ELSEVIER

Computer Physics Communications 104 (1997) 59–69

Computer Physics Communications

Parallel multilevel preconditioned conjugate-gradient approach to variable-charge molecular dynamics

Aiichiro Nakano [1]

Department of Computer Science, Concurrent Computing Laboratory for Materials Simulations, Louisiana State University, Baton Rouge, LA 70803-4020, USA

Received 31 March 1997

Abstract

Physical realism of molecular dynamics (MD) simulations is greatly enhanced by incorporating variable atomic charges which adapt to the local environment dynamically. In the electrostatic plus (ES+) model, atomic charges are determined to equalize electronegativity. However, this model involves costly minimization of the electrostatic energy at each MD step. A preconditioned conjugate-gradient method is developed for this minimization problem by splitting the Coulomb-interaction matrix into short- and long-range components; the computationally less intensive short-range matrix is used as a preconditioner. This preconditioning scheme is found to speed up the convergence significantly. Numerical tests involving up to 26.5 million atoms are performed on a parallel computer, and the preconditioner is shown to improve the parallel efficiency by increasing data locality. The computational cost is further amortized due to the algorithmic similarity to the multiple-time-scale MD. © 1997 Elsevier Science B.V.

1. Introduction

Recently, large-scale molecular dynamics (MD) simulations have been established as a new research mode for understanding how atomistic processes are related to macroscopic materials phenomena such as fracture [1–3]. The current focus of research is how to enhance the physical realism of these simulations. For example, conventional interatomic potential functions used in MD simulations are often fitted to bulk solid properties, and they are not transferable to systems containing defects, cracks, surfaces, and interfaces. In these systems, the partial charges on the atoms vary dynamically according to the change in the local environment. This environment-dependent charge distribution is crucial for the physical properties of these systems including the fracture toughness [4]. Transferability of interatomic potentials is greatly enhanced by incorporating variable atomic charges which dynamically adapt to the local environment. Various approaches have been developed to model charge transfers in condensed-phase simulations [5–10]. Recently a simple semiempirical approach has been developed in which atomic charges are determined to equalize electronegativity [8–10]. In the electrostatic

[1] E-mail: nakano@bit.csc.lsu.edu; URL: http://www.cclms.lsu.edu.

PII S0010-4655(97)00041-6

plus (ES+) model [10], the electrostatic energy due to charge transfer is merged seamlessly with standard potential energies representing nonionic (such as metallic and covalent) bonding.

The increased physical realism in the ES+ model is accompanied by increased computational cost for minimizing the electrostatic energy at every MD step. In the extended Lagrangian formalism [9], a fictitious Newtonian dynamics is introduced for atomic charges to minimize the electrostatic energy concurrently with MD simulations. However, this scheme is not efficient when high-quality solutions are required for the charges. This is the case for (i) simulation of fracture in which new surfaces are constantly created, and (ii) calculation of dynamical matrices to study vibrational properties in which well-converged solutions are required.

In this paper, the costly minimization of the electrostatic energy is dealt with the preconditioned conjugate-gradient method [11–13]. We develop a preconditioner by splitting the Coulomb-interaction matrix into short- and long-range components; the short-range matrix is used as a preconditioner. The new scheme is also implemented on a parallel computer, and is found to improve the parallel efficiency significantly. In the next section, we describe this multilevel preconditioned conjugate gradient (MPCG) method. In Section 3, a parallel implementation of the MPCG method is presented. Section 4 contains numerical experiments, and finally Section 5 includes discussions.

2. Method

2.1. Variable charge molecular dynamics

In the electrostatic plus (ES+) approach [10], the potential energy of the system is expressed as a sum of nonionic, V_0, and electrostatic, V_{es}, terms: $V = V_0 + V_{es}$. The electrostatic energy is a function of atomic positions, $\{\vec{x}_i | i = 1, \ldots, N\}$, and atomic charges, $\{q_i | i = 1, \ldots, N\}$ (N is the number of atoms),

$$V_{es}(\{\vec{x}_i\}, \{q_i\}) = \sum_i \chi_i q_i + \frac{1}{2} \sum_i J_i q_i^2 + \sum_{(i,j)} \int d^3x_1 \int d^3x_2 \frac{\rho_i(\vec{x}_1; q_i)\rho_j(\vec{x}_2; q_j)}{|\vec{x}_1 - \vec{x}_2|} . \tag{1}$$

In Eq. (1), the first two terms represent intra-atomic electrostatic energies, where χ_i and J_i denote the electronegativity and self-Coulomb repulsion of the ith atom [8–10]. In the last term of Eq. (1), the contributions from atomic pairs (i, j) arise from interatomic Coulomb interaction. Atomic charge distribution is modeled by a Slater-type orbital, $\rho_i(\vec{x}; q_i) = (q_i \zeta_i^3/\pi) \exp[-2\zeta_i |\vec{x} - \vec{x}_i|]$, where ζ_i^{-1} is the decay length for atomic orbitals[2].

In MD simulations, atomic trajectories $\{\vec{x}_i(t)\}$ are obtained for a time sequence, t, $t + \Delta t$, $t + 2\Delta t, \ldots$, by numerically integrating Newton's second law of motion,

$$m_i \frac{d^2}{dt^2} \vec{x}_i = -\frac{\partial}{\partial \vec{x}_i} V , \tag{2}$$

where m_i is the mass of the ith atom. At each MD time step, the atomic charges q_i are determined to minimize the electrostatic energy, $V_{es}(\{\vec{x}_i(t)\}, \{q_i\})$, subject to the charge-neutrality constraint, $\sum_i q_i = 0$. This constrained minimization is equivalent to the electronegativity equalization condition that the chemical potentials $\partial V_{es}/\partial q_i$ be equal for all the atoms. This condition leads to a linear equation system,

$$\sum_j M_{ij} q_j = \mu - \chi_i , \tag{3}$$

[2] In [10], atomic charge distribution in Al/Al_2O_3 systems is modeled by an extended form, $\rho_i(\vec{x}; q_i) = z_i \delta(\vec{x} - \vec{x}_i) + [(q_i - z_i)\zeta_i^3/\pi] \exp[-2\zeta_i |\vec{x} - \vec{x}_i|]$.

where M_{ij} denotes the Coulomb-interaction matrix, and the Lagrange's multiplier μ is determined to satisfy the constraint. In practice we solve two linear systems concurrently,

$$\sum_j M_{ij}\tilde{q}_j = -\chi_i\,, \tag{4}$$

$$\sum_j M_{ij}\hat{q}_j = -1\,, \tag{5}$$

and the solution to Eq. (3) is obtained as $q_i = \tilde{q}_i - \mu\hat{q}_i$ where the chemical potential is calculated as $\mu = \sum_i \tilde{q}_i / \sum_i \hat{q}_i$.

2.2. Multilevel preconditioned conjugate gradient method

We use the conjugate gradient (CG) method [11–13] to solve the linear systems, Eqs. (4) and (5). This iterative method proceeds by generating a sequence of iterates $\boldsymbol{q}$ as successive approximations to the solution, residuals $\boldsymbol{r}$ corresponding to the iterates, and search directions (conjugate gradients) $\boldsymbol{p}$ used in updating the iterates and residuals. Here we use a vector notation such that $\boldsymbol{q} = (q_1, q_2, \ldots, q_N)$. In the preconditioned CG method, a preconditioning matrix is used to transform the linear system into an equivalent one with improved spectral properties [11–13]. A good preconditioner approximates the original matrix $\boldsymbol{M}$, but for which solving the linear system is much easier.

The multilevel preconditioned conjugate gradient (MPCG) method is based on the decomposition of the electrostatic energy into short- and long-range components, $V_{\rm es} = V_s + V_l$. Accordingly, the Coulomb interaction matrix is decomposed as $\boldsymbol{M} = \boldsymbol{M}_s + \boldsymbol{M}_l$. The sparse short-range matrix $\boldsymbol{M}_s$ is then used as a preconditioner. A pseudocode for the MPCG algorithm is given in Appendix A. The algorithm has a doubly-nested loop structure. The inner loop computes a preconditioning vector $\boldsymbol{z}$ by solving the linear system, $\boldsymbol{M}_s\boldsymbol{z} = \boldsymbol{r}$, using the CG method. Due to the short range of $\boldsymbol{M}_s$, this system is easier to solve than the original system. The outer loop solves the preconditioned linear system which involves the dense matrix, $\boldsymbol{M}$.

We have implemented two decomposition schemes based on the Ewald method [14–16] and the fast multipole method (FMM) [16–29]. In the Ewald method, the Coulomb potential is decomposed into (i) a sum in the real space; (ii) a sum in the reciprocal space; (iii) and a constant term [14–16]. We define V_s to be the sum of the real-space and constant terms,

$$V_s = \sum_i \chi_i q_i + \tfrac{1}{2}\sum_i J_i q_i^2 + \sum_{(i,j)} q_i q_j \left[\frac{1}{r_{ij}}\operatorname{erfc}\left(\frac{x_{ij}}{2\gamma}\right) + \nu_{ij}(x_{ij})\right] - \frac{1}{2\sqrt{\pi}\gamma}\sum_i q_i^2\,, \tag{6}$$

and V_l to be the reciprocal-space term,

$$V_l = \sum_{\vec{k}(\neq 0)} \frac{2\pi}{\Omega k^2}\exp(-\gamma^2 k^2)\left|\sum_i q_i \exp(i\vec{k}\cdot\vec{x}_i)\right|^2\,. \tag{7}$$

In Eqs. (6) and (7), γ is the Ewald truncation parameter, Ω is the volume of the system,

$$\nu_{ij}(x) = -\frac{(1-\kappa)^2}{4x}(2+\kappa+\zeta_i x)\exp(-2\zeta_i x) - \frac{(1+\kappa)^2}{4x}(2-\kappa+\zeta_j x)\exp(-2\zeta_j x)\,, \tag{8}$$

and $\kappa = (\zeta_i^2 + \zeta_j^2)/(\zeta_i^2 - \zeta_j^2)$.

In the FMM, the simulation box is recursively divided into smaller cells, generating a tree structure [17]. The root of the tree is at level 0, and it corresponds to the entire simulation box. A parent cell at level l is decomposed into $2 \times 2 \times 2$ children cells of equal volume at level $l+1$. The recursive decomposition stops at

the leaf level, $l = L$, so that there are $2^L \times 2^L \times 2^L$ leaf cells. The short-range Coulomb potential energy is the contributions from atomic pairs (i, j) within the nearest neighbor leaf cells,

$$V_s = \sum_i \chi_i q_i + \frac{1}{2} \sum_i J_i q_i^2 + \sum_{(i,j)\in NN} \int d^3x_1 \int d^3x_2 \frac{\rho_i(\vec{x}_1; q_i)\rho_j(\vec{x}_2; q_j)}{|\vec{x}_1 - \vec{x}_2|}, \tag{9}$$

where NN denotes the set of atomic pairs which reside within the nearest-neighbor cells from each other. For the cth cell, the nearest-neighbor cells are the cell c itself and the 26 other cells which share a corner with it. The long-range potential energy is given by

$$V_l = \sum_{(i,j)\notin NN} \frac{q_i q_j}{|\vec{x}_i - \vec{x}_j|}, \tag{10}$$

where we have assumed that ζ_i^{-1} are much shorter than the edges of the leaf cell. Assuming that the number of cells is proportional to the number of atoms N, the entire Coulomb interaction is computed in $O(N)$ operations by recursively computing the truncated multipoles and the Taylor expansion of the potential on the hierarchy of cells [17].

3. Parallel implementation

To implement the MPCG-algorithm on parallel computers, we use a divide-and-conquer strategy based on domain decomposition [15,29]. The total volume Ω of the system is divided into p subsystems of equal volume, and each subsystem is assigned to a node in an array of p processors[3]. The data associated with the atoms in a subsystem are assigned to the corresponding node. Message passing is used to exchange the necessary information on distributed-memory computers.

The parallel implementation uses the FMM-based preconditioner since it is asymptotically faster than the $O(N^{3/2})$ Ewald method [16]. The most time-consuming part of the FMM–MPCG algorithm is the computation of the electrostatic interaction. The computational steps in this algorithm are listed below together with the analysis of the computation and interprocessor communication involved in each step. Below, steps 1 to 5 are for calculating V_l (far-field computation), while step 6 is for calculating V_s (near-field computation). The computation time per outer MPCG iteration involves only one evaluation of the far-field, while the near-field is evaluated $N_{\text{inner}} + 1$ times (once in the outer interaction and N_{inner} times in the inner loop, see Appendix A).

Step 1 Compute multipoles $\Phi_{L,c}$ for all the cells c at the leaf level L of the tree. This computation is performed locally in each processor, and its time complexity is $\tau_1^{\text{comp}} = c_1 N/p$, where c_1 is a constant[4].

Step 2 (upward pass) For tree levels $l = L-1, L-2, \ldots, 0$, compute $\Phi_{l,c}$ for all the cells c by summing the multipoles of their 8 children cells,

$$\Phi_{l,c} = \sum_{c' \in \{children(c)\}} T_1(\Phi_{l+1,c'}), \tag{11}$$

where the multipole-to-multipole (M2M) translation operator T_1 shifts the origin of the multipole representation, and $\{children(c)\}$ is the set of 8 children cells of parent c (see Fig. 1).

For $l \geq \log_8 p$, each processor computes the multipoles for $8^l/p$ cells. For $l < \log_8 p$, each processor is assigned only one cell to compute. Therefore,

[3] We assume that the processors are logically arranged as a three-dimensional cube of size $p^{1/3} \times p^{1/3} \times p^{1/3}$, and that p is a multiple of 8.

[4] The constants in the estimated costs take account of the computation for both $\tilde{q}$ and $\hat{q}$.

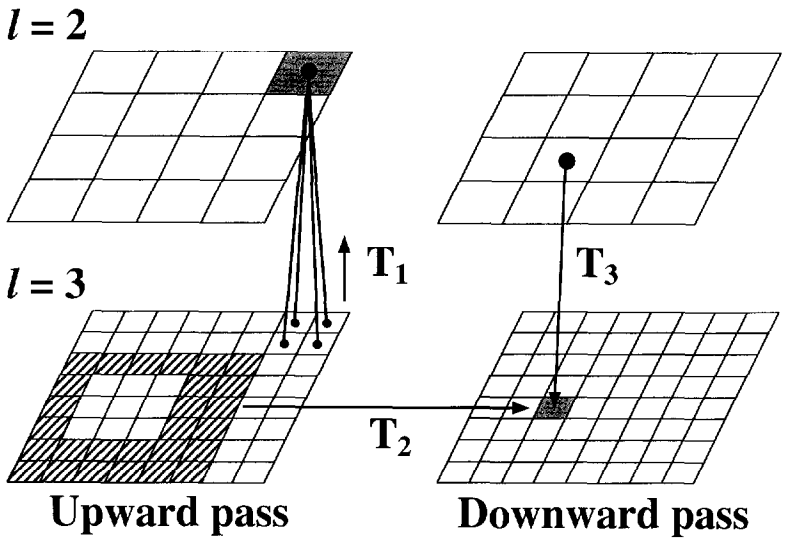

Fig. 1. Schematic of the far-field computation in a two-dimensional system. In the left column, the multipoles of a parent cell (shown in gray) at level 2 are obtained by shifting the multipoles of its 8 children cells at level 3 by the M2M translation operator T_1 and summing them. In the right column, the local Taylor expansion coefficients of a child cell (gray) at level 3 are computed from two contributions. First the local expansions of its parent at level 2 are inherited; the L2L translation operator, T_3, is used to shift the origin of the local expansion. The M2L translation operator, T_2, is then used to compute the contributions due to the interactive cells (shaded) at the same level.

$$\tau_2^{\mathrm{comp}} = 8c_2\left(\sum_{l=\log_8 p}^{L-1}\frac{8^l}{p} + \sum_{l=0}^{\log_8 p-1} 1\right) \sim 8c_2\left(\frac{1}{7\xi^3}\frac{N}{p} + \log_8 p\right), \tag{12}$$

where $c_2 = c_{\mathrm{MM}}$ is the computation time associated with each M2M shift operation. The approximation in Eq. (12) holds for the number of leaf cells $N_c = 8^L \gg p \gg 1$, assuming uniform atomic density. The leaf-cell length in unit of $(\Omega/N)^{1/3}$ is denoted by ξ.

For $l \geq \log_8 p$, the M2M operations are performed locally in each processor. For $l < \log_8 p$, the multipoles of only one child cell are locally available out of 8. The rest must be copied from other processors with the communication cost $\tau_2^{\mathrm{comm}} = 7d_m \log_8 p$ (d_m is the time required for copying the multipoles of one cell).

Step 3 If periodic boundary conditions are used, summation over infinitely repeated image boxes is carried out after step 2 [21,29]. The multipoles $\Phi_{0,0}$ of the total simulation box is used to compute the local Taylor expansion $\Psi_{0,0}$ of the potential due to the $\infty - 27$ well-separated images. This step involves a constant cost $\tau_3^{\mathrm{comp}} = c_3$ (the computation is duplicated in all the processors).

Step 4 (downward pass) For $l = 1, 2, \ldots, L$, compute the local Taylor expansion $\Psi_{l,c}$ for all the cells c (see Fig. 1),

$$\Psi_{l,c} = T_3(\Psi_{l-1,parent(c)}) + \sum_{c' \in \{interactive(c)\}} T_2(\Phi_{l,c'}) . \tag{13}$$

The first term in Eq. (13) contains the contributions from the parent's well-separated cells. This is inherited from the parent, $parent(c)$, of the cth cell by shifting the origin of the Taylor expansion using the local-to-local (L2L) translation operator T_3. The second term is due to the atoms in the interactive cells, which are the children of the parent's nearest-neighbors but are well-separated from the cth cell. The set of all the interactive cells of the cth cell is denoted by $\{interactive(c)\}$. The multipole-to-local (M2L) translation

operator T_2 converts the multipoles of an interactive cell to local Taylor expansion coefficients centered at the cth cell.

Since each cell has $6^3 - 3^3 = 189$ interactive cells, the computation time per cell is $c_4 = c_{\mathrm{LL}} + 189c_{\mathrm{ML}}$, where c_{LL} and c_{ML} denote the computation costs associated with the L2L and M2L operators, respectively. For $l \le \log_8 p$, each processor computes the local expansions for one cell. For $l > \log_8 p$, each processor is assigned the $8^l/p$ local cells,

$$\tau_4^{\mathrm{comp}} = c_4\left(\sum_{l=\log_8 p+1}^{l} \frac{8^l}{p} + \sum_{l=1}^{\log_8 p} 1\right) \sim c_4\left(\frac{8}{7\xi^3}\frac{N}{p} + \log_8 p\right), \tag{14}$$

For $l \le \log_8 p$, the multipoles of up to 189 interactive cells must be copied from other processors. For $l > \log_8 p$, each subsystem must be augmented by copying two boundary layers of cells from other processors. The associated communication cost is

$$\tau_4^{\mathrm{comm}} = d_m\left[\sum_{l=\log_8 p+1}^{L}\left\{\left(\frac{2^l}{p^{1/3}}+4\right)^3 - \frac{8^l}{p}\right\} + \sum_{l=1}^{\log_8 p} 189\right] \sim d_m\left(\frac{16}{\xi^2}\left(\frac{N}{p}\right)^{2/3} + 189\log_8 p\right). \tag{15}$$

Step 5 The far-field contribution is evaluated at each atom's position using the local Taylor expansion $\Psi_{L,c(i)}$ at the leaf level, where $c(i)$ denotes the leaf cell to which the ith atom belongs. This computation is performed locally with the cost $\tau_5^{\mathrm{comp}} = c_5 N/p$.

Step 6 (near field) Contributions to the electrostatic potential from all the atomic pairs within the 27 nearest-neighbor cells are evaluated directly without using multipoles. Using the Newton's third law, $\tau_6^{\mathrm{comp}} = [27(N_{\mathrm{inner}}+1)c_6\xi^3/2](N/p)$. Note that the near-field is evaluated $N_{\mathrm{inner}}+1$ times per outer MPCG iteration.

At step 6, each subsystem is augmented with the atoms in one boundary layer of cells with the cost

$$\tau_6^{\mathrm{comm}} = \frac{d_a N(N_{\mathrm{inner}}+1)}{N_c}\left\{\left(\frac{2^L}{p^{1/3}}+2\right)^3 - \frac{8^L}{p}\right\} \sim 6(N_{\mathrm{inner}}+1)d_a\xi\left(\frac{N}{p}\right)^{2/3}, \tag{16}$$

where d_a is the communication cost per copied atom.

The total execution time is the sum of the above contributions, $\tau(N,p) = \tau^{\mathrm{comp}}(N,p) + \tau^{\mathrm{comm}}(N,p)$, where $\tau^{\mathrm{comp}}(N,p) = \tau_1^{\mathrm{comp}} + \tau_2^{\mathrm{comp}} + \cdots + \tau_6^{\mathrm{comp}}$ and $\tau^{\mathrm{comm}}(N,p) = \tau_2^{\mathrm{comm}} + \tau_4^{\mathrm{comm}}$. Using the memory-bounded scaling [30], the parallel efficiency of the program is defined as $E = \tau^{\mathrm{comp}}(N/p,1)/\tau(N,p)$. According to the above analysis, the parallel efficiency of the MPCG program is estimated to be

$$E^{-1} = 1 + \frac{(8c_2 + c_4 + 196d_m)p\log_8 p/N + \{16d_m/\xi^2 + 6(N_{\mathrm{inner}}+1)d_a\xi\}(p/N)^{1/3}}{c_1 + c_5 + 8(c_2+c_4)/7\xi^3 + 27c_6(N_{\mathrm{inner}}+1)\xi^3/2}. \tag{17}$$

In deriving Eq. (17), we have omitted the term proportional to c_3, which was found small for the systems we have tested. For a large number of processors p, the term proportional to $p\log_8 p/N$ in Eq. (17) degrades the parallel efficiency significantly. With the preconditioning using a large N_{inner}, this logarithmic p dependence is hidden by the term proportional to $(p/N)^{1/3}$. In fact, $E^{-1} = 1 + (12d_a/27c_6\xi^2)(p/N)^{1/3}$ for $N_{\mathrm{inner}} \to \infty$, and this efficiency can be improved by increasing the granularity N/p.

4. Numerical experiment

We first compare the performance of the MPCG scheme with that of the conventional CG method. The physical system is a 4 800 atom α-Al_2O_3 crystal with periodic boundary conditions. The program is run on a

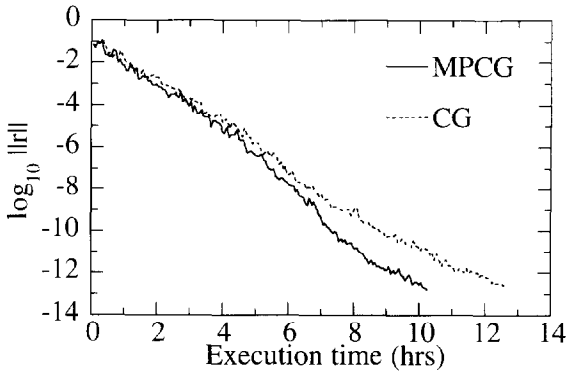

Fig. 2. The residual (in atomic unit) in the conjugate gradient calculation as a function of the execution time. The results for the MPCG method and the conventional CG method without preconditioning are shown with the solid and dashed curves, respectively.

single processor of Digital Alpha 2100/275 with a clock cycle of 275 MHz. We choose $\gamma = 0.9$ Å, and the real-space sum in Eq. (6) is truncated at $x_{ij} = 6$ Å. The number of plane waves $\vec{k}$ in Eq. (7) is 7 152.

Fig. 2 shows the residual of the linear system associated with $\tilde{\boldsymbol{q}}$ as a function of the execution time. (The residual for the $\hat{\boldsymbol{q}}$ vector behaves similarly.) The solid and dashed curves represent the results of the MPCG and CG methods, respectively. The Ewald scheme is used to split the Coulomb interaction matrix. We choose the number of inner iterations $N_{\text{inner}} = 8$ per outer iteration. For this system, the preconditioner reduces the execution time to achieve the same convergence level (10^{-12} a.u.) about 20%.

Next we test the parallel performance of the MPCG program on the IBM SP2 computer at the Maui High Performance Computing Center. The program is written in a message passing programming style using the Message Passing Interface (MPI) standard [31,32]. The FMM is used to partition the Coulomb matrix. The size of the leaf cell is 6 Å, and the number of nonzero rows of the preconditioning matrix is $\sim$ 50. The multipole expansion is truncated at the quadrupole level, and we use $N_{\text{inner}} = 8$. We measure a scaled speedup such that the number of atoms is proportional to the number of processors, $N = 414\,720p$. The largest system contains 26 542 080 atoms on 64 processors.

Fig. 3 shows the total execution time (solid line) and the time spent for communication (dashed line) as a function of the number of atoms. Both times increase only slightly when the number of processors increases from 1 to 64, exhibiting a good scalability.

Fig. 4 shows the parallel efficiency (solid lines) and communication overhead (dashed lines) as a function of the number of atoms. The results with and without preconditioning are denoted by circles and squares, respectively. For the largest (26.5 million-atom) system, the preconditioning improves the parallel efficiency from 0.92 to 0.95. The communication overhead of the MPCG scheme is 5% of the total execution time for the largest system. The multilevel preconditioning scheme enhances the locality of computation by extensively using the short-range interaction matrix $\boldsymbol{M}_s$, and consequently the program runs efficiently on parallel platforms.

5. Discussion

We have also embedded the MPCG algorithm in our parallel MD programs. In fact, the new preconditioning scheme was motivated by the multiple time scale (MTS) method [33–36] used in these MD programs. In the MTS approach, the Liouville operator for an N-atom system is decomposed into two parts: $iL = iL_s + iL_l$, where

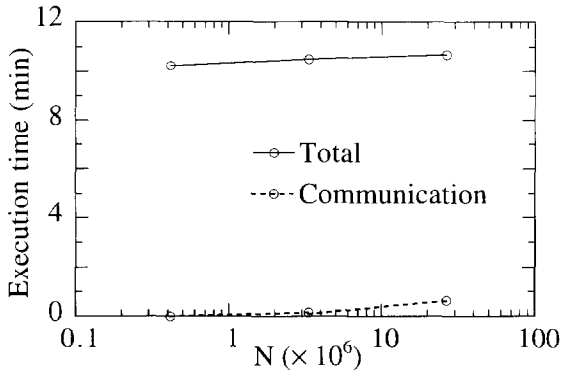

Fig. 3.

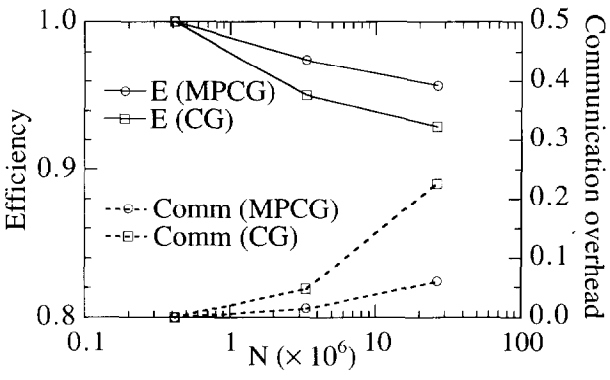

Fig. 4.

Fig. 3. Execution time of the parallel MPCG program (solid line) increases only slightly as the number of atoms increases. Each processor contains 414720 atoms, and the largest system is 26542080 atoms on 64 processors. The communication overhead (dashed line) of the same program is also shown.

Fig. 4. Memory-bounded parallel efficiency E of the MPCG program (solid lines) as a function of the number of atoms. Open circles and open squares are the results for the MPCG and CG methods, respectively. Communication overheads of the same program are shown by the dashed lines.

$$iL_s = \sum_{j=1}^{N} \left(\dot{\boldsymbol{x}}_j \cdot \frac{\partial}{\partial \vec{x}_j} - \frac{\partial (V_0 + V_s)}{\partial \vec{x}_j} \cdot \frac{\partial}{\partial \vec{p}_j} \right) = iK + iP \,, \tag{18}$$

$$iL_l = -\sum_{j=1}^{N} \frac{\partial V_l}{\partial \vec{x}_j} \cdot \frac{\partial}{\partial \vec{p}_j} \,. \tag{19}$$

The Trotter factorization gives the temporal propagator for the positions and momenta of the system, $\Gamma = \{x_j, p_j\}$ [34],

$$\Gamma(\Delta t) = \exp(iL_l \Delta t/2) \exp(iL_s \Delta t) \exp(iL_l \Delta t/2) \Gamma(0) \,, \tag{20}$$

where the short-range propagator characterized by shorter time scales is further factored into

$$\exp(iL_s \Delta t) = [\exp(iP\Delta t/2N_{\text{inner\_md}}) \exp(iK\Delta t/N_{\text{inner\_md}}) \exp(iP\Delta t/2N_{\text{inner\_md}})]^{N_{\text{inner\_md}}} \,. \tag{21}$$

The MTS algorithm based on this decomposition is shown in Appendix B, where the diagonal mass matrix is $\Xi_{ij} = m_i \delta_{ij}$. The MTS algorithm has a doubly-nested loop structure which is similar to the one in the MPCG algorithm. Due to this structural similarity, the interatomic forces for the MD and the electrostatic potential for the electronegativity equalization can be computed in a single loop over atomic pairs. This greatly amortizes the cost of the MPCG scheme in MD simulations.

In summary, we have developed a multilevel preconditioned conjugate-gradient method to speed up the electrostatic-energy minimization in variable-charge MD simulations. The scheme splits the Coulomb-interaction matrix into short- and long-range components, and uses the computationally less intensive short-range matrix as a preconditioner. The new scheme speeds up the solution because of three reasons:

- increased convergence rate of the preconditioned linear system;
- increased parallel efficiency by the enhanced data locality;
- amortized computational cost due to the algorithmic similarity to the multiple-time-scale MD.

Acknowledgements

This work was supported by Army Research Office, Grant No. DAAH04-96-1-0393, Louisiana Education Quality Support Fund (LEQSF), Grant No. LEQSF(96-99)-RD-A-10 and DoD/LEQSF(96-99)-3, National Science Foundation CAREER Program, Grant No. ASC-9701504, and Petroleum Research Fund, Grant No. 31659-AC9. Numerical experiments were performed on the IBM SP2 computer at Maui High Performance Computing Center (MHPCC). The computations were also performed on parallel architectures at the Concurrent Computing Laboratory for Materials Simulations (CCLMS) at Louisiana State University. The facilities in the CCLMS were acquired with the Equipment Enhancement Grants from the LEQSF. The author wishes to thank Drs. Priya Vashishta, Rajiv K. Kalia, Kenji Tsuruta, and Shuji Ogata for fruitful discussions.

Appendix A. Multilevel preconditioned conjugate gradient algorithm

A.1. Outer loop

Compute the residual $r^0 \leftarrow -\chi - M \cdot q^0$ for an initial guess q^0 for charges
for $i \leftarrow 1$ to N_{outer}
  Solve $M_s \cdot z^{i-1} = r^{i-1}$ //Inner loop to obtain the preconditioning vector z^{i-1}
  if $i = 1$
    $p^1 \leftarrow z^0$
  else
    $p^i \leftarrow z^{i-1} + \frac{r^{i-1} \cdot z^{i-1}}{r^{i-2} \cdot z^{i-2}} p^{i-1}$
  endif
  $q^i \leftarrow q^{i-1} + \frac{r^{i-1} \cdot z^{i-1}}{p^i \cdot M \cdot p^i} p^i$
  $r^i \leftarrow r^{i-1} - \frac{r^{i-1} \cdot z^{i-1}}{p^i \cdot M \cdot p^i} M \cdot p^i$
  if not convergent, continue
endfor

A.2. Inner loop

Compute the residual $g^0 \leftarrow r - M_s \cdot z^0$ for an initial guess z^0
for $j = 1$ to N_{inner}
  if $j = 1$
    $h^1 \leftarrow g^0$
  else
    $h^j \leftarrow g^{j-1} + \frac{g^{j-1} \cdot g^{j-1}}{g^{j-2} \cdot g^{j-2}} h^{j-1}$
  endif
  $z^j \leftarrow z^{j-1} + \frac{g^{j-1} \cdot g^{j-1}}{h^j \cdot M_s \cdot h^j} h^j$
  $g^j \leftarrow g^{j-1} - \frac{g^{j-1} \cdot g^{j-1}}{h^j \cdot M_s \cdot h^j} M_s \cdot h^j$
  if not convergent, continue
endfor

Appendix B. Multiple-time-scale molecular dynamics algorithm

B.1. Outer loop

$g_l \leftarrow -\nabla V_l$ //Initial long-range force
for $m = 1$ to N_{outer_md}
  $v \leftarrow v + \frac{\Delta t}{2} \Xi^{-1} g_l$ //Velocity update due to the long-range force
  $\Gamma \leftarrow \exp(-L_s \Delta t)\Gamma$ //Inner loop
  $g_l \leftarrow -\nabla V_l$ //Long-range force calculation
  $v \leftarrow v + \frac{\Delta t}{2} \Xi^{-1} g_l$ //Velocity update due to the long-range force
endfor

B.2. Inner loop

$g_s \leftarrow -\nabla V_s$ //Initial short-range force
for $n = 1$ to $N_{\text{inner\_md}}$
  $v \leftarrow v + \frac{\Delta t}{2N_{\text{inner\_md}}} \Xi^{-1} g_s$ //Velocity update due to the short-range force
  $g_s \leftarrow -\nabla V_s$ //Short-range force calculation
  $x \leftarrow x + \frac{\Delta t}{N_{\text{inner\_md}}} v + \frac{1}{2}\left(\frac{\Delta t}{N_{\text{inner\_md}}}\right)^2 \Xi^{-1} g_s$ //Position update due to the short-range force
  $v \leftarrow v + \frac{\Delta t}{2N_{\text{inner\_md}}} \Xi^{-1} g_s$ //Velocity update due to the short-range force
endfor

References

[1] F.F. Abraham, D. Brodbeck, R.A. Rafey and W.E. Rudge, Phys. Rev. Lett. 73 (1994) 272;
F.F. Abraham, Phys. Rev. Lett. 77 (1996) 869.
[2] S.J. Zhou, P.S. Lomdahl, R. Thomson and B.L. Holian, Phys. Rev. Lett. 76 (1996) 2318;
S.J. Zhou, D.M. Beazley, P.S. Lomdahl and B.L. Holian, Phys. Rev. Lett. 78 (1997) 479.
[3] A. Nakano, R.K. Kalia and P. Vashishta, Phys. Rev. Lett. 73 (1994) 2336; 75 (1995) 3138;
R.K. Kalia, A. Nakano, K. Tsuruta and P. Vashishta, Phys. Rev. Lett. 78 (1997) 689;
R.K. Kalia, A. Nakano, A. Omeltchenko, K. Tsuruta and P. Vashishta, Phys. Rev. Lett. 78 (1997) 2144;
A. Omeltchenko, J. Yu, R.K. Kalia and P. Vashishta, Phys. Rev. Lett. 78 (1997) 2148.
[4] C.L. Briant, Metal. Trans. A 21 (1990) 2339.
[5] J.W. Storer, D.J. Giesen, C.J. Cramer and D.G. Truhlar, J. Computer-aided Mol. Design 9 (1995) 87.
[6] The quantum-mechanical/molecular-mechanical (QMMM) approach embeds a quantum mechanical description of charge transfer and chemical reaction in a classical description of environmental atoms: M.J. Field, P.A. Bash and M. Karplus, J. Comput. Chem. 11 (1990) 700;
C.S. Carmer, B. Weiner and M. Frenklach, J. Chem. Phys. 99 (1993) 1356;
F. Maseras and K. Morokuma, J. Comput. Chem. 16 (1995) 1170.
The QMMM method has been included in software packages such as GAMESS: W. Schmidt et al., J. Comput. Chem. 14 (1993) 1347.
[7] A. Alavi, L.J. Alvarez, S.R. Elliott and I.R. McDonald, Phil. Mag. B 65 (1992) 489.
[8] A.K. Rappé and W.A. Goddard, J. Phys. Chem. 95 (1991) 3358.
[9] S.W. Rick, S.J. Stuart and B.J. Berne, J. Chem. Phys. 101 (1994) 6141.
[10] F.H. Streitz and J.W. Mintmire, Phys. Rev. B 50 (1994) 11996; Langmuir 12 (1996) 4605.
[11] G.H. Golub and C.F. van Loan, Matrix Computations, 2nd Ed. (Johns Hopkins Univ. Press, Baltimore, 1989).
[12] R. Barrett et al., Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods (SIAM, Philadelphia, 1994).
[13] Y. Saad, Iterative Methods for Sparse Linear Systems (PWS, Boston, 1996).
[14] S.W. de Leeuw, J.W. Perram and E.R. Smith, Proc. R. Soc. London A 373 (1980) 27.
[15] R.K. Kalia, S.W. de Leeuw, A. Nakano and P. Vashishta, Comput. Phys. Commun. 74 (1993) 316.
[16] A.Y. Toukmaji and J.A. Board, Comput. Phys. Commun. 95 (1996) 73.
[17] L. Greengard and V. Rokhlin, J. Comput. Phys. 73 (1987) 325.

[18] L. Greengard and W.D. Gropp, Comput. Math. Appl. 20 (1990) 63.
[19] K.E. Schmidt and M.A. Lee, J. Stat. Phys. 63 (1991) 1223.
[20] F. Zhao and S. Lennart Johnsson, SIAM J. Sci. Stat. Comput. 12 (1991) 1420;
Y. Hu and S. Lennart Johnsson, Sci. Prog. 5 (1996) 337.
[21] H.-Q. Ding, N. Karasawa and W.A. Goddard, Chem. Phys. Lett. 196 (1992) 6.
[22] K. Nabors, F.T. Korsmeyer, F.T. Leighton and J. White, SIAM J. Sci. Comput. 15 (1994) 713.
[23] H.G. Petersen, D. Soelvason, J.W. Perram and E.R. Smith, J. Chem. Phys. 101 (1994) 8870.
[24] A.Y. Grama, V. Kumar and A. Sameh, in: Supercomputing '94 (IEEE Comp. Soc., Washington, D.C., 1994).
[25] J.P. Singh, C. Holt, T. Totsuka, A. Gupta and J. Hennessy, J. Par. Dist. Comput. 27 (1995) 118.
[26] M.S. Warren and J.K. Salmon, Comput. Phys. Commun. 87 (1995) 266.
[27] C.A. White and M. Head-Gordon, J. Chem. Phys. 105 (1996) 5061.
[28] E.L. Pollock and J. Glosli, Comput. Phys. Commun. 95 (1996) 93.
[29] A. Nakano, R.K. Kalia and P. Vashishta, Comput. Phys. Commun. 83 (1994) 197.
[30] X.-H. Sun and J. Gustafson, Par. Comput. 17 (1991) 1093.
[31] W. Gropp, E. Lusk and A. Skjellum, Using MPI (MIT Press, Cambridge, 1994).
[32] M. Snir, S. Otto, S. Huss-Lederman, D. Walker and J. Dongarra, MPI: The Complete Reference (MIT Press, Cambridge, 1996).
[33] W.B. Streett, D.J. Tildesley and G. Saville, Mol. Phys. 35 (1978) 639.
[34] M.E. Tuckeman, B.J. Berne and G.J. Martyna, J. Chem. Phys. 94 (1991) 6811.
[35] H. Grubmuller, H. Heller, A. Windemuth and K. Schulten, Mol. Sim. 6 (1991) 121.
[36] J.J. Biesiadecki and R.D. Skeel, J. Comput. Phys. 109 (1993) 318.