Development of a Parallel Molecular Dynamics Code on SIMD Computers: Algorithm for Use of Pair List Criterion

D. ROCCATANO,[1] R. BIZZARRI,[2,3] G. CHILLEMI,[2] N. SANNA,[2] A. DI NOLA[1]

[1]*Dipartimento di Chimica, Università ''La Sapienza,'' Ple Aldo Moro 5, 00185 Rome, Italy*
[2]*CASPUR Consorzio per le Applicazioni di Supercalcolo per Università e Ricerca, c/o Università ''La Sapienza,'' Rome, Italy*
[3]*Dipartimento di Fisica, Università ''La Sapienza,'' Rome, Italy*

Received 24 April 1996; accepted 24 October 1997

ABSTRACT: In recent years several implementations of molecular dynamics (MD) codes have been reported on multiple instruction multiple data (MIMD) machines. However, very few implementations of MD codes on single instruction multiple data (SIMD) machines have been reported. The difficulty in using pair lists of nonbonded interactions is the major problem with MD codes for SIMD machines, such that, generally, the full connectivity computation has been used. We present an algorithm, the global cut-off algorithm (GCA), which permits the use of pair lists on SIMD machines. GCA is based on a probabilistic approach and requires the cut-off condition to be simultaneously verified on all nodes of the machine. The MD code used was taken from the GROMOS package; only the routines involved in the pair lists and in the computation of nonbonded interactions were rewritten for a parallel architecture. The remaining calculations were performed on the host computer. The algorithm has been tested on Quadrics computers for configurations of 32, 128, and 512 processors and for systems of 4000, 8000, 15,000, and 30,000 particles. Quadrics was developed by Istituto Nazionale di Fisica Nucleare (INFN) and marketed by Alenia Spazio.
 J Comput Chem 19: 685–694, 1998

Keywords: molecular dynamics; domain decomposition algorithm; parallel computers; pair list for molecular dynamics code on SIMD machines; array processor elaborator (APE)

Correspondence to: A. Di Nola
Contract/grant sponsors: Ente per le Nuove Tecnologie, l'Energia e l'Ambiente; Alenia Spazio

CCC 0192-8651 / 98 / 070685-10

Introduction

Classical molecular dynamics (MD) is used to study the properties of liquids, solids, and molecules.[1,2] The Newton equation of motion for each particle of the system is solved by numerical integration and its trajectory is obtained. From this microscopic point of view, many microscopic and macroscopic properties can be obtained. The need for numerical integration limits the time step to the femtosecond scale and makes MD simulation a very time consuming task. Therefore, considerable efforts have been concentrated on optimizing MD codes on parallel computers of different architectures.

Parallel computers are frequently described as belonging to one of two types: single instruction stream multiple data stream (SIMD), or multiple instruction stream multiple data stream (MIMD). In general, SIMD machines have a simpler architecture, but they have hardware limitations because the same instruction is executed in parallel on every SIMD processor and, furthermore, some SIMD machines do not have local addressing; that is, the processors are not able to access their own memory using different local addresses. In recent years, several MD codes have been implemented on MIMD architectures with a few dozen of processors[3–5] and, more recently, also on 100- to 1000-processor MIMD machines.[6–8] Parallel implementations of biological MD programs such as CHARMM[9] and GROMOS[10] on MIMD machines have been discussed in the literature.[8,11–13]

Less work has been done using SIMD systems.[14–17] In general, they make use of the full connectivity computation; that is, all atom pair interactions are calculated, and are useful for long-range force systems. This is due to the difficulty of using pair lists of nonbonded atoms on SIMD machines with no local addressing.

In the present study we propose an algorithm that permits the use of pair lists in a MD code for a SIMD machine with no local addressing. The algorithm requires simultaneous use of multiple time step[18] and geometric decomposition[13] methods. In addition, the systolic loop[16] method is used to further reduce computation time.

The method was tested on Quadrics computers,[19–21] a class of SIMD machines developed by INFN and Alenia Spazio, for configurations of 32, 128, and 512 processors. Quadrics is the only massive parallel computer developed with fully European technology. As the Europort2/PACC project[11] has shown, the scalability for a MD code on MIMD architecture, for complex systems such as a protein in solution, is generally satisfactory only up to 12–16 nodes.

Moreover, there are interesting projects being undertaken on mixed architecture MIMD/SIMD machines that could supply the computational power of a SIMD machine, together with the flexibility of a MIMD. It is therefore worthwhile to determine whether these machines are able to perform such calculations.

The following molecular systems have been used as tests:

- *System 1*: Box of 1536 water molecules (4608 atoms).
- *System 2*: Box with a BPTI (bovine pancreatic trypsin inhibitor) molecule and 2712 water molecules (8704 atoms).
- *System 3*: Box with a SOD (superoxide dismutase) dimer and 4226 water molecules (15,360 atoms).
- *System 4*: Box with a SOD (superoxide dismutase) dimer and 9346 water molecules (30,720 atoms).

It should be noted that system 4 is nearly the same as test case I3, used as the industrial benchmark in the framework of the Eurosport2/PACC project[11] (system 4 has 9346 water molecules whereas test case I3 has 9436 water molecules). The results show that the speed-up of the algorithm is comparable to those obtained with MIMD machines.

Hardware

We tested the method on a Quadrics machine, Alenia Spazio's supercomputer derived from the APE100 (Array Processor Elaborator) project, developed by INFN.[19–21] These computers are a family of massively parallel SIMD machines with implementations from 8 to 2048 processors. The biggest implementation allows a peak computing power of 100 GFlops in single precision (32-bit processors).

The processors are arranged in a three-dimensional (3D) cubic mesh and can exchange data with the six neighboring nodes, with periodic boundaries. Each processor board contains eight processors (floating point units) with their own

memory (4 megabytes). Up to 16 boards can be put into a crate. Configurations with more than 128 processors are made up connecting crates of $8 \times 4 \times 4$ (128) nodes.

The Quadrics controller board contains one integer CPU (Z-CPU), which controls the flux of the program and the integer arithmetic. The language used is called Tao, a Fortran-like high-level parallel language, which can be modified and extended through a dynamic ZZ parser. The Quadrics machine is connected to a host computer (a Sun Sparc-10 or -20). A host interface based on a HIPPI standard, which allows an I/O speed between the host and Quadrics of 20 MB/s, has recently been developed. The tests on the sequential machine have been run on a DEC-alpha 3000/500 machine. Barone et al.[22] compared the accuracy of Quadrics in the field of molecular dynamics with that of a conventional computer to assess the limits of the single precision.

Molecular Dynamics

In a molecular dynamics simulation, the classical equations of motion for the system of interest are integrated numerically by solving Newton's equations of motion:

$$m_i \frac{d^2 r_i}{dt^2} = -\nabla_i V(r_1, r_2, \ldots, r_n)$$

The solution gives the atomic positions and velocities as a function of time. The knowledge of the trajectory of each atom permits study of the dynamic or statistical properties of the system. The form of the interaction potential is complex and it includes energy terms that represent bonded and nonbonded (van der Waals and Coulombic) interactions:

$$\begin{aligned} V(r_1, r_2, \ldots, r_N) &= \sum_{\text{bonds}} \frac{1}{2} K_b [b - b_0]^2 + \sum_{\text{angles}} \frac{1}{2} K_\theta [\theta - \theta_0]^2 \\ &+ \sum_{\substack{\text{improper} \\ \text{dihedrals}}} \frac{1}{2} K_\xi [\xi - \xi_0]^2 \\ &+ \sum_{\text{dihedrals}} K_\varphi [1 + \cos(n\varphi - \delta)] \\ &+ \sum_{\text{pairs}\,(i,j)} \left[4\varepsilon_{ij} \left(\frac{\sigma_{ij}^{12}}{r_{ij}^{12}} - \frac{\sigma_{ij}^{6}}{r_{ij}^{6}} \right) + \frac{q_i q_j}{4\pi\varepsilon r_{ij}} \right] \end{aligned}$$

The first four terms represent the bonded potential. b, b_0, and K_b are the actual bond length, its reference value, and the bond stretching force constant, respectively. θ, θ_0, and K_θ are the actual bond angle, its reference value, and the angle bending force constant, respectively, ξ, ξ_0, and K_ξ are the actual improper dihedral angle, its reference value, and the improper dihedral angle bending force constant, respectively. φ, K_φ, n, and δ are the actual dihedral angle, its force constant, the multiplicity, and the phase, respectively. The last term in the equation includes the nonbonded, van der Waals, and Coulombic terms. ε_{ij} and σ_{ij} are the dispersion well depth and the Lennard–Jones distance, q_i and q_j are the electrostatic atomic charges, r_{ij} is the distance between them, and ε is the dielectric constant.

The time step used for the numerical integration is in the femtosecond scale. The highest frequency of bond vibrations would require a time step ≤ 0.5 fs; however, if the simulation is performed with constant bond lengths, the time step can be ≤ 2 fs. For this reason, many MD codes perform simulations with constant bond lengths.

The most frequently used algorithm to perform MD simulation at constant bond lengths is the SHAKE algorithm based on an iterative procedure.[23]

Computational Algorithm for Nonbonded Interactions

In a MD program, the calculation of the nonbonded forces is the most time-consuming task—in fact, it takes about 90% of the computational time, depending on the protocol used.

One of the most frequently used techniques to reduce the number of nonbonded forces is the cut-off criterion. With this method the interactions between atoms beyond a cut-off distance are neglected. If the cut-off radius is appropriate the lost energy contribution to the global potential is small. During a small number of steps the pairs of interacting atoms are considered to remain the same so that it is possible to create a list of these interactions, the nonbonded pair list, which will be updated every n steps (n is generally equal to 10). The number of nonbonded interactions is $N(N-1)/2$ (N is the number of atoms), so that it is proportional to N^2. The use of the cut-off criterion reduces this number to kN (k is a constant).

Unfortunately, the cut-off criterion is not directly applicable to SIMD architecture, as the same instruction is executed at each time on each processor and, consequently, it is not possible to have a local branch (the cut-off condition) in the program flow. Moreover, on Quadrics it is not possible to have a local pair list on each node because local addressing of arrays is not possible. This explains the lower level of efficiency of a MD code on a SIMD machine with respect to a MIMD one. We have recently developed an algorithm, the global cut-off algorithm (CGA), based on a probabilistic approach, which allows the use of the cut-off condition on a SIMD machine with no local addressing.

Because the calculation of bonded interactions and the integration of the trajectory take a small amount of the total calculation time, in the present version we have chosen to carry them out on the front-end computer and to perform only the calculations of the pair list and of the nonbonded forces using the SIMD machine. It is, of course, possible to perform all force calculations and integration in the parallel machine using, for instance, the replicated data method.[8]

GEOMETRIC DECOMPOSITION

The assignment of the atoms to the nodes is obtained by a dynamically geometric decomposition[13] in such a way that the same number of atoms is assigned to each node. In what follows, we discuss a decomposition for a bidimensional case; the extension to a third dimension is straightforward: given the bidimensional box of Figure 1a and a 2D parallel topology of $n = n_x \times n_y$ processors, with $n_x = n_y = 2$, the box is first divided into n_x boxes along the x-axis, as shown in Figure 1b, each containing the same number of atoms. Each box is successively divided into n_y boxes along the y-axis in such a way that each one of the $n_x \times n_y$ boxes contains the same number of atoms (Fig. 1c). When, as in a real case, a third dimension exists, a successive division along the z-axis has to be performed.

It is obvious that, before performing any division along a given axis, it is necessary to sort the atoms of each box along that axis.

The density of a molecular system, such as a protein, is not uniform; thus, the boxes do not have the same axis lengths. However, these differences do not significantly reduce the efficiency of the GCA described in what follows.

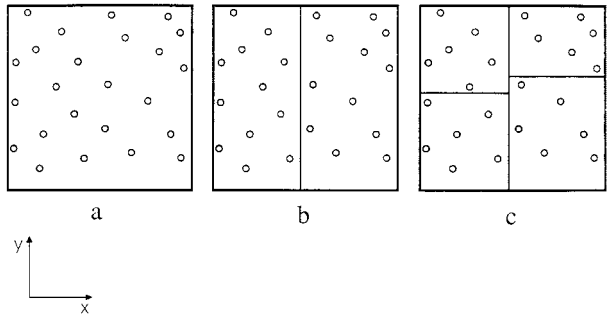

FIGURE 1. Domain decomposition of the molecular system in boxes with the same number of atoms, for a bidimensional case.

SYSTOLIC LOOP METHOD

Quadrics topology makes it possible to use a systolic loop to calculate the nonbonded interactions between the atoms assigned to the different nodes. The systolic loop method is one of the most efficient algorithms for calculation of two-body interactions on MIMD and SIMD machines.[14, 16, 24, 25] The systolic loop algorithm passes the coordinates of all atoms around a ring of P processors in P/2 steps, such that half of the coordinates passes every processor exactly once (transient atoms). Each node also stores the coordinates of a group of atoms of the overall system (resident atoms). During the systolic cycle each processor evaluates and accumulates the interactions of the resident atoms with the transient ones. Only half of the atoms have to pass in each computational node as a consequence of the reciprocity of the interactions.

The systolic loop path for a 32-node Quadrics machine is shown in Figure 2. This machine has two nodes along the y and z directions and eight along the x direction.

The geometric decomposition of the system permits limitation of the search for nonbonded interactions only to the neighboring processors nearer than the cut-off radius, so that, depending on the number of nodes and on the system size, it is generally not necessary to perform the complete systolic loop. The computed forces are passed back to the owning processor to accumulate the full force.

GLOBAL CUT-OFF ALGORITHM

On a SIMD machine, all nodes simultaneously evaluate an interaction, but the atom pairs in each node are different. Figure 3 shows, as an example,

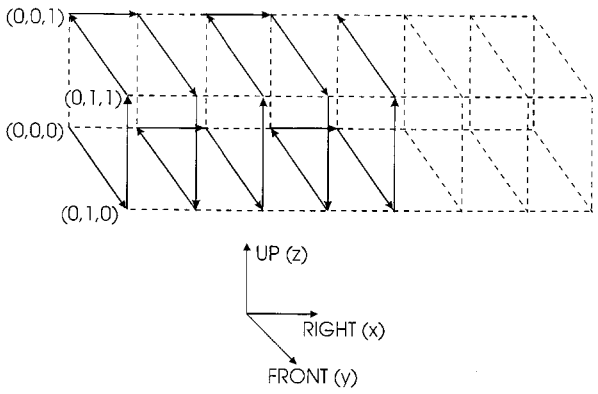

FIGURE 2. Systolic loop path for node (0, 0, 0) of a 32-node Quadrics machine. The transient groups of atoms visit only four neighboring $y-z$ planes, based on Newton's third law.

the case with four nodes: suppose that each node is evaluating the interaction a_i; in this case, all a_i interactions fall within the cut-off radius. When the interactions are of the b_i type all the distances fall outside the cut-off radius and the interactions b_i are skipped. In the case of interactions of type c_i the interaction is outside the cut-off radius in nodes 1, 2, and 3, but it is inside the cut-off radius in node 4, so that all nodes have to calculate this interaction and only will be saved in the forces calculation. If the atoms in each node are ordered randomly, the interactions of type c_i result in being the most frequent.

The basis idea of the global cut-off algorithm (GCA) is to maximize the occurrence of interactions of type a_i and b_i and, conversely, reduce the occurrence of interactions of type c_i. To this end, it is necessary that the atoms in all nodes are sorted with the same criterion. Different types of sorting give comparable results. We have chosen the one shown in Figure 4. After this sorting procedure, a list of the interactions of type a_i and c_i is created in the integer CPU (Z-CPU) of the SIMD machine. This list is equivalent, but not identical, to the nonbonded pair list used in most MD programs and will be referred as the *nonbonded pair list*.

The ordering procedures for the domain decomposition and the sorting procedure previously described are time consuming and have to be performed on the host serial machine; however, as will be shown, they have to be performed every 100 to 200 steps so that they do not significantly affect the global computation time.

The global cut-off condition is based on a probabilistic approach, so that the number of pair interactions to be calculated is larger than the actual number of pairs included within the r_{cut} distance. Depending on the molecular system and on the number of nodes, the ratio between the number of the calculated interactions and the number of interactions actually included within the cut-off dis-

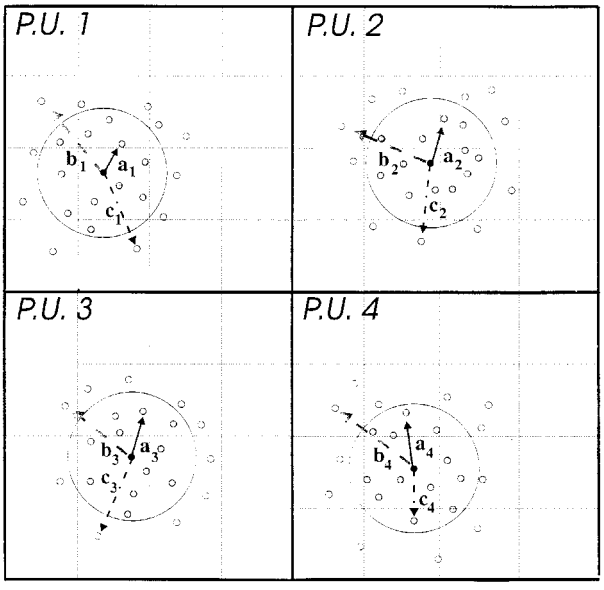

FIGURE 3. Different types of interactions in a case of four nodes. a_i, all the interactions fall within the cut-off distance; b_i, all the interactions fall outside the cut-off distance; c_i, one interaction falls within the cut-off, whereas three fall outside the cut-off. P.U. = processor unit.

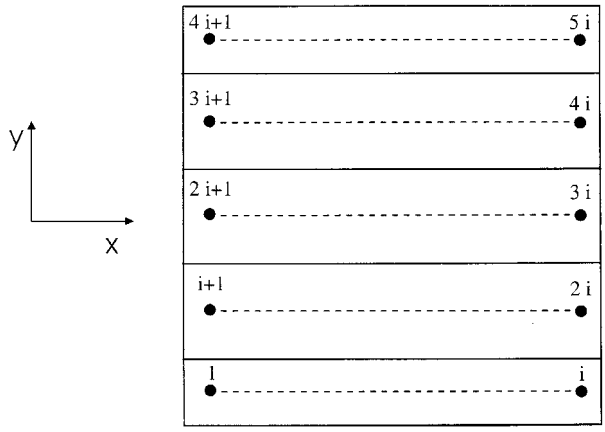

FIGURE 4. Sorting of atoms in each node for a bidimensional case. The atoms are represented as full circles.

tance varies from the three to five. In this sense, the GCA is not very efficient. However, it must be noted that almost all pair interactions within a distance $r_{cut} < r \leq r_{cut} + \Delta r$, with $\Delta r \cong 0.2$ nm, are calculated. As an example, given $r_{cut} = 0.6$ nm, 94% of the interactions in the range $0.6 < r \leq 0.8$ nm are calculated and only 6% of pairs in this range are lost. This suggests adoption of a cut-off value of r_{GCA}, to be used in the global cut-off condition, somewhat shorter than the actual cut-off, r_{cut}, desired for the simulation. In the previous example, with $r_{GCA} = 0.6$ and $r_{cut} = 0.8$ nm, the number of pairs to be calculated is roughly two-to-threefold larger than the actual number of pairs within the r_{cut} distance. The remaining 6% of interactions in the range $0.6 < r \leq 0.8$ nm have to be calculated separately.

It is well known that nonbonded forces vary more slowly than the bonded ones. Moreover, nonbonded forces at large distances vary slower than nonbonded forces at short distances. This suggests updating of the forces at different steps, according to their nature (bonded and nonbonded) and to the distance of the interaction. The short-range interactions can be evaluated every step, and long-range interactions every m steps. Accordingly, the few interactions in the range $0.6 < r \leq 0.8$ nm that were lost using $r_{GCA} = 0.6$ nm can be updated every m steps. As these interactions are calculated while evaluating the *nonbonded pair list* (i.e., updated every n steps), we have chosen $m = n = 10$.

It must be noted that there are now two shells: an inner shell ($r \leq 0.6$ nm) and an outer shell ($0.6 < r \leq 0.8$ nm). All interactions are evaluated every m steps, whereas only those interactions corresponding to the inner shell are evaluated every step. It is therefore not necessary to have a skin distance and to store a list of atom pairs outside the outer shell.

In the present study it is seen that most interactions in the range 0.6 to 0.8 nm are evaluated every step and only a few of them are evaluated every m steps. According to all of the MTS algorithms, this choice does not affect the numerical accuracy; in fact, the same accuracy is obtained when an interaction, within the outer shell, is evaluated every short time step or every long time step. However, as every long time step all interactions within the outer shell are evaluated, it is possible to perform the MTS according to the classical procedure; that is, by collection all the interactions within the outer shell at every long time step and collecting only the interactions within the inner shell at every short time step. Among several algorithms proposed for the multiple time step (MTS)[1, 18, 26] we have chosen the one developed by Scully and Hermans.[18]

It must be noted that all nonbonded interactions (van der Waals and Coulombic) between bonded and nonbonded atoms are calculated in this step, but the interactions between bonded atoms are not saved. This is obtained by attaching to each atom a list of the atoms bonded to itself. This procedure is certainly not efficient, but the time required to perform it is negligible. In the following tests, the values of r_{GCA} and r_{cut} are fixed at 0.6 and 0.8 nm, respectively.

Results

Table I shows the number of interactions within the cut-off radius r_{cut} compared with the number of interactions to be evaluated with the GCA. It can be noted that the number of interactions for calculation is two to three times the actual number of interactions within the cut-off radius. The time

TABLE I.
Number of Actual Interactions, *N*, within a Cut-Off Radius (r_{cut}) = 0.8 nm Compared with the Number of Interactions Calculated Using GCA on 32-, 128- and 512-Node Quadrics Machines.

	N	32 nodes	128 nodes	512 nodes
System 1 (4608 atoms)	488,847	706,944	840,320	873,984
System 2 (8704 atoms)	891,252	1,612,864	1,810,944	2,070,016
System 3 (15,360 atoms)	1,487,054	3,153,344	3,768,576	4,452,864
System 4 (30,720 atoms)	3,129,972	6,754,240	8,249,216	9,938,944

required for performing the MD simulation with the present code is the sum of the following steps:

1. Ordering procedure (performed every k steps by the host computer).
2. Calculation of the *nonbonded pair list* (performed every n steps by Quadrics).
3. Calculation of the nonbonded forces (performed every step by Quadrics).

3a. Calculation of the bonded forces by the host computer while performing step 3.

4. I/O host ↔ Quadrics.
5. SHAKE and integration (performed by the host).

The ordering procedure is a time-consuming task and, due to the diffusion of the system, it has to be periodically repeated every k steps. If no reordering is performed the *nonbonded pair list* will include an increasing number of interactions, thus increasing the computational time. Figure 5 shows the number of interactions to be evaluated versus the number of steps when the ordering procedure is performed at the beginning and not updated, for system 4 on a 32-node machine.

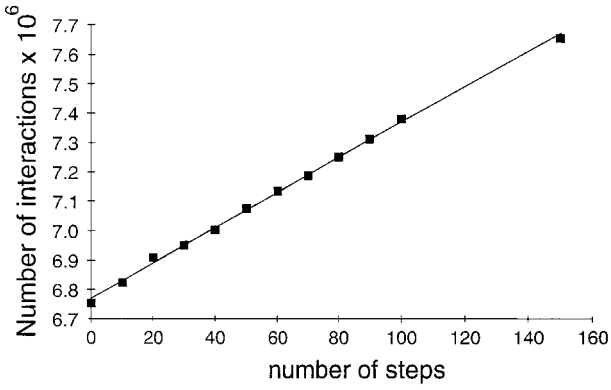

FIGURE 5. Number of interactions to be calculated versus the number of steps for system 4 on a 32-node machine. The atoms are not reordered during the simulation.

The loss of performance is nearly linear, being ~ 0.08% per step. The optimum k value depends on the time required for the ordering procedure and on the time required for items 2 and 3. It shows that, for all the systems and all the different numbers of nodes, the optimum k value is in the range of 100 to 200 steps. The ordering procedure for system 4 on a Sun Sparc-20 (the host computer) required 20 seconds, so that its cpu time per step is in the range of 100 to 200 ms.

The *nonbonded pair list* is evaluated every n steps ($n = 10$ in the present case) and the nonbonded interactions are evaluated every step. The average cpu time required for these tasks is reported in Table II for different systems and different numbers of nodes. It should be emphasized that the parallel machines perform the calculation on a number of pairs two to three times larger than the serial one. The almost linear scalability of these task is also worthy of note.

The data transferred from the host to the Quadrics and vice versa after each reordering step (i.e., every k ~ 100 steps) are reported in Table III (upper panel); that is, 23 words per atom.

The data to be transferred every step are reported in Table III (lower panel)—9 words per atom. The average time spent in transmission depends on the speed of information transfer. Taking into account the speed of the HIPPI interface (20 MB/s) the average I/O times per step required for systems 1 to 4 are 9, 16, 28, and 55 ms/step, respectively.

The sums of times for items 1 to 4 with different numbers of nodes are reported in Table IV. Figure 6 shows the number of steps per second versus the

TABLE II.
cpu Times for Nonbonded Interactions (Expressed in Milliseconds per Step).

	DEC-alpha 3000 / 500	32 nodes	128 nodes	512 nodes
System 1 (4608 atoms)	2943	632	174	68
System 2 (8704 atoms)	6097	1650	386	107
System 3 (15,360 atoms)	10,928	3274	894	214
System 4 (30,720 atoms)	27,610	6208	1624	466

TABLE III.
Data Transfer Results.

	Quantity transferred (words per atom)
Data transferred every *k* steps	
From host to Quadrics	
Coordinates of the atom	3
Coordinates of the geometric center of the charge group	3
Electric Charge	1
Sequential atomic number	1
van der Waals parameters	2
Exclusions 1–3	6
Exclusions 1–4	4
From Quadrics to host	
Forces	3
Total	23
Data transferred every step	
From host to Quadrics	
Coordinates of the atom	3
Coordinates of the geometric center of the charge group	3
From Quadrics to host	
Forces	3
Total	9

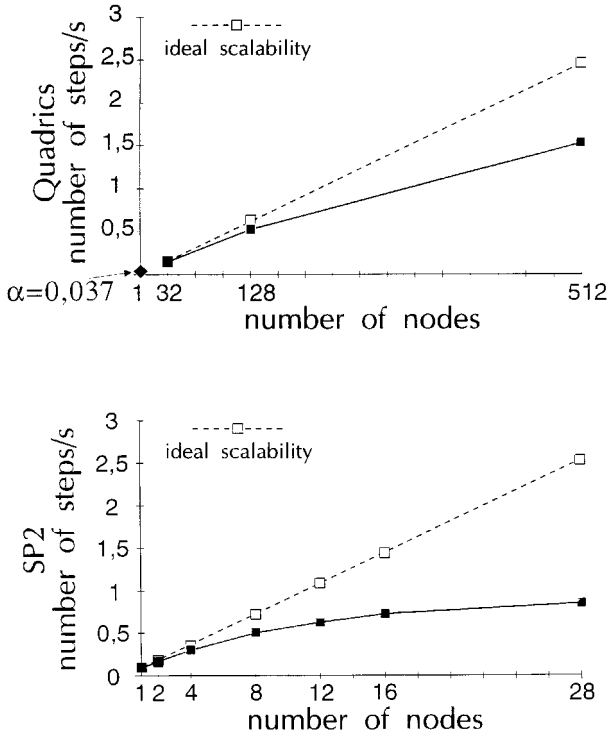

FIGURE 6. Number of steps per second versus the number of nodes for system 4. (a) Quadrics; (b) IBM SP2. The ideal scalability is expressed as a dashed line. Single processor DEC-alpha 3000 / 500 timing is shown for comparison.

number of nodes for Quadrics (Fig. 6a) and for an IBM-SP2 MIMD machine (Fig. 6b) for the same system (system 4). For clarity, the CPU times required for SHAKE and integration are not included. The code used for the MIMD machine was developed within the Europort2/PACC project.[11]

Figure 6 also shows that no significant advantage is obtained with the MIMD machine when the number of nodes is ≥ 12, whereas, a good scalability, up to 512 nodes, is obtained with the SIMD machine.

TABLE IV.
Total cpu Times Including Nonbonded Interactions, Ordering Procedure, and I / O Host Quadrics (Milliseconds per Step).

	32 nodes	128 nodes	512 nodes
System 1 (4608 atoms)	680	205	90
System 2 (8704 atoms)	1790	470	175
System 3 (15,360 atoms)	3540	1050	335
System 4 (30,720 atoms)	6660	1910	685

To complete the evaluation of the total time/steps of the present MD code, the times required on the host for the bonded interactions, SHAKE, and integration have to be calculated. The times required for these tasks, with the present MD code, are reported in Table V. It should be noted that the calculation of bonded interactions is performed by the host, whereas Quadrics computes the nonbonded interactions. As the former's calculation time is less than the latter, this task does not require any extra cpu time.

The integration task requires less cpu time than the nonbonded interaction calculation time (see Table IV) on a 32- or 128-node machine, and a comparable amount of time on a 512-node machine. Therefore, this task must be parallelized for machines with hundreds or thousands of processors. The integration can be implemented easily on a SIMD machine by, for example, the replicated data procedure.

The cpu time required to perform the SHAKE algorithm on the host is the actual bottleneck. It is

TABLE V.
cpu time on the Host (Sun Sparc-20) for Calculation of Bonded Interactions, SHAKE, and Integration (expressed in Milliseconds per Step).

	Bonded Interactions	SHAKE	Integration
System 1 (4608 atoms)	41	236	57
System 2 (8704 atoms)	96	488	117
System 3 (15,360 atoms)	73	1102	161
System 4 (30,720 atoms)	194	3013	437

very difficult to implement the SHAKE algorithm on a SIMD machine; therefore, an alternative procedure must be chosen. The MTS procedure can be used to evaluate the bond vibrations without reducing the time steps required for the nonbonded interactions.

Conclusions

The results reported in the present work show that the GCA permits use of pair lists even on a SIMD machine with no local addressing, thus overcoming one of the most severe disadvantages of SIMD vs. MIMD machines. The penalty to be paid is the number of interactions per step to be calculated; that is, two to three times the actual number of interactions.

The tests performed on Quadrics computers for configurations of 32, 128, and 512 nodes, for systems of different sizes, up to 30,000 particles, show that the scalabilities and performances are satisfactory and comparable to those obtained with MIMD machines. At the present time, the only routines for the calculation of the pair lists and nonbonded interactions have been parallelized. We have shown that the bonded forces can be calculated by the host while the parallel machine calculates the nonbonded ones. Also, the integration task can be calculated by the host if the parallel machine has up to tens of processors. With hundreds or thousands of processors this task must also be parallelized.

The SHAKE algorithm, which allows one to perform MD calculations at constant bond lengths, is the actual bottleneck of the calculation and its implementation is difficult with the parallel machine. We suggest the use of the MTS to evaluate the bond vibration contributions.

Acknowledgments

The authors thank ENEA for the use of the 128 and 512 Quadrics machines. One of us (R.B.) thanks ENEA for his hospitality during the realization of this work.

References

1. M. P. Allen and D. J. Tidsley, *Computer Simulation of Liquids*, Clarendon Press, Oxford, 1987.
2. W. F. van Gunsteren and H. J. C. Berendsen, *Angew. Chem. Int. Ed. Engl.*, **29**, 992 (1990).
3. M. R. S. Pinches, D. J. Tildesley, and W. Smith, *Mol. Simul.*, **6**, 51 (1991).
4. H. Heller, H. Grubmüller, and K. Shulten, *Mol. Simul.*, **5**, 133 (1990).
5. D. C. Rapaport, *Comput. Phys. Commun.*, **62**, 217 (1991).
6. P. A. J. Hilbers and K. Esselink, *Computer Simulation in Chemical Physics*, M. P. Allen and D. J. Tildesley, Eds., Kluwer Academic Publisher, London, 1993, p. 473.
7. S. P. Plimpton, *J. Comput. Phys.*, **117**, 1 (1995).
8. S. P. Plimpton and B. Hendrickson, *J. Comput. Chem.*, **17**, 326 (1996).
9. B. R. Brooks, R. E. Bruccoleri, B. D. Olafson, D. J. States, S. Swaminathan, and M. Karplus, *J. Comput. Chem.*, **4**, 187 (1983).
10. W. F. van Gunsteren and H. J. C. Berendsen, *GROMOS: Groningen Molecular Simulation (GROMOS) Library Manual*, Biomos, Groningen, 1987.
11. D. G. Green, K. E. Meacham, M. Surridge, F. van Hoesel, and H. J. C. Berendsen, *Methods and Techniques in Computational Chemistry: Metecc-95*, E. Clementi and G. Corongiu, Eds., STEF, Cagliari, 1995, p. 435.
12. S. L. Lin, J. Mellor-Crummey, B. M. Pettit, and G. N. Phillips, *J. Comput. Chem.*, **13**, 1022 (1992).
13. W. T. Clark, R. Hanxleden, J. A. McCammon, and L. R. Scott, *Technical Report CRPC-TR93356-S*. Center for Research on Parallel Computation, Houston, TX, November 1993.
14. A. Windemuth and K. Shulten, *Mol. Simul.*, **5**, 353 (1991).
15. D. Fincham, *Mol. Simul.*, **1**, 1 (1987).
16. K. M. Nelson, C. F. Cornwell, and L. T. Wille, *Comput. Mater. Sci.*, **2**, 525 (1994).
17. P. Ballestrero, P. Baglietto and C. Ruggiero, *J. Comput. Chem.*, **17**, 469 (1996).

18. J. L. Scully and J. Hermans, *Mol. Simul.*, **11**, 67 (1993).
19. A. Bartoloni, G. Bastianello, C. Battista, S. Cabasino, F. Marzano, P. S. Paolucci, J. Pech, F. Rapuano, E. Panizzi, R. Sarno, G. M. Todesco, M. Torelli, W. Tross, P. Vicini, N. Cabibbo, A. Fucci, and R. Tripiccione, *Int. J. Modern Phys.*, **4**, 969 (1993).
20. A. Bartoloni, G. Bastianello, C. Battista, S. Cabasino, F. Marzano, P. S. Paolucci, J. Pech, F. Rapuano, E. Panizzi, R. Sarno, G. M. Todesco, M. Torelli, W. Tross, P. Vicini, N. Cabibbo, and R. Tripiccione, *Int. J. Modern Phys.*, **4**, 955 (1993).
21. The APE Collaboration, *Int. J. High Speed Comput.*, **5**, 637 (1993).
22. L. M. Barone, R. Simonazzi, and A. Tenenbaum, *Comput. Phys. Commun.*, **90**, 44 (1995).
23. J. P. Ryckaert, G. Ciccotti, H. J. C. Berendsen, *J. Comput. Phys.*, **23**, 327 (1977).
24. W. Smith, *Comput. Phys. Commun.*, **62**, 229 (1991).
25. J. F. Janak and P. C. Pattnaik, *J. Comput. Chem.*, **13**, 533 (1992).
26. M. Tuckerman, B. J. Berne, and G. J. Martyna, *J. Chem. Phys.*, **97**, 1990 (1992).