ON THE NUMERICAL SIMULATION OF FULLERENE NANOTUBES: $C_{100.000.000}$ AND BEYOND !

ATTILA CAGLAR AND MICHAEL GRIEBEL
Institut für Angewandte Mathematik, Sonderforschungsbereich 256, Universität Bonn, Wegelerstr. 6, D-53115 Bonn, Germany
E-mail: {caglar, griebel}@iam.uni-bonn.de

The investigation of material properties of fullerenes and especially nanotubes in experiments is very difficult. Here, computer simulations are an important tool to gain further insight. To obtain realistic results we use the bond-order potential due to Brenner. This potential describes the bonding structure (the forming/breaking of bonds) and other properties of graphite and hydrocarbons quite well. Based on this potential, we implemented a parallel algorithm for nanotube simulation. It uses the well known linked cell technique for short range pair potentials. However, due to the sophisticated structure of Brenner's many-body potential, a special implementation is necessary to gain an $O(N)$ complexity of the overall computation. For its parallelization, the domain decomposition method can be used in a straightforward way. We present the results of our numerical experiments. We considered the reactive collision of C_{60} fullerenes with benzyne and, moreover, the bending and stretching of nanotubes with nearly 100 million atoms. To our knowledge, this is the largest nanotube simulation up to now. The parallel algorithm runs on a workstation cluster Parnass2 and a CRAY T3E (512 proc.) It shows a nearly linear speed up and exhibits a very good scaling behavior. It turns out that nanotubes, at least in our simulations, can sustain hundreds of Giga Pascal in stretching experiments before fraction occurs.

1 Introduction

With the discovery of pure carbon structures other than graphite and diamond, i.e. the C_{60} fullerene balls by Smalley et al. [1] in 1985, a new fast developing area in nanomaterial science started. Subsequently also other forms of pure carbon were spotted, the most remarkable being very long tube-like structures. These carbon nanotubes were first reported by Iijima [2] in 1991. They were found in the sooth of experiments for producing fullerenes and come either as single or multiple walled

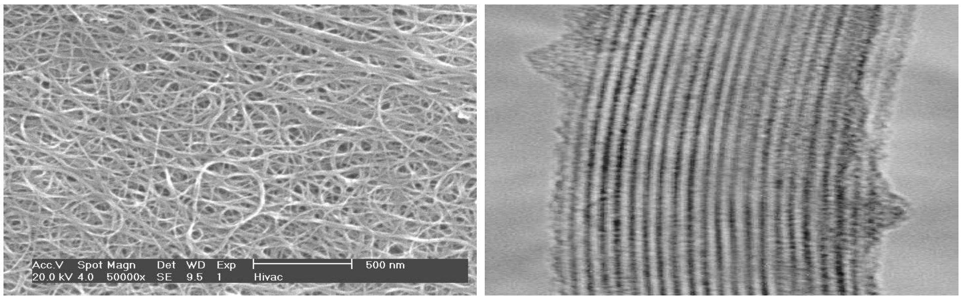

Figure 1. Scanning electron microscope images of nanotubes: Close-up view (left) of a purified nanotube sample, which can be bought at Rice University, Houston. The scale bar for this image is 2 microns. Zoom of the closed tip of a multilayered tube (right).

hollow fibers (see Figure 1).

Nanotubes possess very interesting properties: They are just about one to 17 nanometers in diameter but their length can be in the meter range [3] and is, at least in principle, not bounded. Thus, $C_{1.000.000}$ and even larger molecules might be created. Due to their interior structure and form, nanotubes exhibit fascinating physical properties: Mechanically they seem to sustain extreme distorsion with no fracture, elastic bending, twisting, buckling and other reversible deformations. Here, for the bending stiffness, the analog of Young's modulus is in the Tera Pascal range and a local tension of hundreds of Giga Pascal is anticipated before fraction occurs. In the future, this might lead to interesting applications like extremely durable but lightweighted ropes and wires build from nanotubes. Also extremely strong composites might be produced from nanotubes which do not exhibit the problem of microcrack propagation like most other uniform materials do. Further applications in scanning microscopy and near-field optical microscopy or nanostraws for chemical probing can be envisioned.

Electronically nanotubes behave like both, metals and semiconductors. This is in contrast to graphite which is only weakly conductive. Nanotubes can conduct like any good metal. Furthermore, currents in nanotubes change not continuously with the voltage applied, but merely in a step wise fashion. They are true quantum wires. This might make it possible to produce single electron transistors made from nanotubes, RAM storage devices and microelectronic processors in the future. For a more thorough discussion of the fascinating properties of nanotubes, see [4].

Meanwhile first progress is made to produce nanotubes in the laboratory. Here, gas phase condensation methods have been explored with some success. The carbon vapor is produced by resistive heating of a graphite rod, by a carbon arc discharge or by laser vaporization, see [3]. In principle, the formation of multi-walled nanotubes can be catalyzed by transition metals. But up to now, the produced fibers are not perfectly multi-walled along their length and one end of the tube is not perfectly formed. Furthermore, it was found that single walled nanotubes can be generated by a direct current arc discharge between carbon electrodes if certain metal (Co/Ni) is inserted in the anode. Alternatively they can be grown by laser vaporization of pure graphite (multi-wall nanotubes) or graphite/metal composites (single-walled nanotubes). However the deposition process is not completely understood, the process parameters are widely unclear and the annealing conditions must be further examined, see [3]. Furthermore, work with already produced nanotubes is in its infancy. It is extremely difficult to apply controllable forces to a nanotube for strength tests and fracture experiments even when laser technology is used. It is also a big problem how to connect a nanotube wire to other devices. Finally, measurements in the nanoregime are extremely complicated.

Therefore, computer simulation is an important tool to better understand the properties of nanotubes. The ab initio calculation of the electron structure by e.g. Hartree-Fock or density functional methods was done for C_{60} buckyballs, see [5]. However for larger molecules like long nanotubes this approach is not more feasible due to the complexity of these methods which result in extreme computing times and memory requirements. Thus, a simpler approach based on molecular dynamics must be chosen. But here, a major difficulty in modeling carbon nanotubes by MD computer simulation is the availability of realistic many-body atomic potentials. To this end, we have chosen Brenner's potential to describe bonded interactions [6]. This is a quite complicated generalization of Abell's and Tersoff's potentials [7,8] to carbon atoms and hydrocarbon molecules. This many-body potential has been shown to accurately reproduce experimental atomization energies for a wide range of hydrocarbon molecules. It realistically describes binding energies and bonding structure in graphite, diamond materials and hydrocarbon molecules. In addition, its proper description of bond forming and breaking can be used to observe reactive collision to bond benzyne onto fullerenes and also potential molecular gear failure modes.

We implemented a molecular dynamics code based on Brenner's potential. Here, the well known linked cell technique was employed. Parallelization was achieved using a standard domain decomposition technique and MPI was used for the implementation of the parallel code. We run a number of numerical experiments with C_{60} fullerenes and nanotubes. Here we considered reactive collision of C_{60} to benzyne and the bending and stretching of nanotubes with up to 100 million carbon atoms. The parallelization results of our code on the PC cluster Parnass2 [1] with 128 processors and a CRAY T3E with 512 processors show a nearly linear speed up and exhibit the very good scaling behavior of our algorithm.

The reminder of this paper is organized as follows: In section 2 we give the basic algorithm, discuss shortly the linked cell technique and describe the employed 5th order Beeman-Verlet [9] time integration scheme. In section 3 we present Brenner's potential in detail. A naive implementation would lead to an $O(N^4)$ complexity. But the use of cut-off techniques in the framework of the linked cell method allows for an $O(N)$ algorithm. Then, in section 4, we summarize the details of the parallelization of the code. Here, we follow the well known domain decomposition approach. Section 5 gives the results of our numerical experiments on our PC cluster Parnass2 and on the Cray T3E with up to 512 processors. We conducted bending and stretching experiments with nanotubes made of up to 100 million atoms. To our knowledge, this is the largest simulation for nanotubes up to now. It turned out that nanotubes, at least in our simulations, can sustain hundreds of Giga Pascal in stretching experiments before fraction occurs. This would make them nearly as strong as pure diamond. We close with some concluding remarks.

2 Basic algorithm, time integration, implementation

We consider a Hamiltonian system given by

$$\dot{q}_i = \frac{\partial H}{\partial p_i}, \quad \dot{p}_i = -\frac{\partial H}{\partial q_i}$$

with Hamilton operator H and generalized coordinates q_i and moments p_i. In the case of a conservative potential and cartesian coordinates x_i and velocities $u_i = \frac{p_i}{m_i}$, the Hamiltonian has the form

$$H(\{x_j, p_j\}_{j=1,\dots,N}) = \sum_i \frac{p_i^2}{2m_i} + V(\{x_j\}_{j=1,\dots,N})$$

[1] Parnass2, see http://wissrech.iam.uni-bonn.de/research/projects/parnass2

with a given potential V. Then, from these Hamiltonian equations, we directly obtain Newton's equations of motion

$$\begin{aligned} \dot{x}_i &= u_i, \\ m_i \dot{u}_i &= -\nabla_{x_i} V(\{x_j\}_{j=1,..,N}). \end{aligned} \tag{1}$$

The force F_i exerted on particle x_i is given by

$$F_i = -\nabla_{x_i} V(\{x_j\}_{j=1,..,N}) \tag{2}$$

and hence the acceleration a_i of particle x_i is obtained as

$$a_i = F_i / m_i. \tag{3}$$

Pair potentials V can be written as a sum over pair interactions $U(r_{kj})$,

$$V(\{x_j\}_{j=1,..,N}) = \sum_{k,j:k<j} U(r_{kj}), \quad r_{kj} = \|x_k - x_j\|.$$

Then, the force exerted on particle x_i is a sum over force contributions from the pairwise interactions, i.e.

$$\vec{F}_i = -\sum_{j\ :\ j\neq i} \nabla U(r_{ij}) \nabla_{x_i} r_{ij} = \sum_{j\ :\ j\neq i} \vec{F}_{ij} \tag{4}$$

Now, for a given state (positions $\{x_1, \cdots, x_N\}$ and velocities $\{u_1, \cdots, u_N\}$ of N particles), the time evolution of the system depends only on this state and the potential V.

For the solution of the system of ODEs (1) we have to employ a time integration scheme. Here, methods based on the Verlet approach (for details and variants, see [10], chapter 3) are quite common. In the following we will make use of a higher order generalization, the 5th order Beeman-Verlet technique. Then, (1) becomes

$$x_i^{(n+1)} := x_i^{(n)} + \delta t u_i^{(n)} + \frac{\delta t^2}{360}(323 a_i^{(n)} - 264 a_i^{(n-1)} + 159 a_i^{(n-2)} - 38 a_i^{(n-3)}) \tag{5}$$

$$a_i^{(n+1)} := F_i^{(n+1)} / m_i := -\nabla_{x_i^{(n+1)}} V(\{x_j^{(n+1)}\}_{j=1,..,N}) / m_i \tag{6}$$

$$x_i^{(n+1)} := x_i^{(n)} + \delta t u_i^{(n)} + \frac{\delta t^2}{360}(38 a_i^{(n+1)} + 171 a_i^{(n)} - 36 a_i^{(n-1)} + 7 a_i^{(n-2)}) \tag{7}$$

$$\delta t u_i^{(n+1)} := x_i^{(n+1)} - x_i^{(n)} + \frac{\delta t^2}{360}(97 a_i^{(n+1)} + 114 a_i^{(n)} - 39 a_i^{(n-1)} + 8 a_i^{(n-2)}). \tag{8}$$

Here $x_i^{(n)}$, $u_i^{(n)}$ and $a_i^{(n)}$ denotes the position, velocity and acceleration of particle x_i at time n, respectively. This approach resembles a predictor corrector method and should give a 5th order time accuracy $O((\delta t)^5)$. Here, δt is a fixed time-step size which is to be chosen properly. Equation (5) denotes the predictor step $P(\{x_i\})$ for the positions x_i at time $(n+1)$, equation (7) denotes the corrector step $C(\{x_i\})$ for the positions and equation (8) denotes the corrector step $C(\{u_i\})$ for the velocities at time $(n+1)$. The evaluation of forces and accelerations a_i according to (6) is denoted by $E(\{a_i\})$. Now, the complete algorithm for one time step is just

$$P(\{x_i\}) E(\{a_i\}) C(\{x_i\}) C(\{u_i\}). \tag{9}$$

Note that this scheme has a smaller truncation error than the traditional Verlet-scheme and shows good energy conservation from time step to time step.

Already for simple pair potentials, it gets clear from (5)-(8) that a naive implementation would lead to a $O(N^2)$ cost complexity for the evaluation of the potential and forces in each time step. Here, in the case of short range potentials, i.e. if the potential function decays very fast with distance, a cut off approximation helps to reduce the complexity to $O(N)$. To this end, the true potential function is set to zero in a distance larger than a cut off parameter r_{cut} and is locally modified (switching) such that a smooth function with local support is generated.

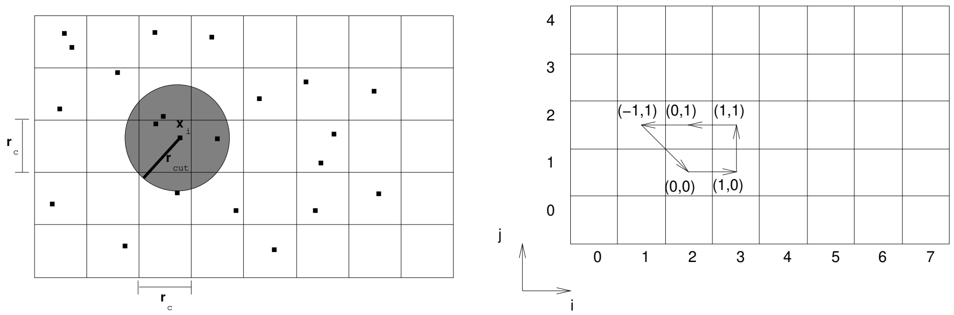

Figure 2. Linked-Cell-Method, 2D example. Cell decomposition of the domain and influence region of r_{cut} (left). Interaction path for the successive treatment of cells in the force evaluation (right).

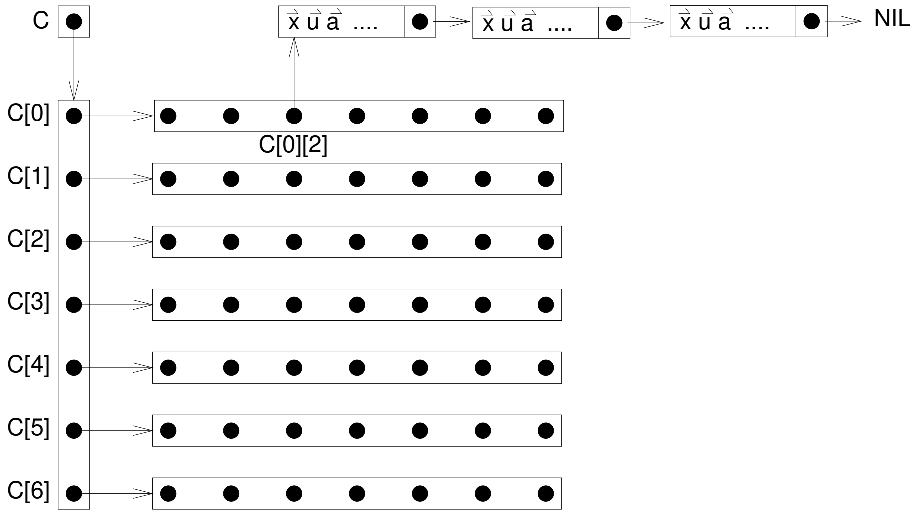

Figure 3. Storage structure of the linked list, 2D case.

The implementation of this principle directly leads to the so called linked cell approach [11]. Here the simulation domain is decomposed into equally sized cells $C_{i,j,(k)}$. To each cell a linked list is associated, which stores the particles that are located in this cell. The overall set of cells itself forms a three-dimensional array, where each element stores the starting address of the respective linked list. For the two-dimensional case, the basic principle is depicted in Figure 2 (left) and Figure 3. Here, the size of the cells is chosen to be slightly larger than r_{cut}. In this way, the force evaluation for one particle is confined to the particles which are placed in the same and the 8 (2D case) and 26 (3D case) neighboring cells only. Now the force evaluation can be computed by one pass through all cells in a linearized order. Altogether, this guarantees the desired $O(N)$ complexity for the force evaluation process. Note that further savings by about a factor of two are possible by using the symmetry which is due to Newton's third law, i.e.

$$\vec{F_{ij}} = -\vec{F_{ji}}. \tag{10}$$

Thus, we can avoid to compute certain forces in neighboring cells. Here, the force contribution of neighboring particles can be evaluated by following an interaction path (see Figure 2 (right)). Further details are given in Table 1 for the two-dimensional case. Its generalization to the 3D case is straightforward. In the following we will refer to this algorithm as *LC-loop*.

3 Potentials for Hydrocarbons

In this section we give Brenner's potential in detail and discuss its implementation in a linked-cell based code.

3.1 Brenner's potential

The potential due to Brenner [6] is one of the most sophisticated potentials for classical MD simulations up to now. It allows to simulate large carbon clusters not only in the classical sense of molecular dynamics, but, to some extent, it also takes quantum effects into account as pure *ab-initio* techniques do. In particular the Brenner potential is able to reproduce intramolecular energetics and bonding in solid diamond and graphite as well as a number of essential hydrocarbon molecules. Furthermore it allows for bond breaking and forming. The design of the potential is based on a bonding formalism due to Abell and Tersoff [7,8]. They derived a general expression for the binding energy as a sum of near-neighbor pair interactions which are moderated by the local atomic environment. Their approach is based on Pauling's bond-order ideas. It can realistically describe carbon-carbon single, double and triple bond-lengths and energies in hydrocarbons, solid graphite and diamond. But Abell's and Tersoff's potential is based on local near-neighbor interactions which results in nonphysical behavior for special situations where, for example, a carbon atom with four neighbors is bonded to a carbon atom with four neighbors. In this case the evaluation of the potential would result in something intermediate between a single and double bond. However, the bond would be better described by a single bond plus a radical orbital.

```
for l = 1 to 5
   nn_x = n_x + o_x[l]
   nn_y = n_y + o_y[l]

   forall particle i in cell (n_x, n_y)
      forall particle j in cell
      (nn_x, nn_y)
         if l > 1 or j > i then
            r_x = x_i - x_j
            r_y = y_i - y_j
            r = sqrt(r_x^2 + r_y^2)
            if r < r_cut then
               F_x(i) = F_x(i) + (F_ij)_x
               F_x(j) = F_x(j) - (F_ij)_x
               F_y(i) = F_y(i) + (F_ij)_y
               F_y(j) = F_y(j) - (F_ij)_y
            end if
         end if
      end forall j
   end forall i
end for l
```

Abbreviations:

N	number of particles
$F_x(i)$	force on particle x_i in x-direction
$F_y(i)$	force on particle x_i in y-direction
o_x	offset-array of interaction path (x-coord.) $o_x = \{0, 1, 1, 0, -1\}$
o_y	offset-array of interaction path (y-coord.) $o_y = \{0, 0, 1, 1, 1\}$
(n_x, n_y)	index of actual cell
(nn_x, nn_y)	index of respective cell in the interaction path
r_{cut}	cut-off parameter
x_i, y_i	coordinates of i-th particle
r_x, r_y, r, $(\vec{F}_{ij})_x$, $(\vec{F}_{ij})_y$	temporary variables: – distance between two particles in x-direction – distance between two particles in y-direction – distance between two particles – x-component of the force on x_i exerted by x_j – y-component of the force on x_i exerted by x_j

Table 1. Linked-Cell loop for the force evaluation in 2D in the case of a pair potential.

Brenner's approach is able to reproduce such a proper description of radicals and many other properties of carbon, because *non-local effects* are taken into account now. Hence it can distinguish conjugated and non-conjugated double bonds. Here, the potential is given as a sum over bonds, i.e.

$$V_b = \sum_i \sum_{j>i} \left[V_R(r_{ij}) - \bar{B}_{ij} V_A(r_{ij}) \right] \tag{11}$$

where $\bar{B}_{ij}$ is a so called bond order term (see below) and V_R denotes the repulsive and V_A the attractive pair term. These terms are derivated from Morse potentials and are defined by

$$V_R(r_{ij}) = f_{ij}(r_{ij}) \frac{D_{ij}^{(e)}}{S_{ij} - 1} e^{-\sqrt{2S_{ij}}\beta_{ij}(r_{ij} - R_{ij}^{(e)})} \tag{12}$$

$$V_A(r_{ij}) = f_{ij}(r_{ij}) \frac{D_{ij}^{(e)} S_{ij}}{S_{ij} - 1} e^{-\sqrt{\frac{2}{S_{ij}}}\beta_{ij}(r_{ij} - R_{ij}^{(e)})} \tag{13}$$

with parameters $D_{ij}^{(e)}, S_{ij}, \beta_{ij}$ and $R_{ij}^{(e)}$. Here, $D_{ij}^{(e)}$ resembles the well depth, S_{ij} and β_{ij} resemble fitting parameters and $R_{ij}^{(e)}$ corresponds to the average or equilibrium distance of the atoms i and j. These parameters are constant but depend on the atomic configuration, e.g. in the case of hydrocarbons they depend on whether i and j are carbon or hydrogen atoms. Note that for $S_{ij} = 2$ the pair terms (12) and (13) reduce to the usual Morse potential.

The most interesting part in (12) and (13) is the function $f_{ij}(r_{ij})$. It is defined as

$$f_{ij}(r) = \begin{cases} 1 & : \; r < R_{ij}^{(1)} \\ \left[1 + cos\left[\frac{\pi(r - R_{ij}^{(1)})}{(R_{ij}^{(2)} - R_{ij}^{(1)})}\right]\right] \Big/ 2 & : \; R_{ij}^{(1)} < r < R_{ij}^{(2)} \\ 0 & : \; r > R_{ij}^{(2)} \end{cases} \qquad (14)$$

where $R_{ij}^{(1)}$ and $R_{ij}^{(2)}$ are two given cut-off parameters. Outside the domain defined by the radius $R_{ij}^{(2)}$ around position x_i the function vanishes identically. Inside the domain given by the radius $R_{ij}^{(1)}$ around x_i we have $f_{ij}(r) = 1$ and in between f_{ij} decays gradually from 1 to 0. Altogether, f_{ij} is continuous. Here, f_{ij} can be seen as an enumerator for bonds and can be used to define the bonding connectivity in the system. With (14), it makes the potential function continuous.

Now, we will discuss the remaining term $\bar{B}_{ij}$, the so called bond order term. Its basic structure can be found already in the approach of Abell [7] and Tersoff [8]. Brenner generalized it in such a way that also nonlocal effects are included and hence cases as overbinding of radicals and conjugated versus nonconjugated bonds can be dealt with. To this end, let $N_i^{(C)}$ and $N_i^{(H)}$ be the number of the carbon and hydrogen atoms bonded to atom i, respectively. These quantities can easily be evaluated with help of the function f_{ij}. We have

$$N_i^{(C)} = \sum_{j(=carb)} f_{ij}(r_{ij}) \qquad \text{and} \qquad N_i^{(H)} = \sum_{j(=hydr)} f_{ij}(r_{ij}) \qquad (15)$$

where $\sum_{j(=carb)}$ and $\sum_{j(=hydr)}$ denote the summation over all other existing carbon and hydrogen atoms, respectively. Furthermore, let $N_i^{(t)}$ be the total number of bonded neighbors, i.e.

$$N_i^{(t)} = N_i^{(C)} + N_i^{(H)}. \qquad (16)$$

First we focus on the conjugated bonds. We define a continuous function

$$N_{ij}^{conj} = 1 + \sum_{k(=carb)(\neq i,j)} f_{ik}(r_{ik})F(z_{ik}) + \sum_{l(=carb)(\neq i,j)} f_{jl}(r_{jl})F(z_{jl}) \qquad (17)$$

which yields appropriate values, depending on whether bond (i,j) is part of a conjugated system or not. For example, if all neighbors k are carbon atoms that have a coordination less than 4 - that means $N_k^{(t)} < 4$ -, then the bond is defined as being part of a conjugated system. Here z_{ik} is defined as

$$z_{ik} = N_k^{(t)} - f_{ik}(r_{ik}) \qquad (18)$$

and

$$F(z_{ik}) = \begin{cases} 1 & : \quad z_{ik} \leq 2 \\ \{1 + cos[\pi(z_{ik} - 2)]\}/2 & : \quad 2 < z_{ik} < 3 \\ 0 & : \quad z_{ik} \geq 3 \end{cases} \tag{19}$$

is an enumerator similar to the former function f_{ij}. Note that the function N_{ij}^{conj} is also continuous in the case of breaking and forming bonds. For $N_{ij}^{conj} = 1$ bond (i, j) is not part of a conjugated system and for $N_{ij}^{conj} \geq 2$ the bond is part of a conjugated system.

Now we define a term B_{ij} as

$$B_{ij} = \left(1 + \sum_{k(\neq i,j)} G_i(\theta_{ijk}) f_{ik}(r_{ik}) e^{\alpha_{ijk}[(r_{ij} - R_{ij}^{(e)}) - (r_{ik} - R_{ik}^{(e)})]} + H_{ij}(N_i^{(H)}, N_i^{(C)}) \right)^{-\delta_i} \tag{20}$$

which resembles the bonding state of atom i with respect to atom j. Here, H_{ij} is a correction term which smoothes the transit from bonded to non-bonded state. It is given as a two-dimensional cubic spline, see Appendix. Furthermore G_i is a function of the angle θ_{ijk} between the bonds (i, j) and (i, k), see Appendix, $R_{ik}^{(e)}$ is a constant which gives the equilibrium distance, δ_i is a parameter depending on the respective atom i and α_{ijk} is a fitting constant, see Appendix.

Finally we are able to describe the empirical bond-order function $\bar{B}_{ij}$ in (11). It is basically the smoothed arithmetic mean of the two bonding states B_{ij} and B_{ji}, i.e.

$$\bar{B}_{ij} = (B_{ij} + B_{ji})/2 + K_{ij}(N_i^{(t)}, N_j^{(t)}, N_{ij}^{conj}) \tag{21}$$

where K_{ij} is a correction term, i.e. a three-dimensional cubic spline (see Appendix) that interpolates between values at discrete numbers of neighbors. It depends on the bonding connectivity, guarantees continuity of the potential and allows to fix overbinding of radicals for bonds between pairs of atoms with different coordinations. For further details and explanations see [6,7,8].

3.2 Implementation of Brenner's potential

The implementation of Brenner's potential and its efficient parallelization is quite cumbersome. Note that a naive computation of the potential and force terms results in an $O(N^4)$ complexity due to the bond order term $\bar{B}_{ij}$ which is basically a three body expression. Here, besides the conventional N^2-factor which stems from the double sums in (11), an additional factor N comes in from the sum over k in (20) and another factor N is due to the dependency of F in (17) from z_{ik} which itself depends via $N_k^{(t)}$ on the sum over j in (15).

Note however, that Brenner's potential is of short range form. Here, $R_{ij}^{(2)}$ in (14) and the clause $z_{ik} \geq 3$ in (19) act as cut-off parameters. Therefore we can follow the previously explained approach in the framework of the linked cell technique. In this way, an implementation with complexity $O(N)$ becomes possible. Nevertheless, due to the quite complicated form of Brenner's potential, an efficient implementation is not a trivial task. Here, the various sums arising in the above definitions (11) - (21) get truncated and may involve the atoms located in neighboring cells. Note however that also atoms might to be accessed which are located two cells away if no further remedies are taken. This is due to the intrinsic dependency of $\bar{B}_{ij}$ on $N_k^{(t)}$ and thus on z_{ik}. Such type of problems can be avoided by performing certain precomputations as we will see later on.

Since we want to evaluate the forces, we need the derivative of the potential, i.e.

$$F_{x_k} = -\nabla_{x_k} \sum_{\substack{i,j \\ i<j}} \left[V_R(r_{ij}) - \bar{B}_{ij} V_A(r_{ij}) \right]. \tag{22}$$

We obtain

$$\begin{aligned} F_{x_k} &= \sum_{\substack{i,j \\ i<j}} \left[-\nabla_{x_k} V_R(r_{ij}) + \bar{B}_{ij} \nabla_{x_k} V_A(r_{ij}) + (\nabla_{x_k} \bar{B}_{ij}) V_A(r_{ij}) \right] && (23) \\ &= \sum_{\substack{j \\ j\neq k}} \left[-\nabla_{x_k} V_R(r_{kj}) \right] + \sum_{\substack{j \\ j\neq k}} \left[\bar{B}_{kj} \nabla_{x_k} V_A(r_{kj}) \right] + \sum_{\substack{i,j \\ i<j}} (\nabla_{x_k} \bar{B}_{ij}) V_A(r_{ij}) && (24) \\ &=: F_{x_k}^R + F_{x_k}^A + F_{x_k}^{\bar{B}} && (25) \end{aligned}$$

Note, that equation (24) holds, since V_R, V_A and $\bar{B}_{ij}$ are symmetric, i.e. $V_R(r_{ij}) = V_R(r_{ji})$, etc.

Now we consider how the three different contributions $F_{x_k}^R$, $F_{x_k}^A$ and $F_{x_k}^{\bar{B}}$ can be efficiently evaluated for all atoms x_k, $k = 1, .., N$. In principle we just use the cut off technique mentioned in the preceding section and apply it to Brenner's potential. However, due to it's quite complicated structure, an efficient implementation is not straightforward.

In the following, to get rid of unnecessary notational overhead, we will denote by F_x the force acting on one given atom x. First of all, the term F_x^R belongs just to a pair potential and depends only on the distance between atoms. Therefore, using the identity $-\nabla_{x_i} V(r_{ij}) = \nabla_{x_j} V(r_{ij})$ and symmetry, we can evaluate it for all atoms by just following the interaction path of the standard linked-cell scheme, see Figure 2 (right).

Also the second term F_x^A can in principle be computed in this manner. $\nabla_{x_k} V_A(r_{kj})$ belongs also just to a pair potential and is symmetric. Furthermore, $\bar{B}_{ij}$ is symmetric as well, i.e. $\bar{B}_{ij} = \bar{B}_{ji}$. However, to evaluate $\bar{B}_{ij}$ we need $N_i^{(t)}, N_j^{(t)}$ and N_{ij}^{conj}. We have to compute them in advance. This can be realized in a simple linked-cell loop. Then we are able to compute $\bar{B}_{ij}$. However since $\bar{B}_{ij}$ depends on B_{ij}, we face the following problem: To evaluate B_{ij} we have to sum over all particles $k \neq (i,j)$. Note that the sum involves the function $f_{ik}(r_{ik})$. Now, since this function is cut off for all $r_{ik} > R_{ik}^{(2)}$, we need only to sum over all neighbors of x_i. However, because of the angle term G_θ we cannot achieve this with the short interaction path, due to missing symmetries. As a remedy, we now have to sum over all neighboring particles according to a long interaction path as given in Figure 4 (left). Furthermore, the evaluation of B_{ji} can be computed in an analogous way, but now f_{jk} is involved. This means that we have to sum over neighbors of particle x_j, see Figure 4 (right).

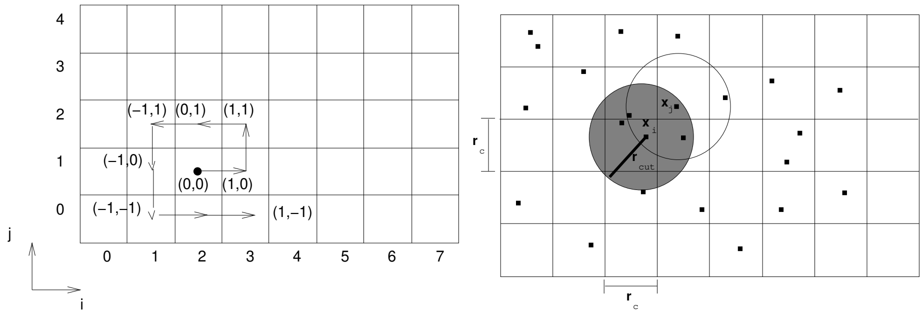

Figure 4. Left: Long interaction path of linked-cell loop in 2D. Right: Second cut-off area around particle x_j in 2D.

It remains to compute the term $F_x^{\bar{B}}$ for all atoms x. This is the most involved step of our algorithm. To this end, we review equation (24). We have

$$F_x^{\bar{B}} = \sum_{\substack{i,j \\ i<j}} (\nabla_x \bar{B}_{ij}) V_A(r_{ij}). \tag{26}$$

Now, we just have to sum over all pairs (i,j) with $(i<j)$. Therefore, we no more need to make use of symmetries, the computations can be done by a linked-cell loop with the standard short interaction path (see Figure 2 (right)). We will refer to this loop as (i,j)-loop in the following.

In addition, we have to take care of the term $(\nabla_x \bar{B}_{ij})$. It reads

$$\begin{aligned} \nabla_x \bar{B}_{ij} &= \frac{1}{2}(\nabla_x B_{ij} + \nabla_x B_{ji}) \\ &\quad + \nabla K_{ij}(N_i^{(t)}, N_j^{(t)}, N_{ij}^{conj}) \cdot (\nabla_x N_i^{(t)}, \nabla_x N_j^{(t)}, \nabla_x N_{ij}^{conj})^T \end{aligned} \tag{27}$$

and depends on B_{ij}. Since we encounter here (compare (20)) a sum over $k \neq i,j$, we get force contributions from $\nabla_x \bar{B}_{ij}$ also in the case that $x \neq x_i, x_j$. Now, we first perform the linked-cell loop over all pairs (i,j) with $(i<j)$. During the loop we get fixed interaction pairs (x_i, x_j) for which we can compute $V_A(r_{ij})$. Then we have to evaluate $\nabla_x \bar{B}_{ij}$ by calling a subroutine. This subroutine needs, beside other parts, the derivative of N_{ij}^{conj}. It reads

$$\begin{aligned} \nabla_x N_{ij}^{conj} = & \sum_{k(=carb)(\neq i,j)} \Big[f'_{ik}(r_{ik}) \nabla_x r_{ik} F(z_{ik}) \\ & + f_{ik}(r_{ik}) F'(z_{ik}) (\nabla_x N_k^{(t)} - f'_{ik}(r_{ik}) \nabla_x r_{ik}) \Big] \\ & + \sum_{k(=carb)(\neq i,j)} \Big[f'_{jk}(r_{jk}) \nabla_x r_{jk} F(z_{jk}) \\ & + f_{jk}(r_{jk}) F'(z_{jk}) (\nabla_x N_k^{(t)} - f'_{jk}(r_{jk}) \nabla_x r_{jk}) \Big] \end{aligned} \tag{28}$$

and, consequently, involves $\nabla_x N_k^{(t)}$. Obviously, with

$$\nabla_x N_k^{(t)} = \sum_{\substack{l(=carb)\\ l\neq k}} f'_{kl}(r_{kl})\nabla_x r_{kl} + \sum_{\substack{l(=hydr)\\ l\neq k}} f'_{kl}(r_{kl})\nabla_x r_{kl}, \tag{29}$$

the above terms contribute to the force vector of each particle only if they don't vanish identically. It is this trivial fact on which our algorithm is based.

Thus, if we reinterprete the equations (27), (28) and (29) as one long sum, we will recognize that each term in this sum contains a term of the type $\nabla_x r_{\tilde{i}\tilde{j}}$ for some fixed index pair $\tilde{i}, \tilde{j}$. This term is obviously non-zero only in the case of $x = x_{\tilde{i}}$ or $x = x_{\tilde{j}}$. Hence, we split the above sum into partial sums, containing only $\nabla_x r_{\tilde{i}\tilde{j}}$-terms with the same indices $(\tilde{i}, \tilde{j})$.

Now, recall that we are still within the (i, j)-loop associated to (26). During the (i, j)-loop, the contributions to the force are added to the force vector F_i and F_j of the corresponding atoms. Here, again we can make use of symmetries. However, we additionally have to perform summations over particles x_k with $k \neq (i, j)$. This can be implemented analogously to earlier cases by following a second local interaction path over particle x_i or x_j. This is due to the fact that the interactions are cut off by the function $f_{ik}(r_{ik})$ or $f_{jk}(r_{jk})$, respectively. Obviously this second local interaction path is a long interaction path as depicted in Figure 4 (left).

Altogether, in this way, an efficient $O(N)$ implementation of the force evaluation can be obtained. Further necessary details are given in the following subsection.

3.3 The final algorithm

In the following, we present the resulting algorithm in detail. To this end, we need the following definitions:

$$(DK_x, DK_y, DK_z)^T := \nabla K_{ij}(N_i^{(t)}, N_j^{(t)}, N_{ij}^{conj})^T \tag{30}$$

$$\begin{aligned}
A_{ijk} &:= V_A(r_{ij})DK_x \\
&\quad \cdot \left[f'_{ik_{(=carb)}}(r_{ik_{(=carb)}})\nabla_x r_{ik_{(=carb)}} + f'_{ik_{(=hydr)}}(r_{ik_{(=hydr)}})\nabla_x r_{ik_{(=hydr)}}\right] \\
B_{ijk} &:= V_A(r_{ij})DK_y \\
&\quad \cdot \left[f'_{jk_{(=carb)}}(r_{jk_{(=carb)}})\nabla_x r_{jk_{(=carb)}} + f'_{jk_{(=hydr)}}(r_{jk_{(=hydr)}})\nabla_x r_{jk_{(=hydr)}}\right] \\
C_{ijk} &:= V_A(r_{ij})DK_z \cdot [f'_{ik}(r_{ik})\nabla_x r_{ik} F(z_{ik}) - f_{ik}(r_{ik})F'(z_{ik})f'_{ik}(r_{ik})\nabla_x r_{ik}] \\
D_{ijk} &:= V_A(r_{ij})DK_z \cdot [f'_{jk}(r_{jk})\nabla_x r_{jk} F(z_{jk}) - f_{jk}(r_{jk})F'(z_{jk})f'_{jk}(r_{jk})\nabla_x r_{jk}] \\
E_{ijk} &:= V_A(r_{ij})DK_z \cdot f_{ik}(r_{ik})F'(z_{ik}) \\
F_{ijk} &:= V_A(r_{ij})DK_z \cdot f_{jk}(r_{jk})F'(z_{jk}) \\
G_{kl} &:= f'_{kl_{(=carb)}}(r_{kl_{(=carb)}})\nabla_x r_{kl_{(=carb)}} + f'_{kl_{(=hydr)}}(r_{kl_{(=hydr)}})\nabla_x r_{kl_{(=hydr)}}
\end{aligned} \tag{31}$$

With help of these expressions, we are able to sketch our algorithm as follows:

1. LC-loop:
   - Evaluate $N_i^{(H)}$ and $N_i^{(C)}$
2. LC-loop:
   - Evaluate $N_i^{conj} = \sum_{\substack{k(=carb)\\ k\neq i}} f_{ik}(r_{ik})F(z_{ik})$
3. LC-loop [(i,j)-loop]:
   - compute F_x^R
   - compute F_x^A:
     - compute:
       $N_{ij}^{conj} = 1 + N_i^{conj} - f_{ij}(r_{ij})F(z_{ij}) + N_j^{conj} - f_{ji}(r_{ji})F(z_{ji})$
     - compute $\bar{B}_{ij}$ with long local interaction path around x_i (see Figure 4 (left))
     - compute $\bar{B}_{ji}$ with long local interaction path around x_j (see Figure 4 (left))
   - compute $F_x^{\bar{B}}$:
     (a) compute $V_A(r_{ij})\nabla_x B_{ij}$ and $V_A(r_{ij})\nabla_x B_{ji}$
     (b) add force contributions
     (c) perform local loop over k according to long interaction path around x_i:
       - If $(k \neq i)$ compute A_{ijk}
       - $F_i := F_i + A_{ijk}$
       - $F_k := F_k - A_{ijk}$
       - If $((k \neq (i,j))$ and $(k = carb))$ compute C_{ijk}
       - $F_i := F_i + C_{ijk}$
       - $F_k := F_k - C_{ijk}$
       - If $((k \neq (i,j))$ and $(k = carb))$ compute E_{ijk}
       - $tmp(x_k) := tmp(x_k) + E_{ijk}$
         ($tmp(x_k)$ is a temporary variable associated to x_k)
     (d) perform local loop over k according to long interaction path around x_j:
       - If $(k \neq j)$ compute B_{ijk}
       - $F_j := F_j + B_{ijk}$
       - $F_k := F_k - B_{ijk}$
       - If $((k \neq (i,j))$ and $(k = carb))$ compute D_{ijk}
       - $F_j := F_j + D_{ijk}$
       - $F_k := F_k - D_{ijk}$
       - If $((k \neq (i,j))$ and $(k = carb))$ compute F_{ijk}
       - $tmp(x_k) := tmp(x_k) + F_{ijk}$
4. LC-loop [(k,l)-loop]:
   - compute expression G_{kl}
   - If $tmp(x_k) \neq 0$ then
     - $F_k := F_k + tmp(x_k) \cdot G_{kl}$
     - $F_l := F_l - tmp(x_k) \cdot G_{kl}$

Here *LC-loop* denotes one linked-cell loop over all atoms as already described in section 2. The implementation uses again the linked cell data storage format as shown in Figure 3. The extension of this algorithm to the additional computation of the energy is straightforward.

The computations in Step 3 (c),(d) follow from the definitions in (31) and the following reinterpretation of equation (26):

$$\begin{aligned} F_x^{\bar{B}} = \sum_{\substack{i,j \\ i<j}} \Big[& \frac{1}{2}(V_A(r_{ij})\nabla_x B_{ij} + V_A(r_{ij})\nabla_x B_{ji}) \\ & + \sum_{k\neq i} A_{ijk} + \sum_{k\neq j} B_{ijk} + \sum_{\substack{k(\neq i,j) \\ (k=carb)}} \left(C_{ijk} + D_{ijk} + E_{ijk}\nabla_x N_k^{(t)} + F_{ijk}\nabla_x N_k^{(t)} \right) \Big] \end{aligned} \tag{32}$$

Note that the expressions $A_{ijk}, C_{ijk}, E_{ijk}$ are cut off by means of the function f_{ik} and, consequently, represent local sums over x_i. Analogously, $B_{ijk}, D_{ijk}, F_{ijk}$ represent local sums over x_j. Recall from the end of section 3.2 that many contributions in the partial sums are zero (case $x \neq x_{\bar{i}}$, $x \neq x_{\bar{j}}$). Therefore the expressions A_{ijk} and C_{ijk} contribute force just to atom x_i and atom x_k, the expressions B_{ijk} and D_{ijk} contribute force to x_j and x_k, etc. The terms E_{ijk} and F_{ijk} are computed in a similar manner. But in contrast to the evaluation of A_{ijk}, etc., we now cannot add the force contributions directly to the corresponding atoms. The reason is that we have to multiply the respective expressions with $\nabla_x N_k^{(t)}$ (see (32)). Now, since the ad hoc evaluation of $\nabla_x N_k^{(t)}$ would result in a further loop inside of the current (i,j)-loop, we would obtain a substantial amount of unnecessary additional time and storage complexity. Furthermore, since the sum over k will be computed for every (i,j) we would have to evaluate the same $\nabla_x N_k^{(t)}$ very often. Alternatively, the computation of $\nabla_x N_k^{(t)}$ could be done in advance. However, since we need the gradient in every direction for every atom, this approach would result in $O(N^2)$ computations and storage complexity and is therefore also not feasible.

This problem can be overcome by the following method: We introduce an additional variable (*tmp*) for each atom. Now we are able to add the values of the expressions E_{ijk} and F_{ijk} to the *tmp*-variable of atom x_k. Then, we quit the current (i,j)-loop, start a new linked-cell loop, which we call (k,l)-loop (see Step 4) and evaluate the expression G_{kl}. This results in the computation of $\nabla_x N_k^{(t)}$ since

$$\nabla_x N_k^{(t)} = \sum_{l\neq k} G_{kl}. \tag{33}$$

Finally we perform the necessary summations onto the force variables according to equation (32).

4 Parallelization

For the parallelization of our algorithm we follow the classical domain decomposition approach. To this end, we assume that we have a decomposition of the simulation domain Ω in $N_x \times N_y \times N_z$ cells from the linked cell approach as already described in section 2. Now we decompose Ω into $i_p \times j_p \times k_p$ subdomains whose boundaries coincide with the boundaries of certain cells. A two-dimensional example of a geometric decomposition of a rectangular domain Ω in twenty subdomains Ω_{11} to Ω_{54} is depicted in Figure 5.

Note that by the specific choice of the parameters i_p, j_p, k_p we can generate different types of decompositions of the domain, i.e. cube-like ($i_p = j_p = k_p$), column-type ($i_p = j_p, k_p = 1$) or slice type ($j_p = k_p = 1$) subdomains. Now, each subdomain and its associated data is assigned to a processor. Each processor stores its subdomain in a local linked list data structure. In principle, each processor can now work in parallel. However special care has to be taken for the intermediate communication of necessary data. To this end, as usual in domain decomposition methods, an additional layer of boundary cells is stored (ghost cells). It overlaps the layer of cells of the neighboring subdomains adjacent to the boundary, see Figure 7. Altogether, the different processors now can compute their part of the force evaluation fully in parallel. Then communication of data is necessary to update the values of the particle data situated in these boundary ghost cells. Additional communication is necessary in each time step after moving the particles. Here, particles can have moved out of a subdomain into a neighboring one. Such communication of data is of course not performed cell by cell but collectively for all cells belonging to one side/face of each subdomain in one communication step. The exchange of data is done in a synchronized, bi-directional fashion, parallel between subdomains in d consecutive steps, one step for each coordinate direction. For details in the two-dimensional case, see Figure 6.

Thus, our approach directly follows the well known techniques from conventional domain decomposition parallelization. For further details, compare [12,13]. The modifications necessary for taking advantage of the symmetry due to Newton's third law (10) and the short interaction path (see Figure 7) for the linked cell method are obvious. Also, the modifications necessary due to Brenner's potential are straightforward.

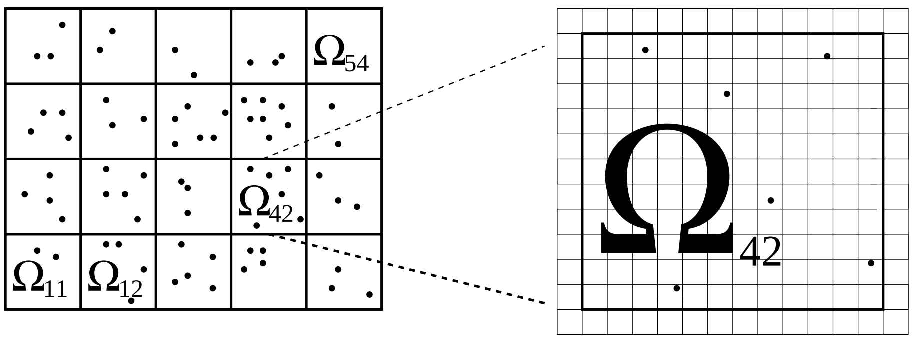

Figure 5. Decomposition of the subdomains in cells as in the linked-cell method, described for subdomain Ω_{42}.

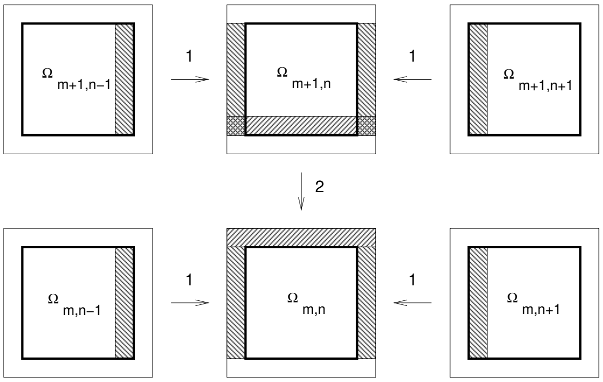

Figure 6. The exchange of particle data is performed in two steps (2D example). Here only communication steps in direction of subdomain $\Omega_{m,n}$ are depicted.

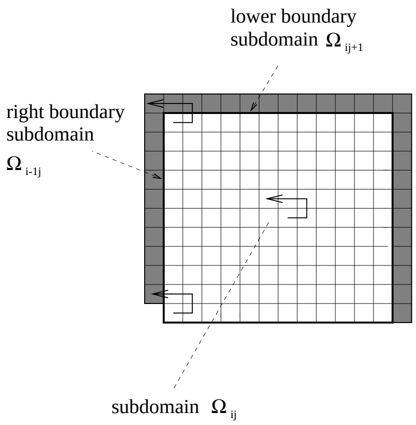

Figure 7. Data exchange: subdomain Ω_{ij} needs the coordinates and particle data from the grey shaded cells of neighboring subdomains. This is induced by the interaction path according to Figure 2 (right).

5 Numerical experiments

We have conducted several experiments with our parallel code. Since the Brenner potential models bond breaking and forming very well, we examined reactive collisions first. Here we considered various cases of collisions of benzyne and C_{60} fullerenes. We also studied the behavior of a large amount of buckyballs ordered either in a crystal lattice or moving freely like in a liquid. Furthermore, we examined bending and stretching properties of $(7,0)$-, $(9,0)$- and $(14,0)$-zigzag carbon nanotubes and simulated their stress/strain behavior. Here we run experiments on our PC cluster Parnass2 and on the Cray T3E-900/600 computer with up to 512 processors where more than 100 million carbon atoms were involved. Our experiments showed a Young's modulus of nearly one Tera Pascal for nanotubes under tension which demonstrates them to be nearly as strong as diamond.

5.1 Reactive collision of buckyballs with benzyne

First we consider the reactive impact of benzyne onto a C_{60} fullerene (buckyball). For similar experiments, compare [14]. Here we accelerate a benzyne molecule (C_6H_4) and let it move towards a C_{60}-fullerene. Our experiments showed that reactive collisions appear in certain cases. Three reaction types could be distinguished. Here, depending on the collision velocity, we observe either a repulsion (elastic collision) of the benzyne molecule, a reaction of both molecules, or a dissociation process. The observed ranges of the collision velocity for the three different cases obtained from our experiments were:

- bouncing phase: $14.17 - 28.34$ [Å/ps],
- reactive phase: $35.43 - 99.20$ [Å/ps],
- dissociation of benzyne/buckyball: ≥ 106.29 [Å/ps].

The simulations were based on the parameter list of Table 7 (see Appendix) for Brenner's potential combined with a Lennard-Jones (12-6) potential for the intermolecular interactions [15]. The simulations were all carried out for 1 ps with an initial temperature of 316 K and a time step of 0.035 [fs]. In Figure (8) a few snapshots from a simulation of the bouncing phase are shown. Further results can be obtained from http://wissrech.iam.uni-bonn.de/research/projects/caglar/nano/nano.html.

5.2 Nanotube Simulations

Nanotubes are large linear fullerenes – close-caged molecules containing only hexagonal and pentagonal faces. A single walled carbon (n, m)-nanotube is a rolled-up sheet of graphene. The (n, m) integer indices correspond to the chiral vector along which the graphene sheet has been rolled up. Here, the integer values (n, m) are given corresponding to the basis vectors $(\vec{a}, \vec{b})$ as shown in Figure 9. We distinguish between three different types of single-walled carbon nanotubes: armchair, zigzag and chiral. Their form depends on the chiral angle θ between the roll-up vector (n, m) and the basis direction $\vec{a}$ (see Figure 9). If the angle is zero we obtain the zigzag form, in the case of $\theta = 30°$ we have armchair nanotubes and for angles somewhere between the angles of zigzag ($\theta = 0°$) and armchair form ($\theta = 30°$) we obtain the so called chiral nanotubes. The tubule diameter $diam$ and the chiral angle θ are uniquely determined by the defining roll-up vector (n, m). We have $diam = 0.078\sqrt{n^2 + nm + m^2}$ nanometers and $\theta = \arctan[\sqrt{3}m/(m + 2n)]$, see [4].

In our simulations we performed various stretching and bending experiments. These experiments can help to examine material properties of nanotubes and their behavior under stress. To this end, we define the stress σ as the force F normalized by the cross-sectional area A (in our case $A = \pi/4 \cdot diam^2$) of a material, i.e.

$$\sigma = F/A. \tag{34}$$

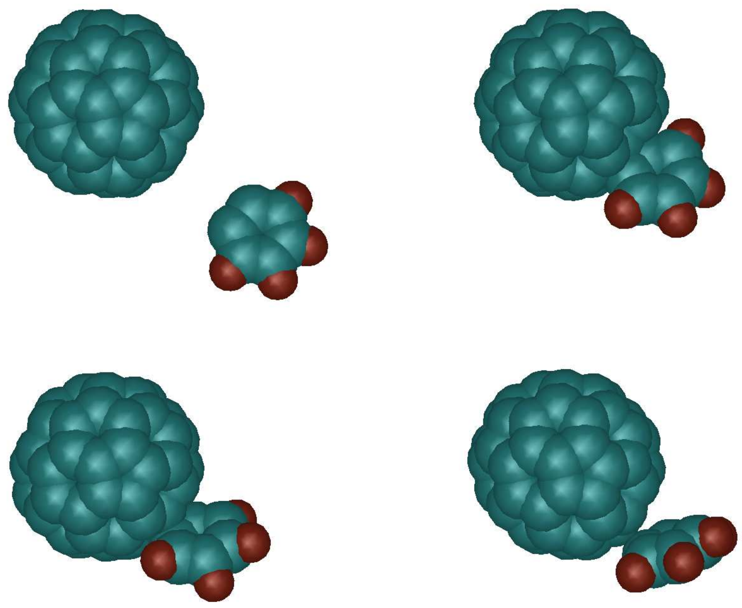

Figure 8. Buckyballs and benzyne: snapshots of a sequence of configurations. The benzyne molecule is accelerated towards the C_{60} molecule. Benzyne bounces elastically back.

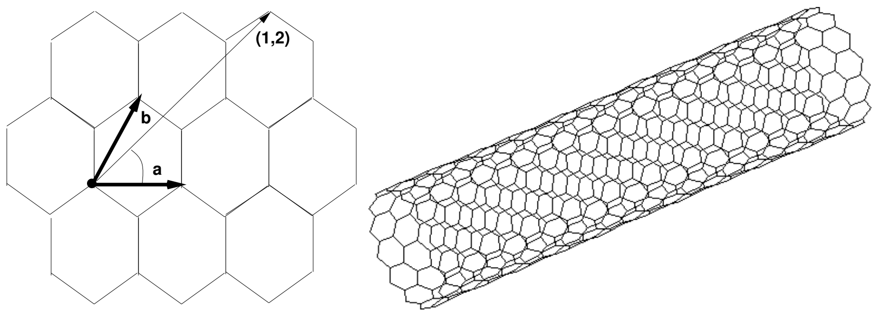

Figure 9. Left: A strip of a graphite sheet is depicted. Here, the basis $(\vec{a}, \vec{b})$ is given and an example of a roll-up vector with $(n, m) = (1, 2)$ is shown. The corresponding chiral angle θ is the angle between the vector $(1, 2)$ and the $\vec{a}$ direction. Right: Example of the structure of a carbon nanotube.

In our MD-simulations, the stress-tensor can be measured according to the following formula:

$$\sigma^{\alpha\beta} = \frac{1}{vol} \sum_{i=1}^{N} \left\{ m_i u_i^{\alpha} u_i^{\beta} + \sum_{j=1, j \neq i}^{N} \frac{1}{2} \vec{F}_{ij}^{\alpha} \vec{r}_{ij}^{\beta} \right\}, \quad \alpha, \beta \in \{1, 2, 3\}, \quad \vec{r}_{ij} := x_i - x_j, \tag{35}$$

where vol denotes the volume of the nanotube. Now, if we would for example pull our sample apart, we could measure the change in length. Thus, we define the strain ϵ as the change in length of the fiber normalized by the initial length, i.e.

$$\epsilon = (l_1 - l_0)/l_0. \tag{36}$$

If we plot finally the uniaxial component of the stress versus strain, then, the slope of this line gives the corresponding tensile elastic modulus or Young's modulus E of the material, i.e. $\sigma = E\epsilon$, see Figure 10 (right). Loosely speaking, E gives the force needed to elongate the nanotube, see Figure 10 (left). Note that this measurement technique only applies to small forces which do not stretch the material irreversibly, i.e. the material must be in the elastic regime. Note furthermore that for example Young's modulus of diamond is 1000 GPa.

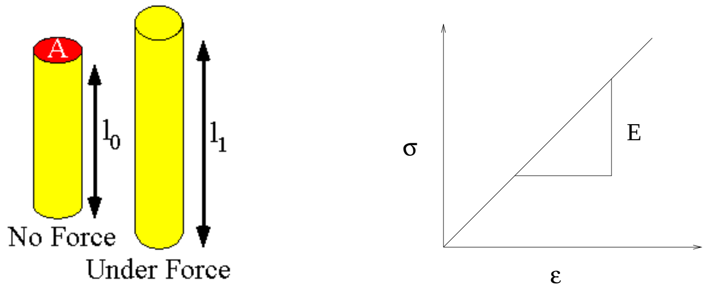

Figure 10. A tensile test (left), a stress-strain curve with Young's modulus E (right)

First, we carried out a small simulation run with a $(7, 0)$-zigzag nanotube consisting of 308 carbon atoms. Here and in the following we used the parameters of Table 7 (see Appendix). We stretched the nanotube by pulling at both ends with a constant force of 3.2 nN. Snapshots of the process at different time steps are given in Figure 11. As we can see the nanotube bursts slowly. In Figure 12 (left) the corresponding yield strength diagram is shown. Fracture occurred at a strain rate of about 35%. Furthermore, our measurements resulted in a Young's modulus of about 1 TPa. Now we considered the same configuration again, but we pulled with a slightly lower force (2.8 nN). Then, the tube did not burst. Figure 12 (right) gives the associated stress strain curve. Here, the maximum strain rate is 30%. We clearly see the elastic behavior of the deformations.

In the following, we considered the same configuration again and pulled the tube at both ends with constant force 2.8 nN in every time step up to 0.176 ps. Then, in contrast to the preceding experiments, we relaxed the tube. Figure 13 (left) gives the corresponding stress strain curve. Of course, due to the limited time of tensile

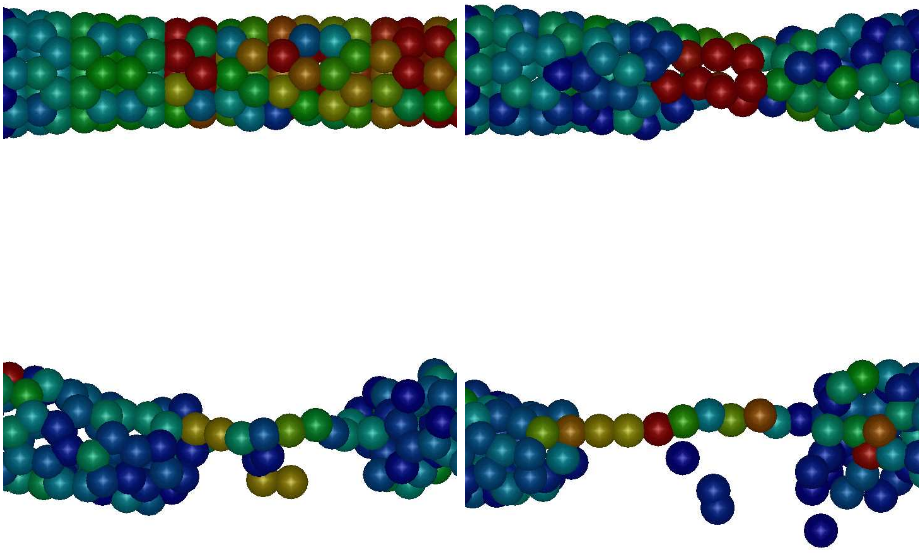

Figure 11. Stretching of a nanotube with constant force of 3.2 nN

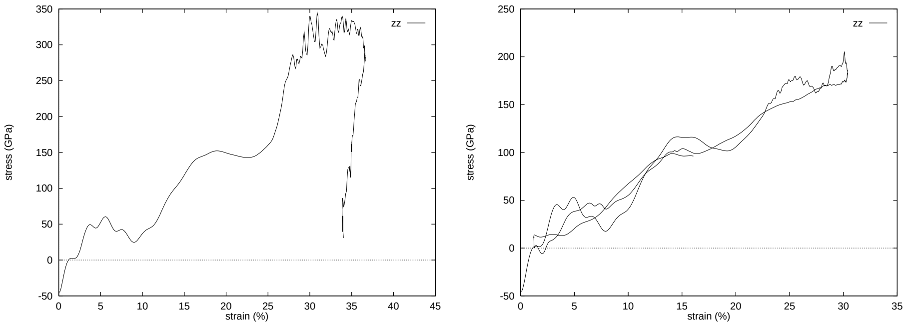

Figure 12. Stress strain behavior of carbon nanotubes under tensile load. Left: bursting tube case. Here we pull with 3.2 nN constant force. Right: non-bursting case. Here, we apply 2.8 nN constant force.

loading, we only achieved a reduced maximum strain rate about 10%. Furthermore, we observe that the tube oscillates around its initial form. If we increase the time of tensile loading to 0.35 ps, we obtain the results given in Figure 13 (right)). Here, we see that a maximum strain rate of about 26% is achieved which is nearly the same as in the previous experiment of Figure 12 (right). However, since we relax the tube at a later time, it shows not more the oscillating behavior of the preceding experiment, but now it seems to undergo a buckling deformation.

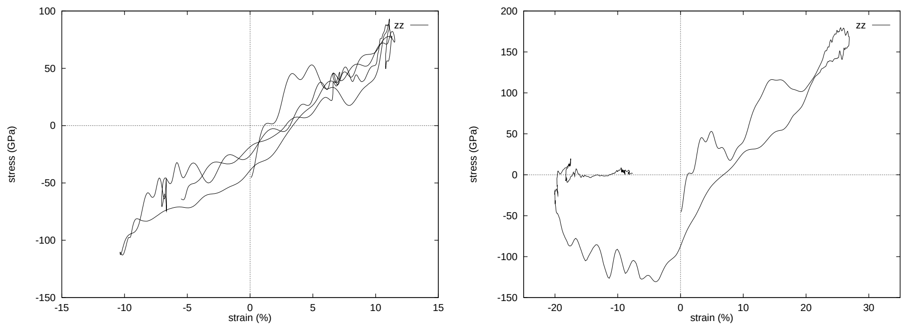

Figure 13. Stress strain behavior of non bursting tubes. Both tubes are pulled with constant force 2.8 nN. Left: Relaxation after 0.176 ps. Right: Relaxation after 0.35 ps.

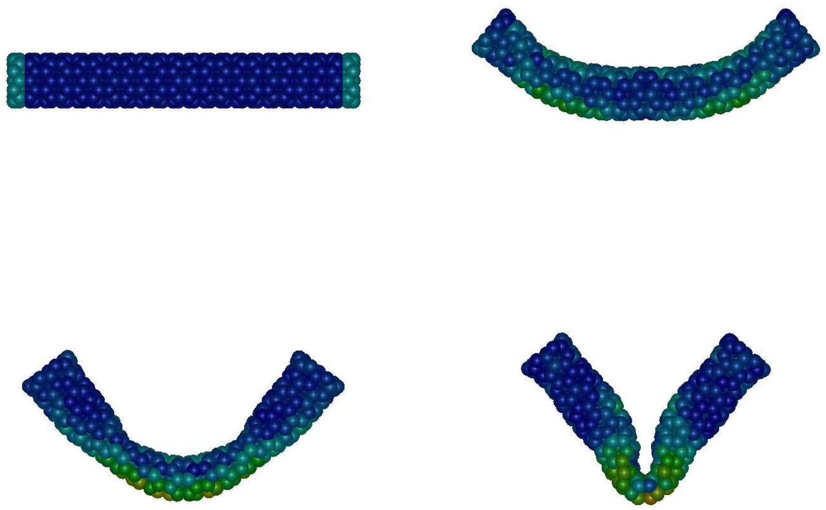

Figure 14. Bending of a nanotube

Finally we performed experiments where we bend the nanotube. Here we apply load at 45 degree at both ends of the tube. Figure 14 illustrates the results. It is interesting that we observed no fracture of the nanotube even for cases of extreme bending. Moreover after relaxation by removing the external constraint, an almost original strain free structure was regained. For similar experiments see also [16].

5.3 Parallelization results

We now discuss the properties of our parallelized algorithm. Here, we stick to the case of tensile load on nanotubes with different length. We run our code on Parnass2, our cluster of Intel Pentium II PCs (400 MHz, see also http://wissrech.iam.uni-bonn.de/research/projects/parnass2) and on the Cray T3E with nearly 100 Mio. atoms. Note that the parameters i_p, j_p and k_p were chosen in such a way that an appropriate, nearly slice type static domain decomposition resulted in our large scale nanotube simulations.

Table 2 gives the measured run times for one time step of the algorithm on Parnass2. Associated speed up figures and efficiencies are shown in Table 3.

time	processors								
particles	1	2	4	8	16	32	64	96	126
10024	**13.16**	6.56	3.34	1.67	0.85	0.44	0.22	0.17	0.17
20020	26.3	**13.16**	6.72	3.34	1.68	0.85	0.44	0.33	0.28
40012	52.84	26.27	**13.4**	6.67	3.34	1.69	0.84	0.59	0.48
80024	**105.3**	52.6	26.8	**13.4**	6.69	3.34	1.68	1.15	0.90
160020	210.6	**105.2**	53.6	26.8	**13.4**	6.70	3.34	2.23	1.72
320012	423.0	210.4	**107.3**	53.6	26.8	**13.4**	6.71	4.48	3.45
640024	846.7	421.3	214.7	**107.3**	53.7	26.8	**13.4**	8.90	6.83
1320984				217.1	**108.8**	54.5	27.2	18.2	**13.9**
2641968					218.3	**109.1**	54.5	36.4	27.6
5283936						218.2	**108.9**	72.7	55.2
10567872							218.3	144.8	**110.3**

Table 2. Parallel execution times (in seconds) for one time-step of the simulation of nanotubes, Parnass2. Scale up is depicted by bold figures.

	processors								
	1	2	4	8	16	32	64	96	126
Speed up	1.00	2.00	3.93	7.86	15.72	31.43	63.05	94.44	122.44
Efficiency	1.00	1.00	0.98	0.98	0.98	0.98	0.99	0.98	0.97

Table 3. Parallel speed up and parallel efficiency for one time-step of the simulation of nanotubes with 160020 atoms, Parnass2.

Table 4 gives the run times on the CrayT3E. The associated speed up properties and efficiencies are shown in Table 5. Here we used two different Cray machines, the T3E-900 and the T3E-600 from Jülich's NIC. Note that the T3E-600 possesses 512 DEC Alpha processors which are running with 300 MHz, whereas the T3E-900 possesses only 256 DEC Alpha processors which are running at a faster clock rate

(450 MHz). This explains the difference of about a factor of 1.5 in the measured run times of Table 4 when switching from one machine to the other.

time	processors										
	CRAY T3E-900							CRAY T3E-600			
particles	1	2	4	8	16	32	64	65	128	256	512
10024	15.3	7.63	3.89	1.95	0.99	0.52					
20020	30.1	15.2	7.81	3.89	1.95	0.99	0.52	0.75			
40012	**60.2**	30.4	15.6	7.75	3.89	1.95	1.00	1.35	0.73		
80024	120.9	**60.8**	31.1	15.6	7.75	3.90	1.97	2.74	1.45	0.73	0.38
160020	240.6	122.4	**62.0**	31.0	15.5	7.77	3.89	5.57	2.86	1.44	0.73
320012				**62.0**	31.0	15.5	7.77	11.1	5.66	2.86	1.44
640024					**62.1**	31.0	15.5	22.4	11.3	5.66	2.86
1320984						**62.9**	31.5	45.3	22.9	11.5	5.74
2641968							**62.9**	90.6	46.1	23.0	11.4
5283936								**182.2**	91.4	45.7	22.9
10567872								363	**185.3**	91.4	45.6
21135744									364.4	**183.7**	91.4
29061648										251.0	**125.3**
59994984											258
99991640											430

Table 4. Execution time for one time step in seconds with up to 64 processors on the CRAY T3E-900 and with up to 512 processors on the CRAY T3E-600. Scale up is depicted by bold figures.

	processors						
	1	2	4	8	16	32	64
Speed up	1.00	1.97	3.88	7.76	15.52	30.97	61.85
Efficiency	1.00	0.99	0.97	0.97	0.97	0.97	0.97

Table 5. Parallel speed up and parallel efficiency for one time-step of the simulation of nanotubes with 160020 atoms, Cray T3E-900.

We clearly see the following: First, our algorithm shows a nearly linear speed up. A doubling of the number of processors results in about a halfing of the run time. Also our algorithm scales perfectly. If we increase the number of atoms by a factor of two but additionally double the number of processors, we obtain basically the same absolute run times. Furthermore this behavior can be observed with up to 512 processors on the T3E.

In summary, we observed for our algorithm very good speed up properties and efficiencies, and, moreover, a very good scale up behavior on both types of parallel computing systems.

6 Concluding remarks

We considered the molecular dynamics simulation of fullerenes and carbon nanotubes. The approach is based on Brenner's bond-order potential. Here we implemented a variant of the linked-cell technique. This results in an efficient force evaluation of Brenner's potential with $O(N)$ time complexity. The algorithm is parallelized by a classical domain decomposition approach.

We applied the algorithm to several impact experiments with C_{60} fullerenes. They showed a very elastic performance of the fullerenes. In addition we examined the elastomechanic response behavior of single walled carbon nanotubes. Here we measured a Young's modulus of about one Tera Pascal. The remarkable result is that nanotubes under tension are nearly as strong as diamond. Furthermore we showed parallelization results on a Cray and Parnass2. The algorithm exhibits very good scale up properties. We were able to run simulations with nearly 100 million atoms on 512 processors.

Acknowledgments

The authors would like to acknowledge that part of this work was done on the CRAY T3E at the *Forschungszentrum Jülich, NIC.*

Appendix.

Carbon		Hydrogen		Hydrocarbons	
$R_{CC}^{(e)}$	1.315 Å	$R_{HH}^{(e)}$	0.74144 Å	$R_{CH}^{(e)}$	1.1199 Å
$D_{CC}^{(e)}$	6.325 eV	$D_{HH}^{(e)}$	4.7509 eV	$D_{CH}^{(e)}$	3.6422 eV
β_{CC}	1.5 Å^{-1}	β_{HH}	1.9436 Å^{-1}	β_{CH}	1.9583 Å^{-1}
S_{CC}	1.29	S_{HH}	2.3432	S_{CH}	1.7386
δ_{CC}	0.80469	δ_{HH}	0.80469	δ_{CH}	0.0
$R_{CC}^{(1)}$	1.7 Å	$R_{HH}^{(1)}$	1.1 Å	$R_{CH}^{(1)}$	1.3 Å
$R_{CC}^{(2)}$	2.0 Å	$R_{HH}^{(2)}$	1.7 Å	$R_{CH}^{(2)}$	1.8 Å
α_{CCC}	0.0	α_{HHH}	3.0	$\alpha_{HHC}, \alpha_{CHH}$	3.0 Å^{-1}
				$\alpha_{HCH}, \alpha_{HCC}$	
a_0	0.011304	-	-	-	-
c_0^2	19^2	-	-	-	-
d_0^2	2.5^2	-	-	-	-
-	-	G_{HH}	4.0	-	-
-	-	-	-	$H_{CC}(1,1)$	-0.0175
-	-	-	-	$H_{CC}(2,0)$	-0.0070
-	-	-	-	$H_{CC}(3,0)$	0.0119
-	-	-	-	$H_{CC}(1,2)$	0.0115
-	-	-	-	$H_{CC}(2,1)$	0.0118
-	-	-	-	$H_{CH}(1,0)$	-0.0760
-	-	-	-	$H_{CH}(2,0)$	-0.2163
-	-	-	-	$H_{CH}(3,0)$	-0.3375
-	-	-	-	$H_{CH}(0,1)$	-0.1792
-	-	-	-	$H_{CH}(0,2)$	-0.2407
-	-	-	-	$H_{CH}(1,1)$	-0.2477
-	-	-	-	$H_{CH}(2,1)$	-0.3320
-	-	-	-	$H_{CH}(0,3)$	-0.3323
-	-	-	-	$H_{CH}(1,2)$	-0.3321
-	-	-	-	$\frac{\partial H_{CH}(1,1)}{\partial C}$	-0.12805
-	-	-	-	$\frac{\partial H_{CH}(2,0)}{\partial C}$	-0.07655
-	-	-	-	$\frac{\partial H_{CH}(0,2)}{\partial H}$	-0.13075
-	-	-	-	$\frac{\partial H_{CH}(1,1)}{\partial H}$	-0.0764
$K(2,3,1), K(2,3,2)$	-0.0465	-	-	-	-
$K(1,2,2)$	-0.0355	-	-	-	-
-	-	-	-	$K(1,1,1)$	0.1511
-	-	-	-	$K(2,2,1)$	0.075
-	-	-	-	$K(1,2,1)$	0.0126
-	-	-	-	$K(1,3,1), K(1,3,2)$	-0.1130
-	-	-	-	$K(0,3,1), K(0,3,2)$	-0.1220
-	-	-	-	$K(0,2,2)$	-0.0445
-	-	-	-	$K(0,2,1)$	0.0320
-	-	-	-	$K(0,1,1)$	0.1100
-	-	-	-	$K(1,1,2)$	0.0074
-	-	-	-	$\frac{\partial K(3,1,1)}{\partial i}$	-0.1160
-	-	-	-	$\frac{\partial K(3,2,1)}{\partial i}$	-0.13205
-	-	-	-	$\frac{\partial K(3,1,2)}{\partial i}$	-0.0610
-	-	-	-	$\frac{\partial K(2,3,2)}{\partial i}$	0.02225
-	-	-	-	$\frac{\partial K(2,4,2)}{\partial i}$	-0.03775
-	-	-	-	$\frac{\partial K(3,4,2)}{\partial i}$	0.0565
-	-	-	-	$\frac{\partial K(3,4,1)}{\partial i}$	0.0565
-	-	-	-	$\frac{\partial K(3,2,2)}{\partial i}$	-0.1065

Table 6. Parameters for equations (12)-(22). All parameters not given are equal to zero, H_{ij} and K_{ij} are multidimensional cubic Hermite splines. $K(i,j,k) = K(j,i,k)$, $K(i,j,k>2) = K(i,j,2)$, and $\partial K(i,j,k)/\partial i = \partial K(j,i,k)/\partial i$. The partial derivatives are needed in the force evaluation (gradients). $G_C(\theta) = a_0\{1 + c_0^2/d_0^2 - c_0^2/[d_0^2 + (1+cos\theta)^2]\}$.

Carbon		Hydrogen		Hydrocarbons	
$R_{CC}^{(e)}$	1.39 Å	$R_{HH}^{(e)}$	0.74144 Å	$R_{CH}^{(e)}$	1.1199 Å
$D_{CC}^{(e)}$	6.0 eV	$D_{HH}^{(e)}$	4.7509 eV	$D_{CH}^{(e)}$	3.6422 eV
β_{CC}	2.1 Å^{-1}	β_{HH}	1.9436 Å^{-1}	β_{CH}	1.9583 Å^{-1}
S_{CC}	1.22	S_{HH}	2.3432	S_{CH}	1.69077
δ_{CC}	0.5	δ_{HH}	0.5	δ_{CH}	0.0
$R_{CC}^{(1)}$	1.7 Å	$R_{HH}^{(1)}$	1.1 Å	$R_{CH}^{(1)}$	1.3 Å
$R_{CC}^{(2)}$	2.0 Å	$R_{HH}^{(2)}$	1.7 Å	$R_{CH}^{(2)}$	1.8 Å
α_{CCC}	0.0	α_{HHH}	4.0	$\alpha_{HHC}, \alpha_{CHH}$ $\alpha_{HCH}, \alpha_{HCC}$	4.0 Å^{-1}
a_0	0.00020813	-	-	-	-
c_0^2	330^2	-	-	-	-
d_0^2	3.5^2	-	-	-	-
-	-	G_{HH}	12.33	-	-
-	-	-	-	$H_{CC}(1,1)$	-0.0226
-	-	-	-	$H_{CC}(2,0)$	-0.0061
-	-	-	-	$H_{CC}(3,0)$	0.0173
-	-	-	-	$H_{CC}(1,2)$	0.0149
-	-	-	-	$H_{CC}(2,1)$	0.0160
-	-	-	-	$H_{CH}(1,0)$	-0.0984
-	-	-	-	$H_{CH}(2,0)$	-0.2878
-	-	-	-	$H_{CH}(3,0)$	-0.4507
-	-	-	-	$H_{CH}(0,1)$	-0.2479
-	-	-	-	$H_{CH}(0,2)$	-0.3221
-	-	-	-	$H_{CH}(1,1)$	-0.3344
-	-	-	-	$H_{CH}(2,1)$	-0.4438
-	-	-	-	$H_{CH}(0,3)$	-0.4460
-	-	-	-	$H_{CH}(1,2)$	-0.4449
-	-	-	-	$\frac{\partial H_{CH}(1,1)}{\partial C}$	-0.17325
-	-	-	-	$\frac{\partial H_{CH}(2,0)}{\partial C}$	-0.09905
-	-	-	-	$\frac{\partial H_{CH}(0,2)}{\partial H}$	-0.17615
-	-	-	-	$\frac{\partial H_{CH}(1,1)}{\partial H}$	-0.09795
$K(2,3,1), K(2,3,2)$	-0.0363	-	-	-	-
$K(1,2,2)$	-0.0243	-	-	-	-
-	-	-	-	$K(1,1,1)$	0.1264
-	-	-	-	$K(2,2,1)$	0.0605
-	-	-	-	$K(1,2,1)$	0.0120
-	-	-	-	$K(1,3,1), K(1,3,2)$	-0.0903
-	-	-	-	$K(0,3,1), K(0,3,2)$	-0.0904
-	-	-	-	$K(0,2,2)$	-0.0269
-	-	-	-	$K(0,2,1)$	0.0427
-	-	-	-	$K(0,1,1)$	0.0996
-	-	-	-	$K(1,1,2)$	0.0108
-	-	-	-	$\frac{\partial K(3,1,1)}{\partial i}$	-0.0950
-	-	-	-	$\frac{\partial K(3,2,1)}{\partial i}$	-0.10835
-	-	-	-	$\frac{\partial K(3,1,2)}{\partial i}$	-0.0452
-	-	-	-	$\frac{\partial K(2,3,2)}{\partial i}$	0.01345
-	-	-	-	$\frac{\partial K(2,4,2)}{\partial i}$	-0.02705
-	-	-	-	$\frac{\partial K(3,4,2)}{\partial i}$	0.04515
-	-	-	-	$\frac{\partial K(3,4,1)}{\partial i}$	0.04515
-	-	-	-	$\frac{\partial K(3,2,2)}{\partial i}$	-0.08760

Table 7. Parameters for equations (12)-(22). All parameters not given are equal to zero, H_{ij} and K_{ij} are multidimensional cubic Hermite splines. $K(i,j,k) = K(j,i,k)$, $K(i,j,k>2) = K(i,j,2)$, and $\partial K(i,j,k)/\partial i = \partial K(j,i,k)/\partial i$. The partial derivatives are needed in the force evaluation (gradients). $G_C(\theta) = a_0\{1 + c_0^2/d_0^2 - c_0^2/[d_0^2 + (1+cos\theta)^2]\}$.

References

1. H. W. Kroto, J. R. Heath, O.B. SC, R. F. Curl, and R.E. Smalley. C60 Buckminsterfullerene. *Nature*, 318:162–163, 1985.
2. S. Iijima. Helical Microtubules of Graphitic Carbon. *Nature*, 354:56–58, 1991.
3. R. E. Smalley and D. T. Colbert. Self Assembly of Fullerene Tubes and Balls. talk to: Robert A. Welch Foundation, October 1995.
4. B. I. Yakobson and R. E. Smalley. Fullerene Nanotubes: $C_{1,000,000}$ and Beyond. *American Scientist*, 85(4):324–337, 1997.
5. D. H. Robertson, D. W. Brenner, and J. W. Mintmire. Energetics of Nanoscale Graphitic Tubules. *Phys. Rev. B*, 45:12592, 1992.
6. D. W. Brenner. Empirical Potential for Hydrocarbons for Use in Simulating the Chemical Vapor Deposition of Diamond Films. *Physical Review B*, 42(15):9458–9471, 1990.
7. G. C. Abell. Empirical Chemical Pseudopotential Theory of Molecular and Metallic Bonding. *Physical Review B*, 31(10):6184–6197, 1985.
8. J. Tersoff. New Empirical Model for the Structural Properties of Silicon. *Physical Review Letters*, 56(6):632–635, 1986.
9. D. Beeman. Some Multistep Methods for Use in Molecular Dynamics Calculations. *Journal of Computational Physics*, 20:130–139, 1976.
10. M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Oxford University Press, 1987. ISBN 0-19-855645-4.
11. R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles*. McGraw-Hill, New York, 1981.
12. Peter S. Lomdahl, Pablo Tamayo, Niels Grønbech-Jensen, and David M. Beazley. 50 GFlops Molecular Dynamics on the Connection Machine 5. *Association for Computing Machinery*, 1993. Los Alamos National Laboratory.
13. D.M. Beazley, P.S. Lomdahl, N. Gronbech-Jensen, R. Giles, and P. Tamayo. Parallel Algorithms for Short-Range Molecular Dynamics. In D. Stauffer, editor, *Annual Reviews of Computational Physics*, volume III, page 119. World Scientific, 1995.
14. J. Han, A. Globus, R. Jaffe, and G. Deardorff. Molecular Dynamics Simulations of Carbon Nanotube Based Gears. *Nanotechnology*, (8):103, 1997.
15. D. W. Brenner, D. H. Robertson, M. L. Elert, and C. T. White. Detonations at Nanometer Resolution Using Molecular dynamics. *Physical Review Letters*, 70(14):2174–2177, 1993.
16. D. Srivastava and S.T. Barnard. Molecular Dynamics Simulation of Large-Scale Carbon Nanotubes on a Shared-Memory Architecture. Technical report, NASA Ames Research Center, 1997.
17. G. D. Smith and R. L. Jaffe. Comparative Study of Force Fields for Benzene. *J. Phys. Chem.*, 100:9624–9630, 1996.
18. A. Nakano, M. E. Bachlechner, T. J. Campbell, R. K. Kalia, A. Omeltchenko, K. Tsuruta, and P. Vashishta. Atomistic Simulation of Nanostructured Materials. *IEEE Computational Science & Engineering*, pages 68–78, 1998.
19. L. A. Girifalco. Molecular Properties of C_{60} in the Gas and Solid Phases. *J. Phys. Chem.*, 96:858–861, 1992.