Theor Chim Acta (1993) 84: 335–342

Theoretica Chimica Acta

Exact three-dimensional time-dependent wave packet calculations on the Connection Machine

David Chasman[1], Robert J. Silbey[1], and Michael D'Mello[2]

[1] Department of Chemistry, Massachusetts Institute of Technology, Cambridge, MA 01239, USA

[2] Thinking Machines Corporation, 245 First Street, Cambridge, MA 02142, USA

Received October 1, 1991/Accepted March 13, 1992

Summary. In this paper, we report our massively parallel implementation of grid techniques for the solution of the time-dependent Schrödinger equation in three spatial dimensions on the Connection Machine, which is a Single Instruction Multiple Data (SIMD) computer. Most of the operations involved in this calculation may be executed independently for each grid point. The few operations which cannot be executed independently are implemented using parallel communication algorithms. In addition, we report a simple modification of the multidimensional FFT, which provides an estimated 15% reduction in computational complexity relative to the standard 2-D FFT. It is suggested that this modification may be very well suited to hypercube communication topologies.

Key words: Connection machine – Single Instruction Multiple Data – Time-dependent Schrödinger equation – Wave packet calculations

1 Introduction

In the past decade, the time-dependent formulation of quantum mechanics (QM) has been used to compute experimental observables for multidimensional problems with tremendous success [1–3]. The results obtained using this approach are completely equivalent to those provided by time-independent approaches. While many molecular vibration/rotation problems have been successfully treated by time-independent techniques, the time-dependent approach offers an alternative computational approach to these problems for the following reason: the most popular time-dependent methods are the so-called "grid techniques". The advantage of these time-dependent grid techniques is that for a basis set of size N, the computational complexity scales as N or $N \log N$ whereas the complexity of time-independent techniques, scales as N^3 in the case of direct diagonalization or N^2 in the case of iterative procedures [4]. The N^3 scaling of the time-independent approach may be prohibitive when a large basis set is necessary [1]. However, the N^2 scaling of iterative techniques may somewhat alleviate this problem. The $N \log N$ scaling of the direct time-dependent approach compares favorably with either of these techniques. The time-dependent approach has been successfully applied to a number of problems, including (a) vibrational dynamics on

dissociative electronic surfaces [5, 6] and (b) high energy dynamics on bound surfaces [7].

Although the use of time-dependent grid techniques avoids the N^3 complexity of matrix solutions, we are forced to face the difficulty of prohibitive running times which arise in the application of these techniques to large three-dimensional problems. Fortunately, wave packet propagation, using the exact grid techniques, is perfectly suited to the architecture of modern parallel computers. In this paper, we present a theoretically optimal implementation of some of the standard wave-packet propagation schemes in 3-dimensions on the CM-2, a SIMD parallel computer.

2 The Connection Machine

The Connection Machine (CM-2) may be viewed as a scalable array of up to 64 K processors which concurrently execute commands issued by a front end computer. Each of these processors can have up to one megabit of memory and executes a normal serial-type instruction set, albeit more slowly than high speed serial computers. For the calculations in this paper we have used a 16 K partition with 256 Kbits of memory per processor.

In addition to the processor instruction set, there is facility to support multidimensional inter-processor communication. There are two types of inter-processor communication – *news* (North East West South) and *send*. *News* communication is optimized for nearest-neighbor communication and the *send* communication is optimized for global communication patterns. What the current generation of CM-2s may lack in processor speed is compensated for by its parallel architecture. For an in-depth discussion of the machine, the reader is referred to [8].

3 Computational procedure

The standard wave-packet propagation techniques can be viewed as two completely separate operations: (a) the evaluation of the hamiltonian (spatial discretization) and (b) the expansion of the propagator (temporal discretization). A common approach is low order differencing in time and space [9, 10]. Another approach is to use function spaces to treat these coordinates. In this paper, we utilize mainly FFTs for the spatial discretization and the Chebeychev expansion method of temporal discretization, which gives stable results as well as a great deal of flexibility in the choice of time step [2]. We also present second order differencing in space and time for the purpose of comparison and completeness.

Here, we briefly review the algorithms used in the various steps of our solution, count the operations required, and examine their suitability for parallelization on the CM-2. An important consideration for the parallelization of any operation is whether it may be executed independently for each point on the grid. Any such operation is intrinsically parallel. Furthermore, the speed of such operations is not in any way limited by speed of inter-processor communication. Operations which cannot be executed independently for each grid point are not intrinsically parallelizeable and a parallel algorithm must be used to obtain optimal speed-up. Also, these will be limited by the speed of inter-processor communication. An algorithm is optimal if the speed-up with N processors is

directly proportional to N. In this discussion, we will assume a grid with $N_g = N_a \times N_a \times N_a$ points, where N_a is the number of points along a given axis, and that there are a total of N_p processors laid out as a $P_a \times P_a$ square grid.

3.1 Evaluation of the hamiltonian

The hamiltonian is given by:

$$\hat{\boldsymbol{H}} = \left[\frac{-\hbar^2}{2m}\nabla^2 + \boldsymbol{V}\right] \tag{1}$$

and its action on the state vector $\psi(\boldsymbol{x}, t)$ is simply the sum of a kinetic energy and a potential energy term. The potential energy term on the grid is evaluated by simple multiplication, that is:

$$\boldsymbol{V}\psi(\boldsymbol{x}, t) = V(\boldsymbol{x}) \cdot \psi(\boldsymbol{x}, t) \tag{2}$$

Since the potential is real valued, this requires a single real times complex ($R \times C$) multiplication at each grid point which may be executed independently for each grid point.

3.2 Evaluation of the kinetic energy operator

The action of the kinetic energy operator is discussed for a second order finite difference scheme and the fourier transform method.

3.2.1 Evaluation of kinetic energy operator using finite difference scheme. The expression for the kinetic energy operator[1] using a second order finite difference scheme is:

$$\psi''(x_i) = \frac{\dfrac{\psi_i - \psi_{i-1}}{l} - \dfrac{\psi_{i+1} - \psi_i}{l}}{l} = \frac{1}{l^2}(2\psi_i - \psi_{i+1} - \psi_{i-1}) \tag{3}$$

where l is the distance between grid points. This requires two $R \times C$ multiplies and two complex plus complex ($C + C$) adds for each grid point. In two dimensions, this operation, which depends on the neighboring grid points, is appropriate to the CM-2's *news* communications. The three-dimensional analog of this formula is very easily implemented using standard circular shift operations.[2]

3.2.2 Evaluation of kinetic energy operator using the fourier method. We recall that:

$$FT(f^{(n)}(x)) = (-i\boldsymbol{k})^n FT(f(x)) \tag{4}$$

where $FT(f(x))$ indicates the fourier transform of $f(x)$. Thus, in order to evaluate the kinetic energy operator, we simply take the fourier transform, multiply by $-\boldsymbol{k}^2$, and take the inverse fourier transform. That is:

$$\nabla^2\psi(\boldsymbol{x}) = FT^{-1}(-\boldsymbol{k}^2 FT(\psi(\boldsymbol{x})) \tag{5}$$

---

[1] This is for the 1-dimensional case – the extension to higher dimensions is clear

[2] Fortran 90 CSHIFT operations

In order to speed this process, a fast fourier transform (FFT) is used. For a review of the FFT and a discussion of the particulars which are relevant to its implementation on the CM-2, we refer the reader to [11–13]. It is important to note that if the vector $\boldsymbol{k}$ in Eq. (4) is arranged in bit-reverse order, the derivative may be computed without bit-reversing the output of the FFT.[3] The communication involved in a bit-reversal is approximately equal to the communication involved in the FFT. By initializing $\boldsymbol{k}$ appropriately, we can avoid the need to undo the bit-reversal of the FFT output. On a parallel computer where the FFT performance is limited by communication speed[4] this will roughly halve the time involved in computing the action of the kinetic energy on the wave-function.

Recently, we have begun to consider possible algorithmic changes which may speed the multidimensional FFT on the CM-2. We have observed that multidimensional FFTs may be written as a sum of smaller multidimensional FFTs instead of the standard sequence of one-dimensional FFTs. We shall refer to the standard multidimensional FFT as the axis-sequential FFT. Our "mixed-axis" FFT has two advantages. First, the computational complexity is decreased by 15% over the axis-sequential FFT. Second, from the point of view of a parallel implementation, the "mixed-axis" approach may simplify the data flow between successive stages of the operation. We present the details of the mixed axis FFT in Appendix A.

3.3 Integration in time

3.3.1 Second order differencing in time. The formula for obtaining the solution of the time-dependent Schrödinger equation (TDSE) using second-order differencing in time is obtained as follows. First, we write the time derivative using the second order difference formula:

$$\frac{\partial \psi}{\partial t} = \frac{\psi(t + \Delta t) - \psi(t - \Delta t)}{2\,\Delta t} \tag{6}$$

By equating $-i\hat{\boldsymbol{H}}\psi/\hbar$ to $\partial\psi/\partial t$ and rearranging this equation, we obtain:

$$\psi(t + \Delta t) \approx \psi(t - \Delta t) - \frac{2i\,\Delta t\,\hat{\boldsymbol{H}}\psi}{\hbar} \tag{7}$$

Time stepping in this scheme requires a $R \times C$ multiply and a $(C + C)$ add for each grid point once $\hat{\boldsymbol{H}}\psi$ has been evaluated. This operation may be executed independently for each grid point.

3.3.2 Expansion of propagator in the Chebeychev polynomials. The integration in time is done by expanding the propagator $e^{\frac{-i\hat{\boldsymbol{H}}\,\Delta t}{\hbar}}$ in the Chebeychev polynomial set. That is:

$$\psi(t + \Delta t) = e^{\frac{-i\hat{\boldsymbol{H}}\,\Delta t}{\hbar}}\psi(t) = \sum_{n=0}^{N_c} a_n \left[\frac{\Delta E\,\Delta t}{2\hbar}\right] \phi_n[-i\hat{\boldsymbol{H}}_{\boldsymbol{norm}}]\psi(t) \tag{8}$$

where:

$$\hat{H}_{norm} = 2\,\frac{\hat{H} - \hat{I}(1/2\,\Delta E + V_{min})}{\Delta E} \tag{9}$$

---

[3] This observation applies to any implementation of the FFT based grid techniques

[4] or, in the case of a serial implementation, the time to do the necessary subscript arithmetic and permute an array in memory

and $\Delta E = E_{max} - E_{min} \cdot E_{max}$ and E_{min} are the maximum and minimum eigenvalues of $\hat{H}$ respectively and V_{min} is the minimum of the potential energy function. The coefficients a_n are given by:

$$a_n[\alpha] = \int_{-i}^{i} dx \frac{e^{i\alpha x}\Phi_n(x)}{[1-x^2]^{1/2}} = 2J_n(\alpha) \tag{10}$$

where $\alpha = (\Delta E\, \Delta t)/(2\hbar)$ and $a_0(\alpha) = J_0(\alpha)$. The Chebeychev polynomials $\phi_n(\hat{X})$ are related by the recurrence relation:

$$\phi_n(\hat{X}) = 2\hat{X}\phi_{n-1}(\hat{X}) + \phi_{n-2}(\hat{X}) \tag{11}$$

which simplifies the application of the series operator to the wave-function. In this case $\hat{X} = -i\hat{H}_{norm}$. Thus, in order to evaluate Eq. (8), we need only retain the value of the two previous Chebeychev polynomials at each grid point. This means that memory demand does not increase as a function of the number of terms in the Chebeychev expansion. This thrifty use of memory is important because each processor in the current generation of massively parallel computers has relatively limited memory. The Chebeychev expansion of the propagator requires $2N_c(R \times C)$ multiplies and $2N_c(C + C)$ adds for each time step Δt, in addition to N_c hamiltonian operations to evaluate the terms of Eq. (8) using Eq. (11). The series in Eq. (8) is convergent because the coefficients a_n decrease exponentially when the argument exceeds the summation index. This makes the Chebeychev expansion precise for an arbitrarily chosen time step [2].

3.4 3-D implementation

In the current implementation, we have considered grids up to size $128 \times 128 \times 128 = 2^{21}$ on a partition of 16 K processors. This amounts to more than one grid point per processor and we have a virtual processor ratio (VP ratio) of 128. A VP ratio of 1 would give us even better performance. In order to lay out the grid on the physical machine, we have placed one dimension of the grid in processor and the other two dimensions are formed over processors as illustrated in Fig. 1. The number of machine cycles necessary to carry out the 3-D propagation is summarized in Table 1.

In Fig. 1, the first two indices of the array A_{ijk} form the processor grid P_{ij} while the index k indicates an "on processor" offset. Although the "k axis" operations are not executed at once for every k, they are simultaneously executed for every ordered pair (i, j) on the processor grid. Further, all "k axis" computations are local to a single processor. This yields additional speedup by allowing us to avoid inter-processor communication. As a result, we observed that performance scales super-linearly in going form 2-D to 3-D. While we defer detailed discussion of performance issues to a subsequent publication, for the purposes of this conference, we note a performance of approximately 450 MFlops on a 16 K partition during the FFT operations. This scales to 1.8 GigaFlops on a full size machine.

4 Summary

We have demonstrated the implementation of grid techniques commonly used in time-dependent approaches to quantum dynamics on the CM-2, a SIMD parallel

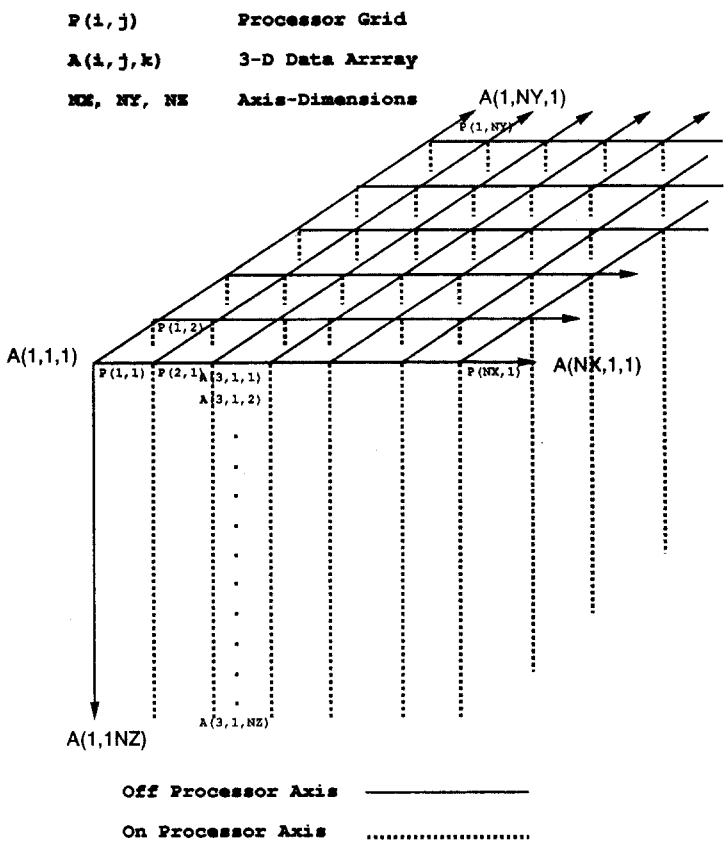

Fig. 1. CM layout for 3-D implementation

Table 1. Computational requirements: 3-D implementation

Estimate of machine cycles used in the various propagation schemes – 3-D			
Hamiltonian ↓	Time→	Second Order Difference	Chebeychev expansion
Fourier transform	Serial	$N_a^3(7+30\log_2 N_a)$	$N_a^3(8N_c+30\log_2 N_a)$
	Parallel	$N_a(7+30\log_2 N_a)$	$N_a(8N_c+30\log_2 N_a)$
Finite difference	Serial	$22N_a^3$	$N_a^3(8N_c+18)$
	Parallel	$22N_a$	$N_a(8N_c+18)$

N_a	Number of grid points per axis
$N_g\ (=N_a\times N_a\times N_a)$	Number of total points on grid
P_a	Number of processors per axis
$N_p\ (=P_a\times P_a)$	Total number of processors
N_c	Number of terms in Chebeychev expansion

computer. We have observed that these techniques are easily and efficiently implemented on this machine. As a caveat, we point out that our present implementation is in three cartesian dimensions with one dimension treated serially. This has direct implications for real molecular systems which would invariably be treated on SIMD architectures using the present layout. The

scaling properties detailed in this paper are therefore applicable in this realm. We feel that the potential offered by this approach will significantly enhance the application of these techniques to many problems of interest in chemical dynamics.

Appendix A: The mixed axis FFT

Here we describe our idea to enhance the performance of multidimensional FFTs in general. These enhancements may be suitable to the architecture of SIMD machines with a hypercube topology. We will assume that the reader is familiar with the standard Cooley–Tukey FFT in one dimension [11]. The traditional approach to the multidimensional FFT is to compute the FFT of the axes sequentially. For an array A_{ij}, we would compute the FFT of the first axis of A_{ij} for all j's simultaneously and then we would compute the FFT of the second axis for all i's simultaneously. The standard 2-D FFT is written as:

$$\hat{z}_{ij} = \sum_{m=1}^{M} \omega_N^{(i-1)(m-1)} \left[\sum_{n=1}^{N} \omega_M^{(j-1)(n-1)} z_{nm} \right] \tag{A.1}$$

If N and M can be factorized in terms of two smaller integers such that $N = PQ$ and $M = RS$ and we substitute $z_{pqrs} = z_{p+P(q-1)r+R(s-1)}$ for z_{ij} we obtain:

$$\hat{z}_{ij} = \sum_{r=1}^{R} \omega_M^{(i-1)(r-1)} \sum_{s=1}^{S} \omega_M^{R(i-1)(s-1)} \sum_{p=1}^{P} \omega_N^{(j-1)(p-1)} \sum_{q=1}^{Q} W_N^{P(j-1)(q-1)} z_{pqrs} \tag{A.2}$$

Following Ref. [11] we note that $\omega_M^R = \omega_S$ and $\omega_N^P = \omega_Q$. If we make this substitution and exchange the sum over s with the sum over p, we have:

$$\hat{z}_{ij} = \sum_{r=1}^{R} \omega_M^{(i-1)(r-1)} \sum_{p=1}^{P} \omega_N^{(j-1)(p-1)} \left[\sum_{s=1}^{S} \omega_S^{(i-1)(s-1)} \sum_{q=1}^{Q} \omega_Q^{(j-1)(q-1)} z_{pqrs} \right] \tag{A.3}$$

Let us refer to the bracketed quantity in Eq. (A.3) as $T_{ij}(p, r)$. We see, following the one-dimensional derivation in [11], that $T_{ij}(p, r)$ is S and Q periodic in the indices i and j respectively. Now, if we assume that $N = M = 2^k$ and $P = R = 2$ and $Q = S = 2^{k-1}$, we have:

$$\hat{z}_{ij} = \sum_{r=1}^{2} \omega_M^{(i-1)(r-1)} \sum_{p=1}^{2} \omega_N^{(j-1)(p-1)} T_{ij}(p, r) \tag{A.4}$$

$T_{ij}(p, r)$ is equal for $\hat{z}_{ij}$, $\hat{z}_{(i+N/2)j}$, $\hat{z}_{i,(j+N/2)}$, and $\hat{z}_{i+N/2,j+N/2}$. In order to compute these four quantities, we need to compute: $T_{ij}(1, 1)$, $T_{ij}(1, 2)$, $T_{ij}(2, 1)$, and $T_{ij}(2, 2)$. The steps involved in this final part of the computation are illustrated in Fig. 2. These two steps require 3 complex multiplies and 8 complex adds in order to compute this stage of the FFT for 4 grid points. This gives us 2 complex adds and an average 3/4 of a complex multiply, or 8.5 operations for each of the N^2 grid points. Since the $T_{ij}(p, r)$ are FFTs of size $N/2 \times N/2$, we may repeat this algorithm recursively k times until we are computing a 1×1 FFT which is equal to the value of a single grid point.

The 2-D FFT of an $N \times N$ grid will require $\log_2 N$ such stages – giving us a total of $8.5N^2 \log_2 N$ operations instead of the $10N^2 \log_2 N$ operations which are required in the axis-sequential FFT. A similar analysis may be applied in 3-D. In addition to this algorithmic improvement, we also wish to point out that the output of each stage of this FFT is the input for the next, which should allow of pipelining of the data between the stages of the FFT.

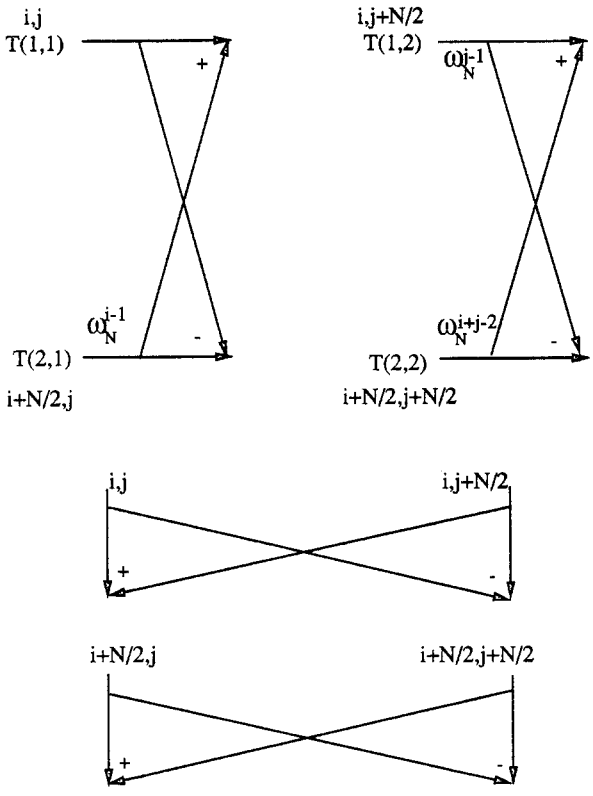

Fig. 2. Schematic diagram of the two parts of a single stage of the mixed-axis 2-D FFT

Acknowledgments. The authors wish to thank the NSF for partial support and Thinking Machines Corporation for computer time and expertise. In particular, we wish to thank Adam Greenberg, J. P. Massar, Robert Krawitz, and Doug MacDonald of Thinking Machines for helpful discussions.

References

1. Kosloff R (1988) J Phys Chem 92(8):2087
2. Gerber RB, Kosloff R, Berman M. (1986) Computer Physics Reports 5(2):59
3. Neuhauser D, Judson RS, Jaffe RL, Baer M, Kouri DJ (1991) Chem Phys Lett 176(6):546 and references therein
4. Thomas D (1979) J Chem Phys 70(6):2979
5. Chasman D, Tannor DJ, Imre D (1988) J Chem Phys 89(11):6667
6. Yun Shi, Tannor DJ (1990) J Chem Phjys 92(4):2517, Yun Shi's Thesis
7. Chasman D, Silbey RJ, Eisenberg M (1990) Chem Phys Lett 175(6):633
8. Hillis WD (1985) The Connection Machine. MIT Press, Cambridge, Massachusetts. ACM Distinguished Dissertation
9. Richardson JL (1991) Comp Phys Comm 63:84
10. DeRaedt H (1987) Comp Phys Rep 7:1
11. Conte SD, de Boor C (1980) Elementary numerical analysis. McGraw-Hill, NY
12. Johnsson SL, Krawitz RL, Frye R, MacDonald D (1989) Cooley–Tukey FFT on the connection machine. Technical Report NA89-4 Thinking Machines Corporation
13. Chasman D, Silbey RJ, Eisenberg M (1991) Theor Chim Acta 79:175