplus b$	$a\oplus b\oplus c$	$a\oplus b\oplus c\oplus d$	$a\oplus b\oplus c\oplus d\oplus e$	f	$f\oplus g$	$f\oplus g\oplus h$

In fact, it is easy to see that a segmented scan can be implemented as an ordinary scan on a data structure that includes the start bit by defining a new combining operation for that data structure.[30] For example, a segmented $+$ scan on data x with start bit b can be implemented as an ordinary (unsegmented) $\oplus$ scan on the data structure (x,b), where the operation $\oplus$ is defined by

$$(x_1,b_1)\oplus(x_2,b_2)\equiv[x_2+(1-b_2)x_1,b_1+b_2-b_1b_2]. \tag{31}$$

Thus, though a segmented scan may seem like a generalization of the concept of a scan, it is in fact a special case of the unsegmented scans described above.

More general communications patterns also employ the CM-2's hypercube wires, but in a manner that is controlled by special purpose adaptive routing hardware known as the *router*. Any processor can send data to any other processor by handing the data and the address of the destination processor to the router. In fact, the router keeps track only of the *relative* address of the destination processor with respect to the current cube address. As messages move through the router towards the destination processor, that relative address is continually updated. When it reaches zero, the message is at its destination cube position and is delivered to the processor there.

When more than one message in the router wants to

move along the same edge of the hypercube, one is sent ahead and the others are temporarily buffered. This is done on a first-in–first-out basis. It is thus clear that if fewer messages are sent, fewer such conflicts will arise and the time taken for delivery of the messages will be shorter; router operations with a "low density" of messages are thus to be preferred.

Router communication is supported in high-level languages for the CM-2; e.g., as vector-valued subscripts in CM FORTRAN (the data to be sent is the array and the address to which it is to be sent is determined by the local value of the subscript). Various options allow the user to set a flag in processors that have received data, or to specify what to do in case of a "destination collision" (i.e., two data being sent to the same location within the same processor simultaneously). Possible ways of dealing with such collisions are to take various logical combinations of the colliding data, to add the colliding data, or to signal an error.

2. CM-2 algorithm for killing and splitting psips

By associating one psip with each virtual processor of the CM-2, it is clear that we can do all the computation for moving psips in parallel. This includes, e.g., the computation of the advection and diffusion steps in DQMC, the sampling from the Green's function in GFQMC, as well as the actual movements of the psips. The only potential obstacle to parallelization is then the creation and destruction of psips, which calls for the dynamic allocation and deallocation of processors, respectively. In this section, we show how this can be done using two scan operations and one low-density router operation.

The killing and splitting step consists of replacing each particle with M copies of itself, where the expectation value of the integer M is equal to the particle's weight W. To accomplish this, we have each processor compute

$$M = \begin{cases} [W]+1 & \text{with probability } W-[W] \\ [W] & \text{otherwise} \end{cases}. \tag{32}$$

Note that there has still been no interprocessor communication. Each processor has determined how many children it will have M by purely local operations.

The rest of the algorithm is shown in the table below. To proceed, compute the sum scan of M; call it A. Processors with $M \neq 0$ then send all of their psips' position data to processor $A-1$ and the receiving processors are marked with a boolean flag R. Finally, we perform a segmented copy scan of the psip position data downwards with start bit R. The result is that each psip is cloned exactly M times. After this is done, the surviving psips' weights should all be reset to unity. The kill and split was performed at each iteration.

Parallel variable	Processor 0	1	2	3	4	5
Data	x_0	x_1	x_2	x_3	x_4	x_5
M	2	0	1	1	2	0
A	2	2	3	4	6	6
Sent data		x_0	x_2	x_3		x_4
R	0	1	1	1	0	1
Copy-scanned data	x_0	x_0	x_2	x_3	x_4	x_4

D. Determination of energies and standard error

After allowing the initial ensemble to equilibrate, we began recording energies. For the GFQMC calculation, these energies are determined from the weighted average of the local energies for each psip at each step. The total number of local energies combined to give the resulting energy was approximately 6×10^{10}. For the DQMC calculation, the computed energies for each time step size were determined in a similar fashion and an extrapolation to zero time step was carried out to obtain the final ground state energy of the H_2 molecule.

In order to estimate the statistical (or sampling) error in the energy from the GFQMC calculation, we divided the recorded energies into blocks of 4000 steps. Then we determined the average energies for these blocks, the variance in the block energies, and calculated the standard deviation for the overall mean according to the relation

$$\sigma_{\text{mean}} = \frac{\sigma_{\text{block}}}{\sqrt{N_{\text{blocks}}}}. \tag{33}$$

Since Eq. (33) is applicable only if the fluctuations in energy are uncorrelated from block to block, we examined the correlation of these energies. The standard deviation of the energy for a single block is given by

$$\sigma_{\text{block}} = \frac{\sigma_i}{\sqrt{N_{\text{psips}}\,(N_{\text{steps}}/S_{\text{factor}})}}, \tag{34}$$

where σ_i is the standard deviation in the local energies for individual psips, N_{steps} is the number of steps in a block, and S_{factor} is the correlation factor that gives the number of steps required before the blocks, which are serially correlated, become independent. This S_{factor} can be computed "experimentally" because we know σ_i and σ_{block} from statistical analysis of the recorded energies. For the GFQMC calculation, we have σ_i of 0.2 a.u., σ_{block} of 0.000 07 a.u., N_{psips} of 250 000, and N_{steps} of 4000. From Eq. (34), the step factor S_{factor} is approximately 120 psip steps. Note that this S_{factor} accounts for the extra iterations required due to the small serial correlations of local energies that may persist on the time scale of the nuclear motion. If we were interested only in the electronic contribution to the energy, S_{factor} would have been substantially smaller.

For the DQMC production runs, we began with an initially arbitrary position data set and ran for many correlation times (more than 10 000 steps to allow for equilibration) before collecting energies. The GFQMC runs were initialized with the final DQMC position data set and allowed to equilibrate. Subsequent runs in both cases were done consecutively.

In each step of the GFQMC computation, there are about 1600 floating point operations that must be performed per psip. These operations are necessary for sampling from the Bessel function distribution, computing the potential energy for the weight, etc. Thus, if we desire an error of σ_{mean}, 9×10^{-6} in this work, the total number of floating point operations is given by the total number of steps in the calculation times the number of floating point operations per-

formed in one step per psip. Thus, from Eqs. (33) and (34), we must perform a total of

$$\left(1600\,\frac{\text{flop}}{\text{psip step}}\right)\left(\frac{\sigma_i}{\sigma_{\text{mean}}}\right)^2(S_{\text{factor}}\,\text{psip steps}) = 9.5\times10^{13}\ \text{flop}. \quad (35)$$

The code that we used achieved a performance of approximately 1.5×10^9 floating point operations per second on a CM-2 Connection Machine computer with $2^{16} = 65\,536$ processors. Thus, the above computation could be performed in about 20 h on a full CM-2.

In practice, it took longer than that for several reasons: First, we did most of our runs on a 16 384 processor CM-2, thereby obtaining a factor of 4 lower in computation rate. Second, the above estimate of the number of floating point operations required is low because it did not account for the initial thermalization of the distribution. Third, we had to checkpoint our results periodically; this, of course, is good software practice for any computation that takes this much time. Finally, we ran the CM-2 with only about 90% of the virtual processors filled with psips; this was done because the number of psips fluctuated throughout the computation and we wanted to avoid a situation where there would be more psips than virtual processors [since the killing and splitting algorithm described in Sec. III C 2 assumes that the new configuration will fit into that many virtual processors (In most situations on the CM-2, this could be remedied by dynamically allocating a larger number of virtual processors and proceeding with the computation. Because we were filling the machine's physical memory to capacity with psips, however, we did not have room to allocate a larger VP set, and so we took care to keep the number of psips well below the number of virtual processors in the machine.)].

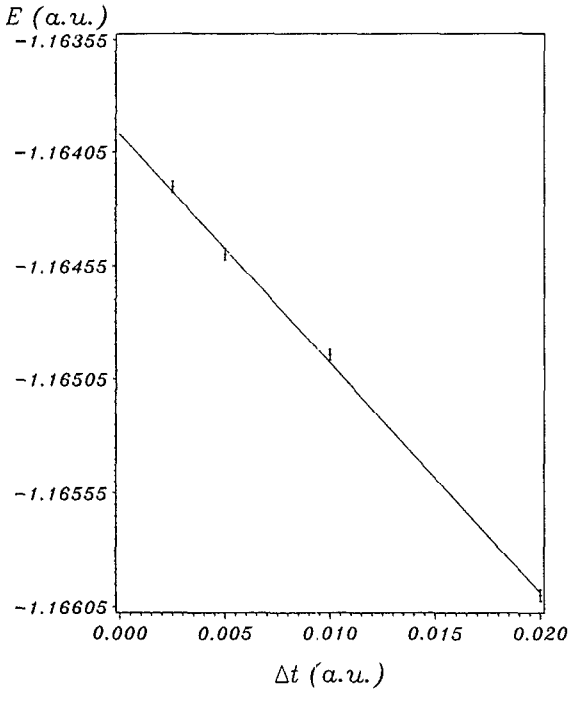

FIG. 3. Energy vs time step.

TABLE I. Dissociation energy of H_2 (D_0).

	Reference	Energy (cm^{-1})
Expt.	Herzberg (Ref. 11)	$36\,116.3 < D_0 < 36\,118.3$
Expt., analysis	Stwalley (Ref. 12)	$36\,118.6 \pm 0.5$
Expt.	McCormack *et al.* (Ref. 15)	$36\,118.1 \pm 0.2$
Nonadiabatic	Wolniewicz (Ref. 7)	36 118.01
Nonadiabatic	Kolos *et al.* (Ref. 8)	36 118.088
Nonadiabatic	Bishop, Cheung (Ref. 9)	36 117.92
DQMC	This work	$36\,107 \pm 11$
GFQMC	This work	$36\,118.6 \pm 2.0$
GFQMC	This work[a]	$36\,117.9 \pm 2.0$

[a]Adding the relativistic and radiative corrections of $-0.54\ cm^{-1}$ (Ref. 4) and $-0.22\ cm^{-1}$ (Ref. 5), respectively.

The ensemble size was limited by the physical memory and by the number of physical processors. As noted above, we used one psip per virtual processor. We were able to fit a maximum of 16 virtual processors into one physical processor. Thus, we used a maximum of 262 144 psips on a 16 384 physical processor CM-2.

IV. RESULTS

The total energy from the diffusion Monte Carlo calculations, after extrapolation to zero time step, is $-1.163\,97 \pm 0.000\,05$ a.u. Expressed as a dissociation energy, the result is $36\,107 \pm 11\ cm^{-1}$. These calculations required approximately 75 h on a 16 384 processor Connection Machine. Figure 3 is a plot of E vs Δt in a.u. and shows the extrapolation to zero time step, with error bars corresponding to one standard deviation. To obtain a factor of 10, lower standard deviations for the energies at each time step would require a factor of 100 in calculational effort at each time step.

We found it more practical to rely on the GFQMC calculations to achieve the desired energy and accuracy. The total energy from the Green's function Monte Carlo calculation, which is free from time step error, is $-1.164\,024 \pm 0.000\,009$ a.u., resulting in a dissociation energy of $36\,118.6 \pm 2.0\ cm^{-1}$, without taking into account relativistic and radiative effects. This required approximately 60 h on a 32 768 processor Connection Machine. With the addition of the relativistic[7] and radiative corrections,[5] the final value for the dissociation energy of H_2 is $36\,117.9 \pm 2.0\ cm^{-1}$. The results are shown in Table I along with the experimental and theoretical values from the literature.

V. CONCLUSIONS

We have calculated the ground state energy of the hydrogen molecule without the use of the Born–Oppenheimer approximation using Monte Carlo techniques. These calculations yield the total energy for the ground state of the hydrogen molecule including nuclear motion, with only the relativistic and radiative corrections to be taken into ac-

count. The results are in agreement with experiment and fall inside the error bars of previous calculations. We were able to take advantage of parallelism inherent in the quantum Monte Carlo procedure in order to solve this problem in a tractable amount of computer time. We are confident that with the advent of larger, more powerful parallel computers, problems like this one will become almost routine.

ACKNOWLEDGMENTS

Support by the National Science Foundation (Grant No. CHE-8714613) and the Petroleum Research Fund administered by the American Chemical Society (Grant No. 18854AC) is gratefully acknowledged. We would like to thank Washington Taylor for many useful discussions during the course of this work and Adam Greenberg for his help with the production runs.

APPENDIX: A RANDOM NUMBER GENERATOR

The random number generator (rng) used in this work is from the CM-2 Scientific Software Library (CMSSL).[31] It uses a Lagged–Fibonacci algorithm[32]

$$X_{n+1} = (X_n + X_{n-k}) \bmod 2^w \tag{A1}$$

to produce uniformly distributed numbers between 0 and $2^w - 1$. These are then used to fill the mantissa bits of a floating point number, normalized between 1.0 and 2.0. Finally, 1.0 is subtracted to yield a random number that is uniformly distributed between 0.0 and 1.0. Thus, since we used a double precision IEEE format floating point representation with 52 bits of mantissa, we took the *width* w to be 52. In all the runs reported here, we took the *table lag* k to be 17.

It is important that the period of the random number generator is much greater than the number of times it is called during the collection of the energies. The period T_p of the generator is discussed in Ref. 32. The result is

$$T_p = (2^k - 1) \cdot 2^w. \tag{A2}$$

Since we used $k = 17$ and $w = 52$, we had a period of approximately $2^{69} \approx 6 \times 10^{20}$. The GFQMC calculation required approximately

$$\left(26 \frac{\text{rng calls}}{\text{psip step}}\right) (6 \times 10^{10} \text{ psip steps}) = 1.5 \times 10^{12} \text{ rng calls} \tag{A3}$$

to the random number generator. Therefore, we can safely conclude that the finite period of the random number generator does not introduce correlation into the calculation of the energy eigenvalue.

[1] H. M. James and A. S. Coolidge, J. Chem. Phys. **1**, 825 (1933).
[2] W. Kolos and C. C. J. Roothaan, Rev. Mod. Phys. **32**, 219 (1960).
[3] W. Kolos and L. Wolniewicz, Rev. Mod. Phys. **35**, 473 (1963); J. Chem. Phys. **43**, 2429 (1965); **49**, 404 (1968).
[4] W. Kolos and L. Wolniewicz, J. Chem. Phys. **41**, 3663 (1964).
[5] J. D. Garcia, Phys. Rev. **147**, 66 (1966).
[6] G. Fischer, *Vibronic Coupling* (Academic, London, 1984).
[7] L. Wolniewicz, J. Chem. Phys. **78**, 6173 (1983).
[8] W. Kolos, K. Szalewicz, and H. Monkhorst, J. Chem. Phys. **84**, 3278 (1986).
[9] D. M. Bishop and L. M. Cheung, Phys. Rev. A **18**, 1846 (1978).
[10] G. Herzberg and A. Monfils, J. Mol. Spectrosc. **5**, 482 (1960).
[11] G. Herzberg, J. Mol. Spectrosc. **33**, 147 (1970).
[12] W.C. Stwalley, Chem. Phys. Lett. **6**, 241 (1970).
[13] T. Namioka, J. Chem. Phys. **43**, 1636 (1965).
[14] E. McCormack and E. E. Eyler, Bull. Am. Phys. Soc. **32**, 1279 (1987).
[15] E. McCormack, J. M. Gilligan, C. Cornaggia, and E. E. Eyler, Phys. Rev. A **39**, 2260 (1989).
[16] Ch. Jungen, I. Dabrowski, G. Herzberg, and M. Vervloet, J. Chem. Phys. **93**, 2289 (1990).
[17] N. Metropolis and S. Ulam, J. Am. Stat. Assoc. **44**, 335 (1949).
[18] J. B. Anderson, J. Chem. Phys. **63**, 1499 (1975); C. A. Traynor and J. B. Anderson, Chem. Phys. Lett. **147**, 389 (1988).
[19] J. B. Anderson, J. Chem. Phys. **65**, 4121 (1976); F. Mentch and J. B. Anderson, *ibid.* **80**, 2675 (1984); D. M. Ceperley and B. J. Alder, *ibid.* **81**, 5833 (1984); R. N. Barnett, P. J. Reynolds, and W. A. Lester, Jr., *ibid.* **82**, 2700 (1985); D. R. Garmer and J. B. Anderson, *ibid.* **86**, 4025 (1987); **86**, 7237 (1987).
[20] R. C. Grimm and R. G. Storer, J. Comp. Phys. **7**, 134 (1971).
[21] M. H. Kalos, D. Levesque, and L. Verlet, Phys. Rev. A **9**, 2178 (1974).
[22] M. H. Kalos, Phys. Rev. **128**, 1791 (1962).
[23] M. A. Lee, P. Vashista, and R. K. Kalia, Phys. Rev. Lett. **129**, 2422 (1983)
[24] V. Mohan and J. B. Anderson, Chem Phys. Lett. **156**, 520 (1989).
[25] D. M. Ceperley and B. J. Alder, Phys. Rev. B **36**, 2092 (1987).
[26] P. J. Reynolds, D. M. Ceperley, B. J. Alder, and W. A. Lester, Jr., J. Chem. Phys. 77, 5593 (1982); D. M. Ceperley and B. J. Alder, Science **231**, 555 (1986).
[27] R. Jastrow, Phys. Rev. **98**, 1479 (1955).
[28] J. B. Anderson, J. Chem. Phys. **82**, 2662 (1985).
[29] J. B. Anderson, J. Chem. Phys. **86**, 2839 (1987).
[30] G. Blelloch, Ph.D. thesis, Massachusetts Institute of Technology, 1989.
[31] CM-2 Connection Machine documentation, version 2.0, © 1990 by Thinking Machines Corp.
[32] D. E. Knuth, *The Art of Computer Programming* (Addison-Wesley, Reading, Mass., 1981), Vol. 2, pp. 26–27.