Available online at www.sciencedirect.com
ScienceDirect
Computer Physics Communications 178 (2008) 73–87
Computer Physics Communications
www.elsevier.com/locate/cpc

A scalable parallel algorithm for large-scale reactive force-field molecular dynamics simulations

Ken-ichi Nomura, Rajiv K. Kalia, Aiichiro Nakano *, Priya Vashishta

Collaboratory for Advanced Computing and Simulations, Department of Computer Science, Department of Physics & Astronomy, Department of Chemical Engineering & Materials Science, University of Southern California, Los Angeles, CA 90089-0242, USA

Received 18 June 2007; received in revised form 16 August 2007; accepted 17 August 2007

Available online 15 September 2007

Abstract

A scalable parallel algorithm has been designed to perform multimillion-atom molecular dynamics (MD) simulations, in which first principles-based reactive force fields (ReaxFF) describe chemical reactions. Environment-dependent bond orders associated with atomic pairs and their derivatives are reused extensively with the aid of linked-list cells to minimize the computation associated with atomic n-tuple interactions ($n \leqslant 4$ explicitly and $\leqslant 6$ due to chain-rule differentiation). These n-tuple computations are made modular, so that they can be reconfigured effectively with a multiple time-step integrator to further reduce the computation time. Atomic charges are updated dynamically with an electronegativity equalization method, by iteratively minimizing the electrostatic energy with the charge-neutrality constraint. The ReaxFF-MD simulation algorithm has been implemented on parallel computers based on a spatial decomposition scheme combined with distributed n-tuple data structures. The measured parallel efficiency of the parallel ReaxFF-MD algorithm is 0.998 on 131,072 IBM BlueGene/L processors for a 1.01 billion-atom RDX system.

PACS: 02.70.-c; 02.70.Ns; 82.20.Db

Keywords: Molecular dynamics; Reactive force field; Parallel computing

1. Introduction

Molecular dynamics (MD) is an atomistic simulation method for studying a wide class of materials, such as metals, ceramics, and biomolecules under ambient as well as extreme conditions. Continuous efforts are being made to incorporate chemical reactions into MD simulations, so that broad material processes such as catalysis, corrosion and detonation can be studied. Currently, a number of software packages are available for first-principles MD simulations based on quantum mechanical (QM) calculations to describe reactive events such as bond breaking and formation [1]. One of the longstanding problems in materials science is mechanically induced chemical reaction such as stress corrosion cracking and shock-induced detonation [2,3], which may drastically change material properties. A study of chemical reactions induced by mechanical loading is a challenging task, because they are often inseparable from geometrical heterogeneity involving millions of atoms. Their understanding thus requires the combination of vast spatial extent and accurate chemistry. Because of the intensive computational requirement and poor scalability, however, the system size in QM simulations is usually limited to $\sim 10^2$ atoms. One approach to tackle this problem is the hybridization of different methodologies, such as QM/molecular mechanics (MM) schemes [4–8]. However, they require adaptive selection of reactive sites and sophisticated treatment of QM/MM boundaries, which are nontrivial for complex mechanochemical processes.

An alternative approach is to fit essential chemical reactions by semi-empirical force fields [9–11], in which parameterized interatomic potentials are "trained" to reproduce datasets obtained with accurate QM calculations. Recent advances in reactive force

* Corresponding author.
E-mail address: anakano@usc.edu (A. Nakano).

0010-4655/$ – see front matter
doi:10.1016/j.cpc.2007.08.014

fields (ReaxFF) [12–14] have opened up a possibility to study reactive processes in a wide range of materials such as hydrocarbons [12], silica [13], alumina [15], nitramines [14], catalysts [16], and polymers [17]. We have recently developed a scalable implementation of ReaxFF MD on massively parallel computers, which provides us with the requisite coupling of quantum-mechanically accurate description of chemical reactions and large length-scale mechanical processes to study mechanochemical processes under extreme conditions. For example, our parallel ReaxFF algorithm has enabled million-to-billion atom MD simulations on a large number ($\sim 10^5$) of processors to study shock-induced initiation of energetic materials, with focuses on the effects of microstructures (e.g., voids and grain boundaries) and nanofluidics (e.g., nanojets) on chemical reactions [18,19]. In this paper, we present the design and implementation of our parallel ReaxFF algorithm along with its scalability tests on various parallel computers. Section 2 describes the ReaxFF MD algorithm, and its parallel implementation is discussed in Section 3. Results of benchmark tests are presented in Section 4, and Section 5 contains summary.

2. Reactive force field molecular dynamics algorithm

In the reactive force field (ReaxFF), interatomic interactions between N atoms comprise valence interactions described with bond order (BO) as well as noncovalent interactions [16]:

$$E_{\mathrm{ReaxFF}}\big(\{\mathbf{r}_{ij}\},\{\mathbf{r}_{ijk}\},\{\mathbf{r}_{ijkl}\},\{q_i\},\{BO_{ij}\}\big) = E_{\mathrm{bond}} + E_{\mathrm{lp}} + E_{\mathrm{over}} + E_{\mathrm{under}} + E_{\mathrm{val}} + E_{\mathrm{pen}} + E_{\mathrm{coa}} + E_{\mathrm{tors}} + E_{\mathrm{conj}} + E_{\mathrm{hbond}} + E_{\mathrm{vdWaals}} + E_{\mathrm{Coulomb}}, \tag{1}$$

in which the valence interactions include the bonding energy E_{bond}, lone-pair energy E_{lp}, overcoordination energy E_{over}, undercoordination energy E_{under}, valence-angle energy E_{val}, penalty energy E_{pen}, 3-body conjugation energy E_{coa}, torsion-angle energy E_{tors}, 4-body conjugation energy E_{conj}, and hydrogen bonding energy E_{hbond}. In Eq. (1), the total energy is a function of relative positions of atomic pairs, $\mathbf{r}_{ij}$, triplets, $\mathbf{r}_{ijk}$, and quadruplets, $\mathbf{r}_{ijkl}$, as well as atomic charges q_i and bond orders BO_{ij} between atomic pairs. The noncovalent interactions comprise van der Waals energy E_{vdWaals} and Coulomb energy E_{Coulomb}, which are screened by a taper function [12]. ReaxFF also incorporates a charge transfer mechanism to describe realistic chemical reaction pathways. The environment-dependent charge distribution is described using the electronegativity equalization method (EEM) [20], in which individual atomic charges vary in time. The analytical formulas for these energy terms are given in the following subsections.

2.1. Bond-order calculation

An essential building block of ReaxFF is the bond order that describes the nature of covalent bonds in terms of three exponential functions corresponding to σ, π and double-π bonds. The bond order BO_{ij} is an attribute of atomic pair (i, j), where $i, j = 1, \ldots, N$ with N being the number of atoms. The BO calculation in ReaxFF involves two steps. The first step computes raw BOs, BO'_{ij}, that are simply functions of the interatomic distances of all pairs of atoms within single covalent-bond cutoff distance r_{cb} (typically 3–4 Å). In the second step, each raw BO is corrected as a function of the valencies, Δ'_i and Δ'_j, of the atoms comprising the pair to obtain the full bond order BO_{ij}. Many-body effects are implicitly included through the valency terms.

We first compute the raw bond order BO'_{ij} as a sum of σ, π and double-π bond contributions,

$$BO'_{ij} = BO'^{\sigma}_{ij} + BO'^{\pi}_{ij} + BO'^{\pi\pi}_{ij} = \exp\left[p_{\mathrm{bo1}}\left(\frac{r_{ij}}{r^{\sigma}_{o}}\right)^{p_{\mathrm{bo2}}}\right] + \exp\left[p_{\mathrm{bo3}}\left(\frac{r_{ij}}{r^{\pi}_{o}}\right)^{p_{\mathrm{bo4}}}\right] + \exp\left[p_{\mathrm{bo5}}\left(\frac{r_{ij}}{r^{\pi\pi}_{o}}\right)^{p_{\mathrm{bo6}}}\right], \tag{2}$$

where $r_{ij} = |\mathbf{r}_{ij}| = |\mathbf{r}_i - \mathbf{r}_j|$ is the interatomic distance between ith and jth atoms ($\mathbf{r}_i$ is the position of the ith atom). These raw bond orders are subsequently corrected to produce correct atomic valences as

$$BO_{ij} = BO^{\sigma}_{ij} + BO^{\pi}_{ij} + BO^{\pi\pi}_{ij}, \tag{3a}$$

$$BO^{\sigma}_{ij} = BO'^{\sigma}_{ij} \cdot f_1\big(\Delta'_i, \Delta'_j\big) \cdot f_4\big(\Delta'_i, BO'_{ij}\big) \cdot f_5\big(\Delta'_j, BO'_{ij}\big), \tag{3b}$$

$$BO^{\pi}_{ij} = BO'^{\pi}_{ij} \cdot f_1\big(\Delta'_i, \Delta'_j\big) \cdot f_1\big(\Delta'_i, \Delta'_j\big) \cdot f_4\big(\Delta'_i, BO'_{ij}\big) \cdot f_5\big(\Delta'_j, BO'_{ij}\big), \tag{3c}$$

$$BO^{\pi\pi}_{ij} = BO'^{\pi\pi}_{ij} \cdot f_1\big(\Delta'_i, \Delta'_j\big) \cdot f_1\big(\Delta'_i, \Delta'_j\big) \cdot f_4\big(\Delta'_i, BO'_{ij}\big) \cdot f_5\big(\Delta'_j, BO'_{ij}\big), \tag{3d}$$

$$\Delta'_i = -Val_i + \sum_{j \in n(i)} BO'_{ij}, \tag{3e}$$

$$f_1(\Delta_i, \Delta_j) = \frac{1}{2}\left(\frac{Val_i + f_2(\Delta'_i, \Delta'_j)}{Val_i + f_2(\Delta'_i, \Delta'_j) + f_3(\Delta'_i, \Delta'_j)} + \frac{Val_j + f_2(\Delta'_i, \Delta'_j)}{Val_j + f_2(\Delta'_i, \Delta'_j) + f_3(\Delta'_i, \Delta'_j)}\right), \tag{3f}$$

$$f_2(\Delta'_i, \Delta'_j) = \exp(-p_{\mathrm{boc1}}\Delta'_i) + \exp(-p_{\mathrm{boc1}}\Delta'_j), \tag{3g}$$

$$f_3(\Delta'_i, \Delta'_j) = -\frac{1}{p_{\mathrm{boc2}}} \ln\left\{\frac{1}{2}\big[\exp(-p_{\mathrm{boc2}}\Delta'_i) + \exp(-p_{\mathrm{boc2}}\Delta'_j)\big]\right\}, \tag{3h}$$

Table 1
Bond-order calculation algorithm

Algorithm BO_calc
Input:
$\{\mathbf{r}_i | i = 1, \ldots, N\}$ = set of atomic positions, where N is the number of atoms
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of the ith atom
Output:
$\{BO_{ij}\}$ = set of full bond orders
$\{\Delta_i\}$ = set of full valencies
Variables:
$\{BO'_{ij}\}$ = set of raw bond orders
$\{\Delta'_i\}$ = set of raw valencies
Steps:
1 *compute raw bond orders* $\{BO'_{ij}\}$ *and valencies* $\{\Delta'_i\}$
  for each atom $\mathbf{r}_i$
    for each atom $\mathbf{r}_j \in n(i)$
      $BO'_{ij} \leftarrow BO'^{\sigma}_{ij}(r_{ij}) + BO'^{\pi}_{ij}(r_{ij}) + BO'^{\pi\pi}_{ij}(r_{ij})$ // see Eq. (2)
    $\Delta'_i \leftarrow -Val_i$ // Val_i is atomic valency in Eq. (3e)
  for each atom $\mathbf{r}_i$
    for each atom $\mathbf{r}_j \in n(i)$
      $\Delta'_i \leftarrow \Delta'_i + BO'_{ij}$ // Eq. (3e)
2 *compute full bond orders* $\{BO_{ij}\}$ *and valencies* $\{\Delta_i\}$
  for each atom $\mathbf{r}_i$
    for each atom $\mathbf{r}_j \in n(i)$
      $BO_{ij} \leftarrow BO'_{ij} \times f_1 \times f_4 \times f_5$ // f_1, f_4 and f_5 are BO correction functions in Eq. (3)
    $\Delta_i \leftarrow -Val_i$
  for each atom $\mathbf{r}_i$
    for each atom $\mathbf{r}_j \in n(i)$
      $\Delta_i \leftarrow \Delta_i + BO_{ij}$ // Eq. (4)

$$f_4(\Delta'_i, BO'_{ij}) = \frac{1}{1 + \exp(-p_{\mathrm{boc3}}(p_{\mathrm{boc4}} BO'_{ij} BO'_{ij} - \Delta'^{\mathrm{boc}}_i) + p_{\mathrm{boc5}})}, \tag{3i}$$

$$f_5(\Delta'_j, BO'_{ij}) = \frac{1}{1 + \exp(-p_{\mathrm{boc3}}(p_{\mathrm{boc4}} BO'_{ij} BO'_{ij} - \Delta'^{\mathrm{boc}}_j) + p_{\mathrm{boc5}})}, \tag{3j}$$

$$\Delta'^{\mathrm{boc}}_i = -Val^{\mathrm{boc}}_i + \sum_{j \in n(i)} BO'_{ij}, \tag{3k}$$

where $n(i)$ is the set of neighbor atoms that are within single covalent-bond cutoff distance r_{cb} from the ith atom, and the parameters in Eqs. (2) and (3) are listed in Table A.1 in Appendix A. After the bond orders BO_{ij} are computed for all pairs (i, j), the valency of the ith atom is computed as

$$\Delta_i = -Val_i + \sum_{j \in n(i)} BO_{ij}. \tag{4}$$

Table 1 shows the BO calculation algorithm, *BO_calc*. The algorithm also computes full valencies Δ_i as a function of the corrected bond orders BO_{ij} (Eq. (4)). To avoid repeated computation of BO derivative values in subsequent energy and force calculations, we compute the derivative of BOs with respect to the raw BO and valency here and keep them until the atomic coordinates are updated.

2.2. Energy and force calculations

The various interaction functions in Eq. (1) represent a variety of chemistries, which can be grouped according to 1) valence or noncovalent type, 2) number of explicitly involved atoms, and 3) degree of the neighbor-atom effect. Here, the number of explicitly involved atoms is the nesting level of summations that constitute each energy term (e.g., 3 for the triply nested summation in E_{val} in Section 2.2.3 below). The degree of the neighbor-atom effect is defined as the number of implicitly involved loops through valency terms in the energy functions. For example, the corrected valency term Δ_i is of second degree, since its evaluation includes doubly nested sums:

$$\sum_j BO_{ij}(BO'_{ij}, \Delta'_i, \Delta'_j) = \sum_j BO_{ij}\left(BO'_{ij}, \Delta'_i\left(\sum_m BO'_{im}\right), \Delta'_j\left(\sum_n BO'_{jn}\right)\right). \tag{5}$$

Table 2
Classification of interaction functions

Number of explicit atoms	Energy terms
1	$E_{\mathrm{lp}}(2)$, $E_{\mathrm{over}}(3)$, $E_{\mathrm{under}}(3)$
2	$E_{\mathrm{bond}}(1)$, $E_{\mathrm{vdWaals}}(0)$[a], $E_{\mathrm{Coulomb}}(0)$[a]
3	$E_{\mathrm{val}}(2)$, $E_{\mathrm{pen}}(2)$, $E_{\mathrm{coa}}(2)$, $E_{\mathrm{hbond}}(2)$
4	$E_{\mathrm{tors}}(2)$, $E_{\mathrm{conj}}(2)$

[a] Noncovalent interactions. The numbers in the parentheses are the greatest degrees of the neighbor atom effect.

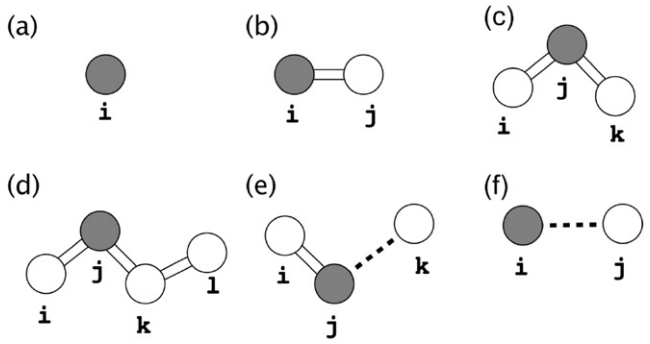

Fig. 1. Schematic of atomic configurations in energy terms: (a) 1-body, (b) 2-body, (c) 3-body, (d) 4-body, (e) hydrogen–bonding and (f) noncovalent interactions, respectively. A gray sphere represents the position of the primary atom in each energy term. Open bars represent covalent bonds, while dotted lines are noncovalent bonds.

In this subsection, we first explain the nature of interaction functions, which is essential to map them onto a parallel computer. We then present actual algorithms.

Table 2 classifies the energy terms according to the number of explicitly involved atoms. The noncovalent interactions are pairwise 2-body functions with two explicit atoms, while valence terms involve up to four explicit atoms. Each explicit atom implicitly involves effects from surrounding atoms through BO and valency terms. The implicit atoms require additional computations and incur increased cutoff distances. For example, in force calculation, the corrected valency term Δ_i causes a doubly nested loop over a neighbor list of the ith atom with the associated computational cost $\mathrm{O}(M^2)$, where M is the average number of covalent bonds per atom. The interaction cutoff length is accordingly increased to $2r_{\mathrm{cb}}$. The level of neighbor list is thus regarded as the degree of environment effect. The numbers in the parentheses in Table 2 shows the greatest degree of neighbor-atom effect among explicit atoms.

Fig. 1 shows the configurations of atoms involved in the computation of 1-body, 2-body, 3-body, 4-body, hydrogen–bonding and noncovalent energy functions, respectively. In the figure, a gray sphere represents the position of a primary atom scanned by the outermost loop of each energy-term calculation. The primary atoms play a pivotal role on our parallel implementation in Section 3.

Note that the energy terms in ReaxFF share common computational structures. In addition to algorithms to compute individual energy terms, we define two algorithms, *force_BO* and *force_Δ*, which constitute the kernel of energy and force calculations. The *force_BO* and *force_Δ* algorithms are given in Tables 3 and 4, respectively. The algorithm *force_BO* in Table 3 computes force components involved in single BO term, where the A_1, A_2 and A_3 terms are the pre-computed derivatives of BO_{ij} with respect to BO'_{ij}, Δ'_i and Δ'_j, respectively. The algorithm *force_Δ* in Table 4 computes the force contributions arising from the derivative with respect to Δ_i, which is simply a wrapper function of *force_BO* looping over the neighbor list of the ith atom.

We now define six algorithms (5 valence and 1 noncovalent) for force calculations, each of which contains energy functions sharing the same computational characteristics, in Sections 2.2.1–2.2.6. Combined with a multiple time-step integrator, this grouping of the force components facilitates efficient time integration, which will be discussed in Section 2.4.

2.2.1. 1-body energies

Among the six force-calculation algorithms, *force_1body* is responsible for the 1-body energy calculations E_{lp}, E_{over} and E_{under}, which contain one explicit atom, see Eqs. (6)–(8). Here, the valency term in the lone-pair function Δ_i^{e}, the number of lone pair electrons $n_{\mathrm{lp},i}$, the coordination effect Δ_i^{lp}, and the correction term $\Delta_i^{\mathrm{lpcorr}}$ due to lone-pair electrons in the over/under coordination energy function E_{over} and E_{under} are defined in Eq. (9). The parameters in Eqs. (6)–(9) are listed in Table A.2 in Appendix A.

$$E_{\mathrm{lp}} = \sum_i \frac{p_{\mathrm{lp2}} \Delta_i^{\mathrm{lp}}}{1 + \exp(-75\Delta_i^{\mathrm{lp}})}, \tag{6}$$

Table 3
Force calculation arising from the derivative of BO_{ij}

Algorithm force_BO
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$n(i), n(j)$ = lists of neighbor atoms for the ith and jth atoms
$C_{\mathrm{BO}} = \partial E/\partial BO_{ij}$
Output:
$\{\mathbf{f}_i\}$ = set of updated atomic forces
Steps:
$\mathbf{f}_i \leftarrow \mathbf{f}_i + C_{\mathrm{BO}} \times A_1 \times \mathbf{r}_{ij}$ // $A_1 = (\partial BO_{ij}/\partial BO'_{ij})(\partial BO'_{ij}/\partial r_{ij})$
$\mathbf{f}_j \leftarrow \mathbf{f}_j + C_{\mathrm{BO}} \times A_1 \times \mathbf{r}_{ji}$
for each atom $\mathbf{r}_k \in n(i)$
$\mathbf{f}_i \leftarrow \mathbf{f}_i + C_{\mathrm{BO}} \times A_2 \times \mathbf{r}_{ik}$ // $A_2 = (\partial BO_{ij}/\partial \Delta'_i)(\partial \Delta'_i/\partial r_{ik})$
$\mathbf{f}_k \leftarrow \mathbf{f}_k + C_{\mathrm{BO}} \times A_2 \times \mathbf{r}_{ki}$
for each atom $\mathbf{r}_k \in n(j)$
$\mathbf{f}_j \leftarrow \mathbf{f}_j + C_{\mathrm{BO}} \times A_3 \times \mathbf{r}_{jk}$ // $A_3 = (\partial BO_{ij}/\partial \Delta'_j)(\partial \Delta'_j/\partial r_{jk})$
$\mathbf{f}_k \leftarrow \mathbf{f}_k + C_{\mathrm{BO}} \times A_3 \times \mathbf{r}_{kj}$

Table 4
Force-calculation arising from the derivative of Δ_i

Algorithm force_Δ
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$n(i)$ = list of neighbor atoms for the ith atom
$C_\Delta = \partial E/\partial \Delta_i$
Output:
$\{\mathbf{f}_i\}$ = set of updated atomic forces
Steps:
for each atom $\mathbf{r}_j \in n(i)$
call *force_BO*$(\mathbf{r}_i, \mathbf{r}_j, C_\Delta)$

$$E_{\mathrm{over}} = \sum_i \frac{\sum_{j\in n(i)} p_{\mathrm{ovun1}} D_e^\sigma BO_{ij}}{\Delta_i^{\mathrm{lpcorr}} + Val_i} \Delta_i^{\mathrm{lpcorr}} \left[\frac{1}{1+\exp(p_{\mathrm{ovun2}} \Delta_i^{\mathrm{lpcorr}})} \right], \tag{7}$$

$$E_{\mathrm{under}} = \sum_i (-p_{\mathrm{ovun5}}) \frac{1-\exp(p_{\mathrm{ovun6}} \Delta_i^{\mathrm{lpcorr}})}{1+\exp(-p_{\mathrm{ovun2}} \Delta_i^{\mathrm{lpcorr}})} \frac{1}{1+p_{\mathrm{ovun7}} \exp[p_{\mathrm{ovun8}} \sum_{j\in n(i)} (\Delta_j - \Delta_j^{\mathrm{lp}})(BO_{ij}^\pi + BO_{ij}^{\pi\pi})]}, \tag{8}$$

$$\Delta_i^{\mathrm{e}} = -Val_i^{\mathrm{e}} + \sum_{j\in n(i)} BO_{ij}, \tag{9a}$$

$$n_{\mathrm{lp},i} = \left\lfloor \frac{\Delta_i^{\mathrm{e}}}{2} \right\rfloor + \exp\left[-p_{\mathrm{lp1}} \left(2 + \Delta_i^{\mathrm{e}} - 2\left\lfloor \frac{\Delta_i^{\mathrm{e}}}{2} \right\rfloor \right)^2 \right], \tag{9b}$$

$$\Delta_i^{\mathrm{lp}} = n_{\mathrm{lp\text{-}opt},i} - n_{\mathrm{lp},i}, \tag{9c}$$

$$\Delta_i^{\mathrm{lpcorr}} = \Delta_i - \frac{\Delta_i^{\mathrm{lp}}}{1+p_{\mathrm{ovun3}} \exp(p_{\mathrm{ovun4}} \sum_{j\in n(i)} (\Delta_j - \Delta_j^{\mathrm{lp}})(BO_{ij}^\pi + BO_{ij}^{\pi\pi}))}. \tag{9d}$$

Due to the triply nested neighbor-list loops in the summation of the valency Δ (see Table 5), *force_1body* requires the traversal of a deep tree structure emanating from each primary atom in force computation with a relatively large interaction cutoff distance. In Table 5, $E_{1\mathrm{body}}$ denotes one of E_{lp}, E_{over} and E_{under} terms.

2.2.2. Bonding (2-body) energy

E_{bond} in Eq. (10) describes the covalent bond energy of atomic pairs, which includes energy contributions from the σ, π and double-π bond terms. The parameters in the 2-body interactions are listed in Table A.3 in Appendix A.

$$E_{\mathrm{bond}} = \sum_i \sum_j \left\lfloor -D_e^\sigma \cdot BO_{ij}^\sigma \cdot \exp\left[p_{\mathrm{be1}} \left(1 - \left(BO_{ij}^\sigma \right)^{p_{\mathrm{be2}}} \right) \right] - D_e^\pi \cdot BO_{ij}^\pi - D_e^{\pi\pi} \cdot BO_{ij}^{\pi\pi} \right\rfloor. \tag{10}$$

Algorithm *force_2body* in Table 6 computes E_{bond}. Though Table 6 does not distinguish the σ, π and double-π sub-terms for simplicity, they are handled simply by inputting different derivatives in algorithm *force_BO*. In Table 6, $ID(i)$ denotes a sequential index of the ith atom. (Our parallel MD algorithm maintains global sequential indices over all atoms across processors.)

Table 5
1-body force calculation

Algorithm force_1body
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of the ith atom
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_1 = \partial E_{1\text{body}}/\partial BO_{ij}$, $C_2 = \partial E_{1\text{body}}/\partial \Delta_i$
Steps:
for each atom $\mathbf{r}_i$
compute C_1 and C_2
for each atom $\mathbf{r}_j \in n(i)$
call $force\_BO(\mathbf{r}_i, \mathbf{r}_j, C_1)$
call $force\_\Delta(\mathbf{r}_j, C_2)$

Table 6
2-body force calculation

Algorithm force_2body
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of ith atom
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_3 = \partial E_{\text{bond}}/\partial BO_{ij}$
Steps:
for each atom $\mathbf{r}_i$
for each atom $\mathbf{r}_j \in n(i)$
if $(ID(i) > ID(j))$ then
compute C_3
call $force\_BO(\mathbf{r}_i, \mathbf{r}_j, C_3)$

2.2.3. 3-body energies

The 3-body interaction functions, E_{val}, E_{pen}, and E_{coa} in Eqs. (11)–(13), describe the stability of the valence angle Θ_{ijk} and covalent bonds between (i, j) and (j, k) atoms. In E_{val}, the function f_7 in Eq. (11a) guarantees that the energy term goes to zero as the two covalent bonds from the center atom dissociate. Eqs. (11c)–(11f) compute the equilibrium valence angle taking into account π, double-π and lone pair electron effects. E_{pen} is the penalty energy function in atomic triplets. E_{coa} is the conjugation energy among triplet atoms, for example an NO_2 group in nitramines. The parameters in Eqs. (11)–(13) are listed in Table A.4 in Appendix A.

$$E_{\text{val}} = \sum_i \sum_j \sum_k f_7(BO_{ij}) f_7(BO_{jk}) f_8(\Delta_j) \left\{ p_{\text{val1}} - p_{\text{val1}} \exp\left[-p_{\text{val2}} \left(\Theta_0(SBO_j) - \Theta_{ijk} \right)^2 \right] \right\}, \tag{11}$$

$$f_7(BO_{ij}) = 1 - \exp\left(-p_{\text{val3}} BO_{ij}^{p_{\text{val4}}} \right), \tag{11a}$$

$$f_8(\Delta_j) = p_{\text{val5}} - (p_{\text{val5}} - 1) \frac{2 + \exp(p_{\text{val6}} \Delta_j^{\text{angle}})}{1 + \exp(p_{\text{val6}} \Delta_j^{\text{angle}}) + \exp(-p_{\text{val7}} \Delta_j^{\text{angle}})}, \tag{11b}$$

$$SBO_j = \sum_{m \in n(j)} \left(BO_{jm}^{\pi} + BO_{jm}^{\pi\pi} \right) + \left[1 - \prod_{m \in n(j)} \exp\left(-BO_{jm}^{8} \right) \right] \left(-\Delta_j^{\text{angle}} - p_{\text{val8}} n_{\text{lp},j} \right), \tag{11c}$$

$$\Delta_j^{\text{angle}} = -Val_j^{\text{angle}} + \sum_{m \in n(i)} BO_{jm}, \tag{11d}$$

$$SBO2 = \begin{cases} 0 & (SBO \leqslant 0) \\ SBO^{p_{\text{val9}}} & (0 < SBO \leqslant 1) \\ 2 - (2 - SBO)^{p_{\text{val9}}} & (1 < SBO \leqslant 2) \\ 2 & (SBO > 2) \end{cases}, \tag{11e}$$

$$\Theta_0(SBO) = \pi - \Theta_{0,0} \left\{ 1 - \exp\left[-p_{\text{val10}} (2 - SBO2) \right] \right\}, \tag{11f}$$

Table 7
3-body force calculation

Algorithm force_3body
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of ith atom
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_{4ij} = \partial E_{3\text{body}}/\partial BO_{ij}$, $C_{4jk} = \partial E_{3\text{body}}/\partial BO_{jk}$
$C_{5i} = \partial E_{3\text{body}}/\partial \Delta_i$, $C_{5j} = \partial E_{3\text{body}}/\partial \Delta_j$, $C_{5k} = \partial E_{3\text{body}}/\partial \Delta_k$
Steps:
for each atom $\mathbf{r}_j$
for each atom $\mathbf{r}_i \in n(j)$
for each atom $\mathbf{r}_k \in n(j)$
if $(ID(i) > ID(k))$ then
compute C_4 and C_5
call *force_BO*$(\mathbf{r}_i, \mathbf{r}_j, C_{4ij})$
call *force_BO*$(\mathbf{r}_j, \mathbf{r}_k, C_{4jk})$
call *force_Δ*$(\mathbf{r}_i, C_{5i})$
call *force_Δ*$(\mathbf{r}_j, C_{5j})$
call *force_Δ*$(\mathbf{r}_k, C_{5k})$
compute force components from 3-body angle derivatives

$$E_{\text{pen}} = \sum_i \sum_j \sum_k p_{\text{pen1}} f_9(\Delta_j) \exp\left[-p_{\text{pen2}}(BO_{ij} - 2)^2\right] \exp\left[-p_{\text{pen2}}(BO_{jk} - 2)^2\right], \tag{12}$$

$$f_9(\Delta_j) = \frac{2 + \exp(-p_{\text{pen3}}\Delta_j)}{1 + \exp(-p_{\text{pen3}}\Delta_j) + \exp(p_{\text{pen4}}\Delta_j)}, \tag{12a}$$

$$E_{\text{coa}} = \sum_i \sum_j \sum_k p_{\text{coa1}} \frac{1}{1 + \exp(p_{\text{coa2}}\Delta_j^{\text{val}})} \exp\left[-p_{\text{coa3}}\left(-BO_{ij} + \sum_{m\in n(i)} BO_{im}\right)^2\right]$$
$$\times \exp\left[-p_{\text{coa3}}\left(-BO_{jk} + \sum_{m\in n(k)} BO_{km}\right)^2\right] \exp\left[-p_{\text{coa4}}(BO_{ij} - 1.5)^2\right] \exp\left[-p_{\text{coa4}}(BO_{jk} - 1.5)^2\right]. \tag{13}$$

Algorithm *force_3body* in Table 7 computes E_{val}, E_{pen} and E_{coa} energies, which have three explicit atoms (Eqs. (11)–(13)). In *force_3body* (and also in *force_4body* described below), the force arising from angular derivatives needs to be computed separately from BO derivatives, as is described elsewhere [21]. In Table 7, $E_{3\text{body}}$ denotes one of E_{val}, E_{pen} and E_{coa}.

2.2.4. 4-body energies and parameters

4-body dihedral angle energy E_{tors} and 4-body conjugation energy E_{conj} are described by Eqs. (14) and (15). The angles Θ_{ijk} and Θ_{jkl} are computed from atom triplets (i, j, k) and (j, k, l). The dihedral angle ω_{ijkl} is the angle between the two planes defined by the two atom triplets. The parameters in the 4-body interactions are listed in Table A.5 in Appendix A.

$$E_{\text{tors}} = \frac{1}{2} \sum_i \sum_j \sum_k \sum_l f_{10}(BO_{ij}, BO_{jk}, BO_{kl}) \sin\Theta_{ijk} \sin\Theta_{jkl}$$
$$\times \left[V_1(1 + \cos\omega_{ijkl}) + V_2 \exp\left\{p_{\text{tor1}}\left(2 - BO_{jk}^{\pi} - f_{11}(\Delta_j, \Delta_k)\right)^2\right\}(1 - \cos 2\omega_{ijkl}) + V_3(1 + \cos 3\omega_{ijkl})\right], \tag{14}$$

$$f_{10}(BO_{ij}, BO_{jk}, BO_{kl}) = \left[1 - \exp(-p_{\text{tor2}}BO_{ij})\right]\left[1 - \exp(-p_{\text{tor2}}BO_{jk})\right]\left[1 - \exp(-p_{\text{tor2}}BO_{kl})\right], \tag{14a}$$

$$f_{11}(\Delta_j, \Delta_k) = \frac{2 + \exp[-p_{\text{tor3}}(\Delta_j^{\text{angle}} + \Delta_k^{\text{angle}})]}{1 + \exp[-p_{\text{tor3}}(\Delta_j^{\text{angle}} + \Delta_k^{\text{angle}})] + \exp[p_{\text{tor4}}(\Delta_j^{\text{angle}} + \Delta_k^{\text{angle}})]}, \tag{14b}$$

$$E_{\text{conj}} = \sum_i \sum_j \sum_k \sum_l p_{\text{cot1}} f_{12}(BO_{ij}, BO_{jk}, BO_{kl})\left[1 + \left(\cos^2\omega_{ijkl} - 1\right)\sin\Theta_{ijk}\sin\Theta_{jkl}\right], \tag{15}$$

$$f_{12}(BO_{ij}, BO_{jk}, BO_{kl}) = \exp\left[-p_{\text{cot2}}(BO_{ij} - 1.5)^2\right] \exp\left[-p_{\text{cot2}}(BO_{jk} - 1.5)^2\right] \exp\left[-p_{\text{cot2}}(BO_{kl} - 1.5)^2\right]. \tag{15a}$$

Table 8
4-body force calculation

Algorithm force_4body
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of ith atom
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_{6ij} = \partial E_{4\text{body}}/\partial BO_{ij}$, $C_{6jk} = \partial E_{4\text{body}}/\partial BO_{jk}$, $C_{6kl} = \partial E_{4\text{body}}/\partial BO_{kl}$
$C_{7j} = \partial E_{4\text{body}}/\partial \Delta_j$, $C_{7k} = \partial E_{4\text{body}}/\partial \Delta_k$
Steps:
for each atom $\mathbf{r}_j$
  for each atom $\mathbf{r}_k \in n(j)$
    if $(ID(j) > ID(k))$ **then**
      for each atom $\mathbf{r}_i \in n(j)$
        if $(ID(i) \neq ID(k))$ then
          for each atom $\mathbf{r}_l \in n(k)$
            if $(ID(i) \neq ID(k)$ **and** $ID(i) \neq ID(l)$ **and** $ID(j) \neq ID(l))$ **then**
              compute C_6 and C_7
              call $force\_BO(\mathbf{r}_i, \mathbf{r}_j, C_{6ij})$
              call $force\_BO(\mathbf{r}_j, \mathbf{r}_k, C_{6jk})$
              call $force\_BO(\mathbf{r}_k, \mathbf{r}_l, C_{6kl})$
              call $force\_\Delta(\mathbf{r}_j, C_{7j})$
              call $force\_\Delta(\mathbf{r}_k, C_{7k})$
              compute force components from 4-body angle derivatives

Algorithm *force_4body* in Table 8 computes E_{tors} and E_{conj} energies that involve four explicit atoms (Eqs. (14) and (15)). *force_4body* consumes large amount of computation time to handle atomic quadruplets at every MD step, which can be reduced by the multiple time-step scheme described in Section 2.4. In Table 8, $E_{4\text{body}}$ denotes either E_{conj} or E_{tors}.

2.2.5. Hydrogen–bonding energy and parameters

The hydrogen–bonding energy function E_{hbond} in Eq. (16) takes a 3-body interaction form, i.e., as a function of the (i, j) and (j, k) bonds as well as the angle Θ_{ijk}. The E_{hbond} term thus consists of three explicit atoms, where the jth atom is hydrogen, the ith atom is chosen from atom j's neighbor list, and the kth atom is selected within the noncovalent interaction cutoff radius r_{cnb}(∼10 Å) from atom i. The parameters in the hydrogen–bonding interactions are listed in Table A.6 in Appendix A.

$$E_{\text{hbond}} = \sum_i \sum_j \sum_k p_{\text{hb1}}\left[1 - \exp(p_{\text{hb2}} BO_{ij})\right] \exp\left[p_{\text{hb3}}\left(\frac{r^o_{\text{hb}}}{r_{jk}} + \frac{r_{jk}}{r^o_{\text{hb}}} - 2\right)\right] \sin^8\left(\frac{\Theta_{ijk}}{2}\right). \tag{16}$$

To handle the unique form of E_{hbond}, we define a separate algorithm *force_hbond* in Table 9. Because of its asymmetric structure consisting of valence and noncovalent bonds, it is necessary that atomic triplets (i, j, k) and (k, j, i) are dealt differently. Forces arising from the derivative with respect to noncovalent atoms k are thus computed separately in Table 9.

Table 9
Hydrogen–bonding force calculation

Algorithm force_hbond
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
$\{n(i)\}$ = set of neighbor lists, where $n(i)$ is the list of neighbor atoms of ith atom
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_8 = \partial E_{\text{hbond}}/\partial BO_{ij}$
Steps:
for each atom $\mathbf{r}_j$
  for each atom $\mathbf{r}_i \in n(j)$
    for each atom $\mathbf{r}_k \in \{\mathbf{r}_k \,||\mathbf{r}_{ik}| \leqslant r_{\text{cnb}}\}$
      compute C_8
      call $force\_BO(\mathbf{r}_i, \mathbf{r}_j, C_8)$
      compute force components from 3-body angle derivatives
      compute force components from noncovalent term

Table 10
Noncovalent force calculation

Algorithm force_noncov
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
Output:
$\{\mathbf{f}_i\}$ = set of atomic forces
Variables:
$C_9 = \partial E_{\text{noncov}}/\partial r_{ij}$
Steps:
for each atom $\mathbf{r}_i$
for each atom $\mathbf{r}_j \in \{\mathbf{r}_j | |\mathbf{r}_{ij}| \leqslant r_{\text{cnb}}\}$
if $(ID(i) > ID(j))$ then
compute C_9
compute energy and force components

2.2.6. Noncovalent energies and parameters

The noncovalent interactions consist of E_{vdWaals} (Eq. (17)) and E_{Coulomb} (Eq. (18)) terms. The functions are screened by a taper function with a cutoff length r_{cnb} in Eq. (17b), where the coefficients of the polynomial are $Tap_7 = 20$, $Tap_6 = -70$, $Tap_5 = 84$, $Tap_4 = -35$, $Tap_3 = 0$, $Tap_2 = 0$, $Tap_1 = 0$, and $Tap_0 = 1$. Noncovalent energy functions in ReaxFF are all-atom interactions, which do not require any exclusion rule [22] as in some other force fields.

$$E_{\text{vdWaals}} = \sum_i \sum_j D_{ij} Tap(r_{ij}) \left\{ \exp\left[\alpha_{ij}\left(1 - \frac{f_{13}(r_{ij})}{r_{\text{vdW}}}\right)\right] - 2\exp\left[\frac{1}{2}\alpha_{ij}\left(1 - \frac{f_{13}(r_{ij})}{r_{\text{vdW}}}\right)\right]\right\}, \tag{17}$$

$$f_{13}(r_{ij}) = \left[r_{ij}^{p_{\text{vdW1}}} + \left(\frac{1}{\gamma_w}\right)^{p_{\text{vdW1}}}\right]^{1/p_{\text{vdW1}}}, \tag{17a}$$

$$Tap(r_{ij}) = \sum_{\alpha=0}^{7} Tap_\alpha \left(\frac{r_{ij}}{r_{\text{cnb}}}\right)^\alpha, \tag{17b}$$

$$E_{\text{Coulomb}}\big(\{\mathbf{r}_i\}, \{q_i\}\big) = \sum_i \chi_i q_i + \frac{1}{2}\sum_i \sum_j q_i H(r_{ij}) q_j, \tag{18}$$

$$H(r_{ij}) = J_i \delta_{ij} + \frac{Tap(r_{ij})}{[r_{ij}^3 + (1/\gamma_{ij})^3]^{1/3}}(1 - \delta_{ij}),$$

$$\delta_{ij} = \begin{cases} 1 & (i = j) \\ 0 & (i \neq j) \end{cases} \tag{18a}$$

where χ_i is the electronegativity, J_i is the self Coulomb repulsion coefficient, and γ_{ij} is a parameter for the smeared Coulombic function.

Algorithm *force_noncov* in Table 10 computes van der Waals energy E_{vdWaals} and Coulomb energy E_{Coulomb}, which are functions of interatomic distance with the cutoff distance r_{cnb} (Eqs. (17) and (18)). We use potential tables to compute the noncovalent interactions in order to reduce the number of floating-point operations. In Table 10, E_{noncov} denotes either E_{vdWaals} or E_{Coulomb}. Since the noncovalent interactions are functions of interatomic distance, precomputing the function values along with a proper interpolation scheme significantly reduce the computation time.

2.3. Electronegativity-equalization scheme

In ReaxFF, atomic charges q_i are variables that change dynamically in time. Whenever after atom coordinates are updated, the EEM subroutine updates the charge distribution $\{q_i\}$ by minimizing the Coulomb energy E_{Coulomb} under the charge-neutrality constraint, $\sum_i q_i = 0$. With the Lagrange-multiplier method, the constrained energy minimization is equivalent to solving the electronegativity equalization problem [20,23–25],

$$g_i \equiv -\frac{\partial E_{\text{Coulomb}}}{\partial q_i} = -\mu, \tag{19}$$

where μ is the electronegativity. We solve the constrained minimization problem using the conjugate-gradient method [26,27] in Table 11, in which the gradient vector g_i is orthogonalized to the uniform shift vector $e_i = 1 (i = 1, \ldots, N)$. Because of the electronegativity equalization condition, Eq. (19), the force contribution from $E_{\text{Coulomb}}(\{\mathbf{r}_i\}, \{q_i\})$,

$$-\frac{\partial E_{\text{Coulomb}}}{\partial \mathbf{r}_i} - \sum_{k=1}^{N} \frac{\partial E_{\text{Coulomb}}}{\partial q_k}\frac{\partial q_k}{\partial \mathbf{r}_i} = -\frac{\partial E_{\text{Coulomb}}}{\partial \mathbf{r}_i} - \mu \frac{\partial}{\partial \mathbf{r}_i}\sum_{k=1}^{N} q_k = -\frac{\partial E_{\text{Coulomb}}}{\partial \mathbf{r}_i}, \tag{20}$$

Table 11
Electronegativity-equalization computation

Algorithm EEM
Input:
$\{\mathbf{r}_i\}$ = set of atomic positions
Output:
$\{q_i\}$ = set of atomic charges
Variables:
$\{g_i\} = \{-\partial E_{\text{Coulomb}}/\partial q_i\}$ = gradient vector
$\{h_i\}$ = conjugate-gradient vector
Steps:
compute the initial gradient vector $\{g_i^0\}$
$n \leftarrow 0$
do
if $(n = 1)$ **then**
$h_i^0 = g_i^0$ $(i = 1 \text{ to } N)$
else
$h_i^n \leftarrow g_i^{n-1} + \frac{g_i^{n-1}\cdot g_i^{n-1}}{g_i^{n-2}\cdot g_i^{n-2}} h_i^{n-1}$ $(i = 1 \text{ to } N)$
$q_i^n \leftarrow q_i^{n-1} + \frac{g_i^{n-1}\cdot g_i^{n-1}}{h_i^n \cdot H_{ij} \cdot h_j^n} h_i^n$ $(i = 1 \text{ to } N)$
$g_i^n \leftarrow g_i^{n-1} + \frac{g_i^{n-1}\cdot g_i^{n-1}}{h_i^n \cdot H_{ij} \cdot h_j^n} H_{ij} h_j^n$ $(i = 1 \text{ to } N)$
$n \leftarrow n + 1$
until the energy is converged

does not contain chain-rule terms associated with variables q_k, and thus the noncovalent force calculation described in Section 2.2.6 remains valid (the last equality in Eq. (20) results from the charge-neutrality condition, $\sum_k q_k = 0$). It is also noteworthy that the Hessian $H(r_{ij})$ in Eq. (18) is a constant until atom coordinates are updated. Thus, reusing pre-computed $H(r_{ij})$ during the EEM loop in Table 11 significantly reduces the total computation time.

2.4. Time integrator

Our parallel ReaxFF algorithm employs the rRESPA time integrator [28–30], which is a time-reversible algorithm based on symmetric Trotter decomposition of the time-propagation operator. It allows decomposing the energy functions into groups with distinct characteristic time scales. Accordingly, the forces $\mathbf{f}_i$ are decomposed into stiff force components $\mathbf{f}_{\text{stiff},i}$, which need to be updated with higher frequency, and soft and slowly varying force components $\mathbf{f}_{\text{soft},i}$:

$$\mathbf{f}_i = \mathbf{f}_{\text{soft},i} + \mathbf{f}_{\text{stiff},i}. \tag{21}$$

Table 12 shows the rRESPA algorithm comprising of a doubly nested time integration loop. The compute intensive force calculation (i.e., $\mathbf{f}_{\text{soft},i}$) is performed only once per outer loop with a large time step Δt (typically $\Delta t \sim 1$ fs). The atomic velocities are updated according to $\mathbf{f}_{\text{soft},i}$ at the beginning and the end of the outer loop. The inner loop updates the atomic positions and velocities with a finer time resolution $\delta t = \Delta t / N_{\text{mts}}$ using the stiff force components $\mathbf{f}_{\text{stiff},i}$ (typically $N_{\text{mts}} \sim 5$).

It is often the case that the soft force components $\mathbf{f}_{\text{soft},i}$ are computationally intensive [28]. Thus proper force decomposition significantly reduces the total computation time. For example, in dense covalently bonded materials, the 4-body force calculation is most time consuming. In test calculations on a diamond, the 4-body force calculation accounts for nearly half of the total computation time. We have reduced the total simulation time by 30% with the number of multiple time step $N_{\text{mts}} = 5$ (see Table 12) by separating out the 4-body force terms into $f_{\text{soft},i}$, i.e., by performing the time-consuming 4-body force calculation only once in 5 MD steps.

3. Parallelization

To design a scalable ReaxFF-MD simulation algorithm on massively parallel computers, we employ spatial decomposition [31] combined with linked-list cell and Verlet neighbor-list methods [21]. In spatial decomposition, the simulation space (with lengths L_x, L_y and L_z in the x, y and z directions, respectively) is decomposed into $P = P_x \times P_y \times P_z$ rectangular sub-domains, $\Omega_0 - \Omega_{P-1}$, of equal volume, which are then mapped to P processors in a parallel computer. Specifically atom i at position $\mathbf{r}_i = (r_{ix}, r_{iy}, r_{iz})$ is assigned to processor $p \in [0, P-1]$, where

$$\begin{cases} p = p_x P_y P_z + p_y P_z + p_z, \\ p_\alpha = \lfloor r_{i\alpha} P_\alpha / L_\alpha \rfloor \quad (\alpha = x, y, z). \end{cases} \tag{22}$$

Table 12
Multiple time-step algorithm

Algorithm multiple_time_step
Input:
$\{\mathbf{r}_i(t)\}$ = set of atomic positions at time t
$\{\mathbf{v}_i(t)\}$ = set of atomic velocities at time t
N_{step} = number of total MD steps
N_{mts} = number of multiple time steps
$\Delta t = N_{\text{mts}}\delta t$ = time discretization unit
Output:
$\{\mathbf{r}_i(t + N_{\text{step}}\delta t)\}$ = set of atomic coordinates at $t + N_{\text{step}}\delta t$
$\{\mathbf{v}_i(t + N_{\text{step}}\delta t)\}$ = set of atomic velocities at $t + N_{\text{step}}\delta t$
Variables:
$\{\mathbf{f}_i\}$ = set of atomic forces
Steps:
compute soft force components $\{\mathbf{f}_{\text{soft},i}\}$
for *outer* = 1 to $N_{\text{step}}/N_{\text{mts}}$
$\mathbf{v}_i \leftarrow \mathbf{v}_i + (\Delta t/2m_i)\mathbf{f}_{\text{soft},i}$ // m_i is the mass of the ith atom
compute stiff force components $\{\mathbf{f}_{\text{stiff},i}\}$
for *inner* = 1 to N_{mts}
$\mathbf{v}_i \leftarrow \mathbf{v}_i + (\delta t/2m_i)\mathbf{f}_{\text{stiff},i}$
$\mathbf{r}_i \leftarrow \mathbf{r}_i + \delta t\mathbf{v}_i$
compute stiff force components $\{\mathbf{f}_{\text{stiff},i}\}$
$\mathbf{v}_i \leftarrow \mathbf{v}_i + (\delta t/2m_i)\mathbf{f}_{\text{stiff},i}$
compute soft force components $\{\mathbf{f}_{\text{soft},i}\}$
$\mathbf{v}_i \leftarrow \mathbf{v}_i + (\Delta t/2m_i)\mathbf{f}_{\text{soft},i}$

Each processor stores the coordinates $\mathbf{r}_i$, velocities $\mathbf{v}_i$, and types σ_i (e.g., hydrogen, carbon, nitrogen, or oxygen) of all atoms that are assigned to it, and is also in charge of updating the coordinates and velocities by numerically integrating the Newton's equation of motion.

In the linked-list cell method, each domain is further decomposed into smaller rectangular cells. The atom information ($\mathbf{r}_i$, $\mathbf{v}_i$, σ_i) in each cell is organized using a linked list, which reduces the computational cost of neighbor-list calculation to $O(N/P)$, where N is the total number of atoms. We define the dimension of a linked-list cell L_c to be no less than the single covalent-bond cutoff r_{cb}. The r_{cb} is determined as the interatomic distance at which any type of BO becomes negligible. We use a criterion, $\forall BO'_{ij}(r_{\text{cb}}) < 10^{-4}$.

At the domain boundaries, it is necessary to copy sufficient atom information from neighbor domains to compute forces. In 3D simulation space, each domain has 26 adjacent domains, from which atom information needs to be cached into buffer layers that surround the domain (shown as a shaded area in Fig. 2). The requisite thickness of the buffer layer L_b is determined by the greatest cutoff distance among all interaction functions. Specifically, $L_b = \max(r_{\text{cnb}}, l_{\max} r_{\text{cb}})$, where $l_{\max}$ is the deepest level of covalently bonded trees with the primary atom as a root (see Fig. 2B). The caching operation is implemented as the exchange of

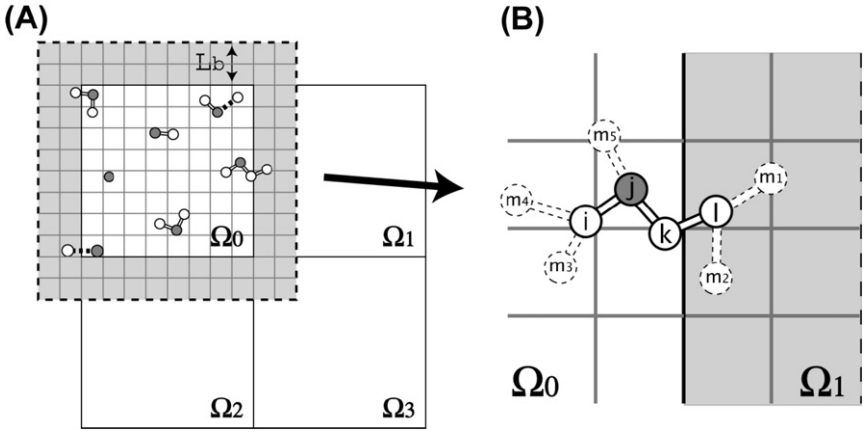

Fig. 2. (A) Schematic of the spatial decomposition consists of sub-domains Ω_0–Ω_3. On the sub-domain Ω_0 atom information from neighbor domains (Ω_1, Ω_2 and Ω_3 are stored in the buffer layer with the thickness L_b shown as shaded area surrounding Ω_0. The primary atoms in each cluster (gray spheres) must be in Ω_0, while other atoms (white spheres) may be copied from neighbor domains. (B) An example of 4-body atom configuration on a domain boundary. The explicit atoms are shown with solid-line spheres and bonds (i, j, k, and l), and so are implicit atoms with dotted-line spheres and bonds (m_1–m_5). Here jth atom is the primary of the cluster. In this example, the deepest level from the primary atom is $l_{\max} = 3$.

messages between processors, where the messages contain the information of atoms that lie within L_b from the domain boundary. After caching, processor p has the information of a set of atoms,

$$S = S_{\text{resident}} \cup S_{\text{cached}} = \{i \mid \mathbf{r}_i \in \Omega_p\} \cup \left\{i \mid \mathbf{r}_i \notin \Omega_p \wedge \|\mathbf{r}_i - \partial\Omega_p\| < L_b\right\}, \tag{23}$$

where $\partial\Omega_p$ is the outer-boundary surface of domain Ω_p and $\|\mathbf{r}_i - \partial\Omega_p\|$ is the shortest distance between atom i and $\partial\Omega_p$. This allows each processor to compute the forces on its resident atoms, $i \in S_{\text{resident}}$, without time-consuming interprocessor communications.

Our parallel ReaxFF algorithm attempts to minimize the total computation time by avoiding duplicated energy/force calculations among processors. For this purpose, care needs to be taken for the cached atoms stored in the buffer layer. Here, we illustrate the energy/force calculations in the parallel MD framework. Fig. 2B shows a sample case of 4-body calculation that involves four explicit atoms (denoted as i, j, k, and l) and five implicit atoms m_1–m_5. At the beginning of the energy/force calculation on domain Ω_0, atoms l, m_1, and m_2 are cached from domain Ω_1 to Ω_0 via communication network. We require that the primary atom j is a resident atom of domain Ω_0. The computed potential energy of the atomic cluster is stored in the domain of the primary atom. At the end of energy/force calculation, the reaction forces on the cached atoms are transferred back to the original atoms in domain Ω_1. Subsequently, atom coordinates are updated according to the time integration algorithm in Table 12. Resident atoms migrate to new domains if they have moved out of the current domain. All inter-domain communications follow a 6-way communication method [32]. The amount of communication is proportional to the number of atoms on domain surface, which is $O((N/P)^{2/3})$.

The parallel ReaxFF program is written in the Fortran 90 language with the message passing interface (MPI) library [33] for communications.

4. Performance test

The scalability of parallel ReaxFF program has been tested on various high-end supercomputers, including the 131,072-processor IBM BlueGene/L at the Lawrence Livermore National Laboratory (LLNL), the 10,240-processor SGI Altix 3000 at the NASA Ames Research Center, and an AMD dual-core Opteron cluster at the High Performance Computing Center (HPCC) of the University of Southern California (USC). The parallel ReaxFF algorithm maximally utilizes data locality regardless of actual physical architecture, and accordingly no performance degradation has been observed on any of these architectures. Description of each platform and computational resources used for each benchmark are given below.

BlueGene/L

The BlueGene/L system at the LLNL comprises of 65,536 computational node (CN) chips, each of which has two PowerPC 400 processors (131,072 processors in total) with 700 MHz clock speed. On single CN, the two processors share 512 MB memory. Each processor has a 32 KB instruction/data cache, a 2 MB L2 cache, and a 4 MB L3 cache. The theoretical peak performance is 2.8 Gflops per processor. Two types of interconnection (3-D torus and tree topologies) are designed for distinct purposes. The 3-D torus network is used mostly for common (e.g., point-to-point) communications, while the tree network is optimized for collective communications. The interconnection bandwidths are 175 MB/s and 350 MB/s per link, respectively.

Altix 3000

The SGI Altix 3000 system named Columbia at NASA-Ames consists of 20 of SGI Altix model 3700 boxes, each consisting of 512 Intel 1.5 GHz Itanium2 processors. Each processor has 128 floating-point registers, a 32 KB L1 cache, a 256 KB L2 cache, and a 6 MB L3 cache, and its theoretical peak performance is 6 Gflops. The SGI NUMAlink4 interconnection provides 1TB memory globally shared among cluster nodes within an Altix box. We have used up to four Altix boxes (up to 1920 processors) for our benchmark.

Opteron

USC-HPCC operates 1,824-node Linux cluster with 15.8 Tflops Linpack performance. We have used up to 512 nodes of dual-CPU dual-core AMD 2 GHz Opteron processors (2,048 processors in total). Each node has 4 GB memory, and Myrinet interconnection provides 256 MB/s bandwidth. Single Opteron core has 64 KB instruction/data caches and 1 MB L2 cache.

Fig. 3 shows the parallel efficiency of the parallel ReaxFF algorithm on the three architectures. Here, the isogranular parallel efficiency is defined as the total execution time divided by that on $P = 1$, where the grain size (i.e., the number of atoms per processor) is kept constant—$N/P = 10{,}752$ on the BlueGene/L, 36,288 on Columbia, and 107,520 on the Opteron cluster. In all performance tests, we simulate energetic molecular crystal, RDX (1,3,5-trinitro-1,3,5-triazine, $C_3N_6O_6H_6$) [14]. Overall, the parallel ReaxFF algorithm achieves nearly perfect ($\sim$1) efficiency. On 131,072 BlueGene/L processors, the parallel efficiency

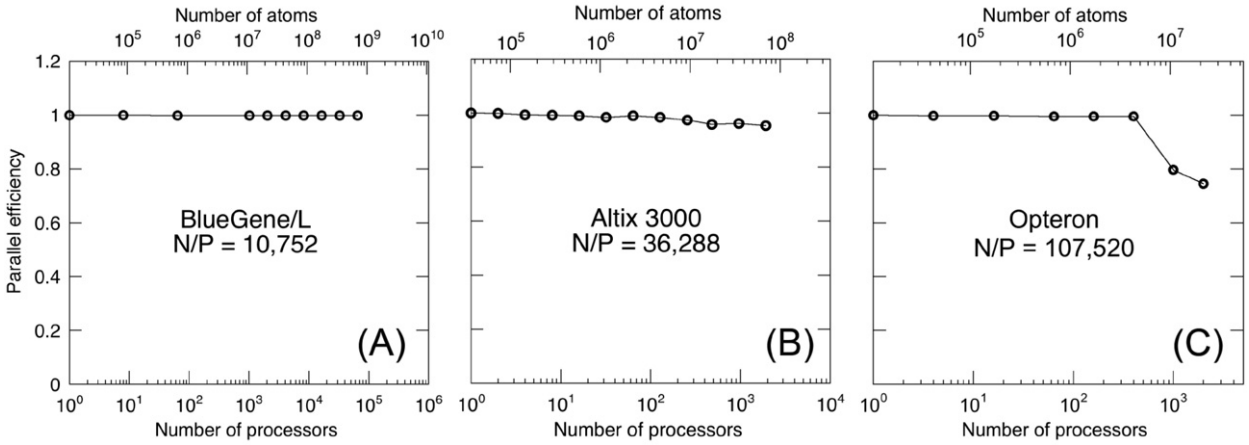

Fig. 3. Isogranular parallel efficiency of the parallel ReaxFF algorithm as a function of the number of processors on (A) IBM Blue Gene/L with the number of atoms per processor $N/P = 10{,}752$, (B) SGI Altix 3000 with $N/P = 36{,}288$, and (C) AMD dual-core Opteron cluster with $N/P = 107{,}520$.

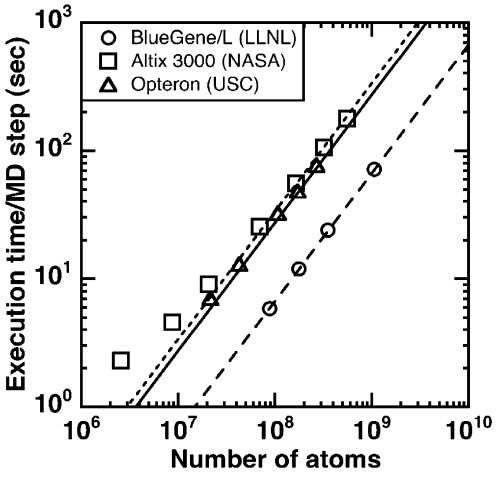

Fig. 4. Execution time of the parallel ReaxFF program as a function of the number of atoms on 131,072-processor IBM Blue Gene/L, 1920-processor SGI Altix 3000, and 2048-core AMD Opteron.

is 0.998. Only exception is the sudden drop in the parallel efficiency on the Opteron cluster, which is likely due to the non-dedicated testing environment, in which other users' processes share the same network.

Fig. 4 shows the execution time as a function of the number of atoms, while the total number of processors is fixed. We use 131,072 IBM BlueGene/L processors, 1920 Itenium2 processors of SGI Altix 3000, and 2048 AMD Opteron cores. The largest number of atoms is 1,056,964,608 on the BlueGene/L. The parallel ReaxFF algorithm exhibits perfect linear scalability as a function of the problem size.

5. Summary

For chemically-reactive molecular dynamics simulations, we have designed a scalable parallel algorithm that maximally exposes data locality. The parallel ReaxFF algorithm incorporates spatial decomposition and multiple time-stepping for efficient valence/noncovalent force calculations. Environment-dependent variable charges are incorporated with an electronegativity equalization method. Benchmark tests have exhibited high scalability (parallel efficiency is 0.998 on 131,072 BlueGene/L processors) as well as good performance portability. The parallel ReaxFF algorithm enables unprecedented scales of chemically reactive simulations, which pave a way to study longstanding scientific problems such as mechanically induced chemical reactions.

Acknowledgements

This work was partially supported by ARO—MURI, DOE, DTRA, and NSF. Performance tests were performed at the University of Southern California using the 5384-processor Linux cluster at the Research Computing Facility and the 2048-processor Linux

cluster at the Collaboratory for Advanced Computing and Simulations. We thank Dr. A.C.T. van Duin and Prof. W.A. Goddard, III, for valuable discussions.

Appendix A. ReaxFF parameters

Appendix A contains Tables A.1–A.7.

Table A.1
The parameters in the bond-order functions

p_{bo1}–p_{bo6}	bond parameters
$r_o^{\sigma}, r_o^{\pi}, r_o^{\pi\pi}$	bond radius parameters
$Val_i, Val_i^{\mathrm{boc}}$	valencies of atom i
$p_{\mathrm{boc1}}, p_{\mathrm{boc2}}$	overcoordination parameters
p_{boc3}–p_{boc5}	1–3 bond order corrections

Table A.2
The parameters in the 1-body energy

Val_i^e	valency of atom i
$n_{\mathrm{lp\text{-}opt},i}$	the number of lone pair electrons at normal condition
p_{lp1}	lone pair parameter
p_{lp2}	lone pair energy parameter
p_{ovun1}–p_{ovun4}	overcoordination parameters
p_{ovun5}	undercoordination energy parameter
p_{ovun6}–p_{ovun8}	undercoordination parameters

Table A.3
The parameters in the E_{bond}

$D_e^{\sigma}, D_e^{\pi}, D_e^{\pi\pi}$	σ, π, and double π bond energy parameters
$p_{\mathrm{be1}}, p_{\mathrm{be2}}$	bonding energy parameters

Table A.4
The parameters in the 3-body energy

p_{val1}	valence angle energy parameter
p_{val2}–p_{val7}	valence angle parameters
p_{val8}	valency/lone pair parameter
$p_{\mathrm{val9}}, p_{\mathrm{val10}}$	valence angle parameters
$\Theta_{0,0}$	equilibrium valence angle
Val_i^{angle}	atom valency
p_{pen1}	penalty energy parameter
p_{pen2}	double bond/angle parameter
$p_{\mathrm{pen3}}, p_{\mathrm{pen4}}$	double bond/angle parameters for overcoordination
p_{coa1}	3-body conjugation energy parameter
p_{coa2}–p_{coa4}	valency angle conjugation parameters

Table A.5
The parameters in the 4-body energy

V_1–V_3	torsion energy parameters
p_{tor1}	torsion parameter
p_{tor2}	torsion/BO parameter
$p_{\mathrm{tor3}}, p_{\mathrm{tor4}}$	torsion overcoordination parameters
p_{cot1}	4-body conjugation energy parameter
p_{cot2}	4-body conjugation parameter

Table A.6
The parameters in the hydrogen–bonding energy

p_{hb1}	hydrogen bond energy parameter
$p_{\mathrm{hb2}}, p_{\mathrm{hb3}}$	hydrogen bond parameters
r_{hb}^o	hydrogen bond radius

Table A.7
The parameters in the noncovalent energy

D_{ij}	van der Waals energy parameter
α_{ij}	van der Waals parameter
r_{vdW}	van der Waals parameter
p_{vdW1}	van der Waals shielding
γ_{w}	van der Waals parameter
γ_{ij}	Coulomb parameter

References

[1] D.G. Truhlar, V. McKoy, Computing in Science & Engineering 2 (2000) 19.
[2] J.J. Gilman, Science 274 (1996) 65.
[3] J.J. Gilman, Materials Science and Technology 22 (2006) 430.
[4] S. Ogata, E. Lidorikis, F. Shimojo, A. Nakano, P. Vashishta, R.K. Kalia, Computer Physics Communications 138 (2001) 143.
[5] S. Dapprich, I. Komáromi, K.S. Byun, K. Morokuma, M.J. Frisch, Journal of Molecular Structures (Theochem) 461–462 (1999) 1.
[6] M.J. Field, P.A. Bash, M. Karplus, Journal of Computational Chemistry 11 (1990) 700.
[7] U.C. Singh, P.A. Kollman, Journal of Computational Chemistry 7 (1986) 718.
[8] A. Warshel, M. Levitt, Journal of Molecular Biology 103 (1976) 227.
[9] E.E. Santiso, K.E. Gubbins, Molecular Simulation 30 (2004) 699.
[10] J. Tersoff, Physical Review B 37 (1988) 6991.
[11] D.W. Brenner, Physica Status Solidi (b) 217 (2000) 23.
[12] A.C.T. van Duin, S. Dasgupta, F. Lorant, W.A. Goddard, Journal of Physical Chemistry A 105 (2001) 9396.
[13] A.C.T. van Duin, A. Strachan, S. Stewman, Q.S. Zhang, X. Xu, W.A. Goddard, Journal of Physical Chemistry A 107 (2003) 3803.
[14] A. Strachan, A.C.T. van Duin, D. Chakraborty, S. Dasgupta, W.A. Goddard, Physical Review Letters 91 (2003) 098301.
[15] Q. Zhang, T. Cagin, A.C.T. van Duin, W.A. Goddard, Y. Qi, L.G. Hector, Physical Review B 69 (2004) 045423.
[16] K.D. Nielson, A.C.T. van Duin, J. Oxgaard, W.Q. Deng, W.A. Goddard, Journal of Physical Chemistry A 109 (2005) 493.
[17] K. Chenoweth, S. Cheung, A.C.T. van Duin, W.A. Goddard, E.M. Kober, Journal of the American Chemical Society 127 (2005) 7192.
[18] K. Nomura, R.K. Kalia, A. Nakano, P. Vashishta, A.C.T. van Duin, W.A. Goddard, Physical Review Letters, in press.
[19] P. Vashishta, R.K. Kalia, A. Nakano, B.E. Homan, K.L. McNesby, Journal of Propulsion and Power 23 (2007) 688.
[20] W.J. Mortier, S.K. Ghosh, S. Shankar, Journal of the American Chemical Society 108 (1986) 4315.
[21] M.P. Allen, D.J. Tildesley, Computer Simulation of Liquids, Oxford University Press, 1989.
[22] W.D. Cornell, P. Cieplak, C.I. Bayly, I.R. Gould, K.M. Merz, D.M. Ferguson, D.C. Spellmeyer, T. Fox, J.W. Caldwell, P.A. Kollman, Journal of the American Chemical Society 118 (1996) 2309.
[23] A.K. Rappe, W.A. Goddard, Journal of Physical Chemistry 95 (1991) 3358.
[24] F.H. Streitz, J.W. Mintmire, Physical Review B 50 (1994) 11996.
[25] T.J. Campbell, R.K. Kalia, A. Nakano, P. Vashishta, S. Ogata, S. Rodgers, Physical Review Letters 82 (1999) 4866.
[26] A. Nakano, Computer Physics Communications 104 (1997) 59.
[27] B. Bayraktar, T. Bernas, J.P. Robinson, B. Rajwa, Pattern Recognition 40 (2007) 659.
[28] G.J. Martyna, M.E. Tuckerman, D.J. Tobias, M.L. Klein, Molecular Physics 87 (1996) 1117.
[29] M.E. Tuckerman, B.J. Berne, G.J. Martyna, Journal of Chemical Physics 97 (1992) 1990.
[30] M.E. Tuckerman, D.A. Yarne, S.O. Samuelson, A.L. Hughes, G.J. Martyna, Computer Physics Communications 128 (2000) 333.
[31] A. Nakano, R.K. Kalia, P. Vashishta, T.J. Campbell, S. Ogata, F. Shimojo, S. Saini, Scientific Programming 10 (2002) 263.
[32] A. Nakano, R.K. Kalia, K. Nomura, A. Sharma, P. Vashishta, F. Shimojo, A.C.T. van Duin, W.A. Goddard, R. Biswas, D. Srivastava, Computational Materials Science 38 (2007) 642.
[33] W. Gropp, E. Lusk, A. Skjellum, Using MPI, second ed., The MIT Press, 1999.