A Method to Obtain a Near-Minimal-Volume Molecular Simulation of a Macromolecule, Using Periodic Boundary Conditions and Rotational Constraints

HENK BEKKER,[1] JUR P. VAN DEN BERG,[1,*] TSJERK A. WASSENAAR[2]
[1]*Institute for Mathematics and Computing Science, University of Groningen, P.O.B. 800 9700 AV Groningen, The Netherlands*
[2]*Department of Biophysical Chemistry, Groningen Biomolecular Sciences and Biotechnology Institute (GBB), University of Groningen, Nijenborgh 4, 9747 AG, Groningen, The Netherlands*

Received 20 May 2003; Accepted 17 September 2003

Abstract: If the rotational motion of a single macromolecule is constrained during a molecular dynamics simulation with periodic boundary conditions it is possible to perform such simulations in a computational box with a minimal amount of solvent. In this article we describe a method to construct such a box, and test the approach on a number of macromolecules, randomly chosen from the protein databank. The essence of the method is that the molecule is first dilated with a layer of at least half the cut-off radius. For the enlarged molecule a near-densest lattice packing is calculated. From this packing the simulation box is derived. On average, the volume of the resulting box proves to be about 50% of the volume of standard boxes. In test simulations this yields on average a factor of about two in simulation speed.

 J Comput Chem 25: 1037–1046, 2004

Key words: molecular simulation; box shape; minimal volume; lattice packing

Introduction

To avoid edge effects in molecular dynamics simulations, periodic boundary conditions (PBC) are commonly used. That means that the computational box is surrounded in a space-filling way by replica boxes with identical content. It has been shown[1] that in 3D there are five convex box types that can be stacked in a space-filling way, namely the triclinic box, the hexagonal prism, the rhombic dodecahedron, the elongated rhombic dodecahedron, and the truncated octahedron (see Fig. 1a–e). More recently, Bekker[2] showed that a molecular simulation in any one of these box types can be transformed into an identical simulation in any one of the other box types.

A large class of problems where the choice of the periodic box is important is the simulation of a single macromolecule in a solvent. For reasons of efficiency it is highly desirable that the amount of solvent is minimized. Nowadays, most molecular simulations of this type are set up in a box with a complex shape, often the rhombic dodecahedron or the truncated octahedron, which may subsequently be transformed into a simulation in a triclinic box, depending on the simulation package. The reason that the simulation is set up in a box with a complex shape is that it is (erroneously) believed that in this way a tighter fitting box may be constructed around the molecule, resulting in a minimal amount of solvent, and a more efficient simulation. The reason that in some packages the actual simulation is performed in the triclinic box is that it is most easily implemented and is computationally efficient.

The method presented here to minimize the amount of solvent is not based on constructing a tight fitting box as is normally done, but by calculating a near densest lattice packing. From the resulting packing a computational box can be derived, where the type of the box may be chosen to be any of the five box types mentioned above. Because the triclinic box is simple to implement and results in efficient simulations, we will only consider the triclinic box. Before describing our method we will consider current methods to construct a computational box.

Let us call the macromolecule to be simulated m. Usually, in molecular simulations short range interactions (e.g., Lennard-Jones) are truncated beyond a distance r_{co}, whereas long range interactions are either truncated or handled by the Ewald-sum technique, the PME technique, or something similar. In molecular simulations replicas of m are allowed to have long range interaction, but normally should not have short range interactions with

Correspondence to: H. Bekker; e-mail: bekker@cs.rug.nl

*Current address: Institute of Information and Computing Sciences, Utrecht University, P.O.B. 80089, 3508 TB Utrecht, The Netherlands

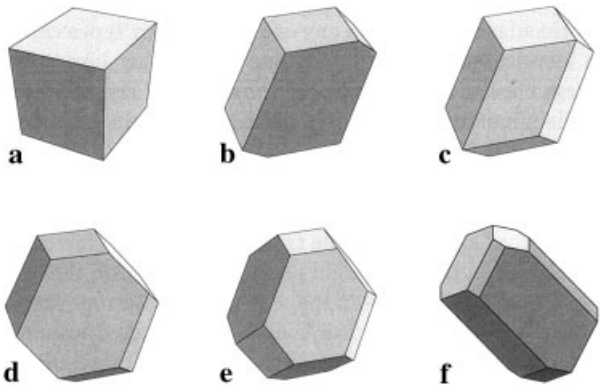

Figure 1. The five convex space filling boxes commonly used in molecular dynamics (MD) simulations with periodic boundary condition: (a) the triclinic box, (b) the hexagonal prism, (c) the rhombic dodecahedron, (d) the elongated rhombic dodecahedron, and (e) the truncated octahedron. In current MD software, of the first box type the regular and the general instance is used, namely the cube and the triclinic box. Of the other four box types only the regular instances are used. However, generalized box types may also be used. For example, a general truncated octahedron is shown (f).

each other. That means that no two replicas of m should be closer than r_{co}. This may be reformulated by introducing a shape M, defined as m dilated by a layer of width r_{lw} (see Fig. 2). In principle it is enough that $r_{lw} = \frac{1}{2} r_{co}$, but to allow for some conformational change during the simulation it is advisable to make r_{lw} larger than $\frac{1}{2} r_{co}$. In the examples presented $r_{lw} = \frac{10}{7} \times \frac{1}{2} r_{co} \approx 1.43 \times \frac{1}{2} r_{co}$. When two replicas of M do not overlap but at most touch each other we can be sure that no two replicas of m are closer than r_{co}. Because the mathematics of contact situations is more simple than the mathematics of minimal distance situations, in the following we will mainly work with contact situations between replicas of M instead of minimal distance situations between replicas of m.

During an unconstrained MD simulation m will undergo rotational diffusion. Ideally, the computational box B is constructed by enclosing M with one of the five box types, such that m may rotate freely in B and may show some conformational change without leaving B. Let the space outside M and inside B be called D, so, $D = B - M$ (see Fig. 2c). Now, an important observation is that the volume of D may be minimized provided that M does not overlap its own images, so, in that sense the solvent in D is superfluous. Minimizing the volume of D is relevant because per unit volume, simulating the solvent in D takes approximately the same CPU effort as simulating the molecules in M, so, denoting the volume of D by $\mathrm{vol}(D)$, $\approx \dfrac{\mathrm{vol}(D)}{\mathrm{vol}(B)}$ of the CPU time is spent on simulating irrelevant solvent. In the following, with setting up a minimal volume simulation we will mean constructing a computational PBC box B, so that $\mathrm{vol}(D)$ is minimal.

In the molecular simulation community, a triclinic box generally refers to a box spanned by three linearly independent vectors. Thus, the shape and orientation of a triclinic box are described by nine parameters, namely the components of its spanning vectors. In contrast, the terms rhombic dodecahedron and truncated octahedron generally refer to the *regular* forms of these box types. That is, they have a fixed shape scaled with one parameter. Therefore, the shape and orientation of these boxes is completely described by four parameters, namely one for the scale and three for the orientation. This indicates that it is not a good idea to use these boxes to set up a minimal volume simulation because the number of parameters to adjust the shape is less than the number of parameters of the general triclinic box. Much more freedom to adapt the shape of the computational box is obtained by using the *general* rhombic dodecahedron or *general* truncated octahedron. In ref. 2 it is shown that the shape and orientation of these boxes are determined by 12 and 14 parameters, respectively. In Figure 1f a general truncated octahedron is shown. General box shapes are not commonly used for two reasons. First, there is no general method to handle a shape matching problem in so many degrees of freedom. Second, the molecule will undergo rotational diffusion during the simulation. Random rotational motion means that during a simulation, potentially m occupies every position inside the minimal circumscribed sphere around m. So, instead of packing M in a box, the minimal circumscribed sphere of M has to be packed in a box. Because the regular rhombic dodecahedron is the minimal-volume PBC box containing the sphere this box or the truncated octahedron is often used.

Recently, Amadei et al. [3] proposed a method to constrain the rotational motion of m during a molecular simulation, without affecting the statistical mechanical consistency of the system (see the Discussion and Conclusion for more details). One reason to devise this method was to enable minimal volume molecular simulations. However, until now the method has not been used to perform minimal volume simulations. In the first place this is because no general method to fit a general PBC box around M is available. But even when such a method would be available,

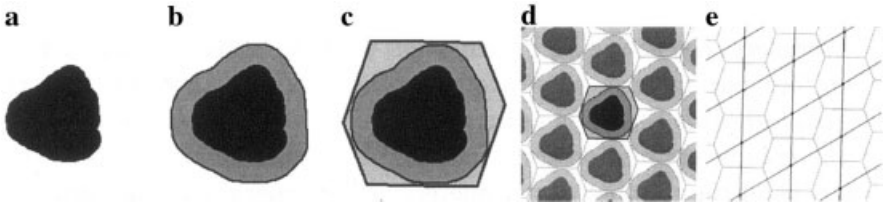

Figure 2. (a) A molecule m. (b) m surrounded by a layer of width r_{lw}, giving M. (c) A PBC box containing M. (d) Part of an infinite MD system S, formed by tessellating space with a PBC box s. (e) The lattice defined by the boxes in (d).

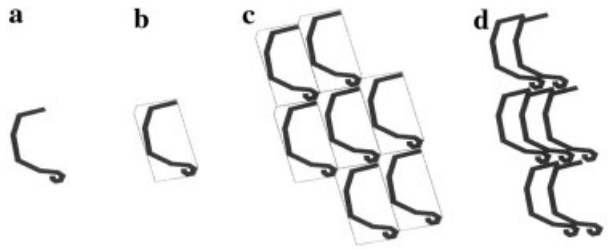

Figure 3. (a) A nonconvex 2D shape M. (b) A single system s consisting of M in an enclosing minimal PBC box. (c) Part of the infinite system S constructed by tessellating space with s. (d) A packing of M that is denser than the packing in (c). This shows that the smallest enclosing PBC box does not necessarily result in the densest infinite system.

constructing a minimal volume PBC box around M does in general not result in a minimal volume simulation. This can be illustrated with the following 2D example.

Suppose we want to set up a minimal volume simulation of the 2D body M shown in Figure 3a. In Figure 3b a single system s is shown consisting of M in its minimal-volume enclosing PBC box. In Figure 3c part of the infinite system is shown, which we will call S, obtained by tessellating space with s. In Figure 3d a lattice packing of M is shown that is denser than the packing in 3c. This example shows that the smallest enclosing PBC box does not necessarily result in the densest infinite system. It can be shown that only for convex M the minimal-volume enclosing PBC box results in a minimal-volume simulation. As we will see later, for nonconvex M, that is, for realistic enlarged macromolecules, in order to get a minimal volume simulation, in general M has to be fragmented in the box.

To summarize, we saw that the most general PBC box is determined by 14 parameters, and noted that no method is known to fit such a box in a minimal-volume way around M. In addition, we must consider the possibility that M has to be fragmented. Thus, we may conclude that this way of constructing a minimal-volume PBC box containing a possibly fragmented M is not possible currently. However, as we will show in the following, it is possible to do it the other way around, that is, instead of enclosing M by a small box and stacking this box, it is possible to stack M in a dense lattice packing and to derive from this packing a box.

Boxes, Their Related Lattices, and Lattice Packings

Let us denote by s a single molecular system, that is, a single PBC box B containing a molecule m and solvent. By S we denote the infinite molecular system, formed by tessellating space with an infinite number of translates of s, in a space filling way (see Fig. 2d). In ref. 2 it is shown that S consists of an infinite number of single systems s located at lattice points, where the lattice L is an infinite set of points defined as

$$L = i \times \mathbf{a} + j \times \mathbf{b} + k \times \mathbf{c} \quad i, j, k \text{ integer.}$$

The vectors **a, b,** and **c** are called the *lattice vectors.* They are determined by the type and shape of s.

In the previous section a PBC molecular system was viewed as a single system s, surrounded by an infinite number of identical systems. So, one might think that in S the box B can be recognized. That is, however, not the case; only the lattice structure can be recognized, in S no such thing as a box exists. Boxes are introduced for computational reasons in order to define a representative part of the infinite system. In general the box is defined as the Voronoi region of the lattice L, where the metric used to define the Voronoi region may be chosen freely.[2] So, a given lattice has an infinite number of shapes of Voronoi regions, the type of which may be any of the five space fillers mentioned earlier.

In the previous section we formulated the problem of setting up a minimal volume molecular simulation as finding a box B containing M (possibly fragmented), so that vol(D) is minimal. In the lattice view this may be reformulated as *finding the densest lattice packing of M.* This observation is essential for our method.

In the rest of this section we discuss in general terms how to find the densest lattice packing of M, and, assuming that we found the densest packing, how a PBC box may be constructed from that packing. The actual packing method is introduced in the following section. To keep things well defined we assume that M is polyhedral. Moreover, to avoid degenerate situations we assume that the volume of M is nonzero and that M has no points at infinity. For polyhedral convex M an analytical method to determine the densest lattice packing exists.[4] However, most bio-macromolecules m result in a nonconvex M. For nonconvex M there exists no analytical algorithm to determine the densest lattice packing, and in the computational geometry community this problem is considered as hard. For this reason we have devised a heuristic method to approximate the densest lattice packing of M, which we will call the near-densest lattice packing (NDLP) of M. The first step of our NDLP heuristic is similar to the approach in the case of convex bodies. A check is then added to filter out incorrect packings due to the nonconvexity of M.

Having found a NDLP with a lattice L defined by the vectors **a, b, c,** we must construct a computational box B with the property that, in a tessellation of space with B, translates of B are positioned at lattice points of L. This condition, however, does not completely determine the type and shape of B. B may be any Voronoi cell of L, where the metric is completely free. Thus, the type of B may be any of the five general space fillers. In the current context this approach is, however, too general. As was stated earlier, the triclinic box is most easily implemented and is computationally efficient. Therefore, we choose B as the triclinic box spanned by **a, b,** and **c.**

The last step of the NDLP method involves placing m in B. In principle, the location of m in B is arbitrary. Nevertheless, for convenience it is desirable to locate m in the middle of B. Sometimes, however, m will not fit entirely in B, it sticks out no matter where it is located in B. This does not matter, we simply locate m somewhere in the middle of B (see Fig. 4). For every atom of m protruding B it holds that it can be shifted over some vector $\mathbf{d} = i \times \mathbf{a} + j \times \mathbf{b} + k \times \mathbf{c}$ i, j, k integer, such that it falls in B. For every protruding atom such a vector is calculated and the atom is translated over this vector. Now all atoms of m are in B but m is possibly fragmented. This is no problem because in S complete

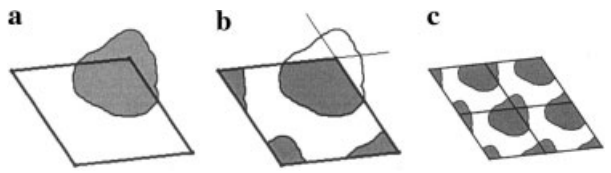

Figure 4. (a) 2D example of a minimal volume triclinic box B containing a protruding molecule m. (b) The protruding parts of m have been reset in B by shifting them over lattice vectors. This results in a fragmented molecule in B. (c) Part of the infinite MD system formed by tessellating space with B with fragmented m. In the infinite system, whole molecules are formed by fragments from various boxes.

molecules are formed from these fragments (see Fig. 4c). Finally, all voids in B are filled with solvent. Herewith, we have constructed the near-minimum-volume triclinic system s.

Calculating the NDLP of M

The first step in the NDLP algorithm is to construct the *contact body* of M, designated by N. Denoting by $M_{\mathbf{a}}$ the body M translated over the vector **a,** N has the following important property. *The boundary of N consists of all points* **a** *for which it holds that M and* $M_{\mathbf{a}}$ *touch without overlapping* (see Fig. 5). To construct N the Minkowski[5] sum is used. In Appendix A the Minkowski sum and its applications are discussed. The Minkowski sum takes as input two bodies P and Q, and returns a body R, defined as $R \equiv P \oplus Q \equiv \{\mathbf{a} + \mathbf{b} : \mathbf{a} \in P, \mathbf{b} \in Q\}$. The body M inverted in the origin, denoted by $-M$, is defined as $-M \equiv \{-\mathbf{a} : \mathbf{a} \in M\}$. The contact body N is constructed as

$$N \equiv M \oplus -M. \tag{1}$$

It can be shown easily that N is symmetric, and centered at the origin.

Let us now explain the core of the NDLP heuristic. We want to position M and three of its translates, $M_{\mathbf{a}}$, $M_{\mathbf{b}}$, and $M_{\mathbf{c}}$, in such a way that they do not overlap and *the volume of the triclinic cell spanned by* **a, b, c** *is minimal.* In principle, for this we have to search through all combinations of **a, b, c.** Therefore, in principle we are dealing with a nine-dimensional minimization problem. The key property of our NDLP method is that we reduce this nine-dimensional problem to a 3D problem by making the following choice. *We only search those combinations of* **a, b, c** *for which it holds that every body in the set* $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$ *is touched by the three other ones.* We call such a situation *all-contact* of $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$. That this choice leads to a three-dimensional minimization problem can be seen from the following. M is placed at an arbitrary location, **a** is chosen on the boundary of N. Thus, M and $M_{\mathbf{a}}$ touch. Because **a** is chosen on the surface of N there are two degrees of freedom in choosing **a**. Now we calculate the intersection of N and $N_{\mathbf{a}}$, which is a curve in 3D. On this curve we choose **b.** So, $M_{\mathbf{b}}$ touches M and $M_{\mathbf{a}}$. Because **b** is chosen on a curve there is one degree of freedom in choosing **b.** Finally, we choose **c** on the intersection of N, $N_{\mathbf{a}}$ and $N_{\mathbf{b}}$, which is a small set of points, for typical M ranging from 2 to 10. So, the number of degrees of freedom in choosing **c** is zero. Herewith the number of degrees of freedom of the search problem proves to be $2 + 1 + 0 = 3$. Searching through this 3D parameter space is the core of the NDLP heuristic.

Now let us assume that we are searching through all combinations of **a, b,** and **c,** according to our NDLP heuristic, with an appropriate search granularity to be discussed later. For every combination of **a, b,** and **c** we have to calculate $|\det(\mathbf{a}, \mathbf{b}, \mathbf{c})|$ and store the **a, b, c** that give minimal $|\det(\mathbf{a}, \mathbf{b}, \mathbf{c})|$. Obviously, M, $M_{\mathbf{a}}$, $M_{\mathbf{b}}$, and $M_{\mathbf{c}}$ do not overlap. However, for nonconvex M, there may exist one or more lattice points $\mathbf{d} = i \times \mathbf{a} + j \times \mathbf{b} + k \times \mathbf{c}$ i, j, k integer, for which M and $M_{\mathbf{d}}$ overlap. That this may happen is not easily understood as there is no 2D analogy. In Appendix B we try to develop some feeling for this. To filter out these cases, for every **a, b, c** with minimal $|\det(\mathbf{a}, \mathbf{b}, \mathbf{c})|$ we have to perform an additional test. That M and $M_{\mathbf{d}}$ overlap means that there are i, j, k not all 0, such that the lattice point $\mathbf{d} = i \times \mathbf{a} + j \times \mathbf{b} + k \times \mathbf{c}$ i, j, k integer is in the interior of N. So we have to test for all lattice points within a range 1/2 diam(N) of the origin whether they fall in the interior of N, where diam(N) is the diameter of N. The vectors **a, b,** and **c** giving the smallest value $|\det(\mathbf{a}, \mathbf{b}, \mathbf{c})|$ and passing the lattice test are the lattice vectors we are searching for.

Summarizing, the complete NDLP algorithm outline is as follows:

```
from m and r_lw construct M;
from M construct N;
forall a on boundary of N do
 forall b on boundary_intersection of N, N_a do
  forall c on boundary_intersection of N, N_a,
   N_b do
   if |det(a, b, c)| < old_det_abc and
   not point_of_L_inside_N then store(a, b, c);
  end // c loop
 end // b loop
end // a loop
put m in box a, b, c
fill voids with solvent
```

Let us briefly comment on our choice only to search through those **a, b, c** that fulfil the all-contact situation of $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$. It has been shown that there are other contact situations giving minimal volume.[4] Very probably this also holds for nonconvex bodies, but little is known about that. However, searching

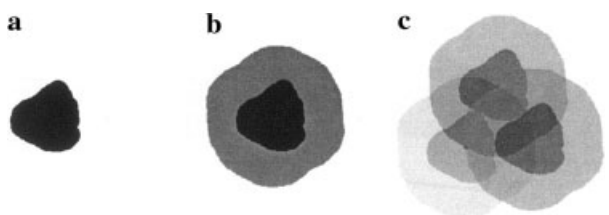

Figure 5. (a) 2D example of a body M, (b) the contact body N of M, (c) N used to construct a situation where three copies of M touch each other. Notice that N is point-symmetric.

through these situations would take much more CPU time than searching through the situations of our choice, probably without finding a considerably denser packing.

Implementation

In this section we explain how we transform the NDLP algorithm into an efficient and robust program. In the previous section we assumed that M was polyhedral, however in the implementation m and M are point sets, and N is polyhedral.

In our implementation the α-hull algorithm[8] plays an important role at two places. The α-hull algorithm takes as input a set of points and constructs an outer hull around these points. Whether a point is considered to be a boundary point or an interior point depends on the value of the parameter α. Conceptually, the α-hull algorithm works with a sphere of radius α. Points that can be touched by the sphere while it moves on the outside over the surface of the point set are by definition boundary points of the α-shape. So, for $\alpha = \infty$ the hull is the convex hull of the point set, and for $\alpha = 0$ every point is considered as a boundary point. The α-hull algorithm not only determines boundary points but also generates a triangulation of the hull. For every three boundary points that can be touched simultaneously by the ball a triangle is created.

Let us now construct M. Recall that m dilated with a layer of width r_{lw} gives M. We start by constructing a spherical point set *ball* by distributing a number of points more or less evenly on the boundary of a sphere with radius r_{lw}. We cover the sphere with 66 points but this number is somewhat arbitrary; we think that any number in the range 50–100 is acceptable. We choose 66 points because there is a nice algorithm to place this number of points more or less evenly on a sphere. We assume that r_{lw} has been chosen large enough to allow for some variation in the conformation of m. The macromolecule m is taken from some coordinate file, for example from the Protein Data Bank (PDB).[7] How the atoms are connected is irrelevant, for us m is simply a point set, where every point represents an atom. In m we incorporate all atoms of the macromolecule listed in the PDB file. A straightforward and correct way to calculate M would be $M \equiv m \oplus \text{ball}$. However, in this way the number of points in M would be 66 times the number of points in m. Because we only need boundary points of M we limit the number of intermediate and result points as follows. We determine the boundary points of m by applying the α-hull algorithm with $\alpha = r_{lw}$ to the point set m. Let us call this set of boundary points m_{bd}. From m_{bd} we construct M by calculating the Minkowski sum of m_{bd}. and ball, so, $M \equiv m_{bd} \oplus \text{ball}$. From the resulting set of points we delete points inside the α-hull of m, and points that are within a range of r_{lw} of two or more points of m_{bd}. Applying this technique, for example, to PDB molecule 1A32, consisting of 1102 atoms, results in 638 points defining the boundary of M (see Fig. 6). A similar method for constructing a dilated point set is described in ref. 6.

From M we construct the contact body N by taking the Minkowski sum of M and $-M$, so, $N \equiv M \oplus -M$. The number of points in N is the square of the number of points in M, so it was worth the effort to minimize the number of points in M in the previous step. Of N we only need part of the points. We want to have control over the number of remaining points of N, and we want to have these points distributed more or less evenly over the volume of N. This we do by using a grid-based selection method, that is, we construct a rectangular grid covering N, and determine for each point of N the cell in which it falls. In our experiments a grid of $32 \times 32 \times 32$ cells turned out to be a good choice as we will explain later. From every nonempty cell one point is kept. In interior cells a random point is chosen, and of the points in a boundary cell the one nearest to the boundary is kept. In this way N consists of typically 10,000 points. Now we switch from the point set representation of N to a polyhedral representation of N. For this we use the α-hull algorithm. We choose α to be the diameter of a grid-cell. In that way the shape of the polyhedron returned by the α-hull algorithm closely matches the shape of the input point set. Herewith we have a polyhedral representation of N consisting of a set of triangles defining the boundary of N. The reason that in the grid-cell selection method not only boundary points are kept but also interior points is that the α-hull algorithm is more reliable when the point set has also interior points.

In the NDLP method not only N but also translates of N are used. Translating N over some vector $\mathbf{a}$ is done by adding $\mathbf{a}$ to every coordinate of N, giving $N_{\mathbf{a}}$. To calculate the intersection curve of N and $N_{\mathbf{a}}$, represented by $N \cap N_{\mathbf{a}}$, we use the OBBTree algorithm.[9] This algorithm calculates of two sets of triangles in 3D which pairs of triangles intersect. For each pair of intersecting triangles we calculate the line segment that is in both triangles. The resulting set of intersection segments forms the intersection curve $N \cap N_{\mathbf{a}}$. Subsequently, we calculate the intersection of $N \cap N_{\mathbf{a}}$ and $N_{\mathbf{b}}$.

As explained before, the NDLP method is in essence a search problem in three continuous parameters. To make the method practicable a finite set of discrete points has to be chosen from the parameter space, that is, the granularity of the search process has to be determined. In our implementation the search granularity depends on the granularity of the triangulation of N. More precisely, the vector $\mathbf{a}$ runs through all of the centers of the triangles of N. In the same way, $\mathbf{b}$ runs through all of the centers of the line segments of $N \cap N_{\mathbf{a}}$. As the triangulation of N depends on the number of points returned by the grid-based selection method, the search granularity is controlled by the number of grid-cells. In our experience, using a grid of $32 \times 32 \times 32$ cells gives good search granularity.

We implemented the NDLP method in C++ using the α-hull algorithm from the computational geometry library CGAL,[10] and using the OBB-Tree algorithm.

Results

The NDLP algorithm was first tested on simple bodies for which the densest packing is known. We calculated the near-densest lattice packing of the sphere and the regular tetrahedron. The density of the densest lattice packing of the sphere is known to be $\frac{\pi}{3\sqrt{2}} \approx 0.7405$. The density calculated by the NDLP algorithm was 0.7447. That the NDLP density is marginally higher than the theoretical value is probably because the sphere was approximated

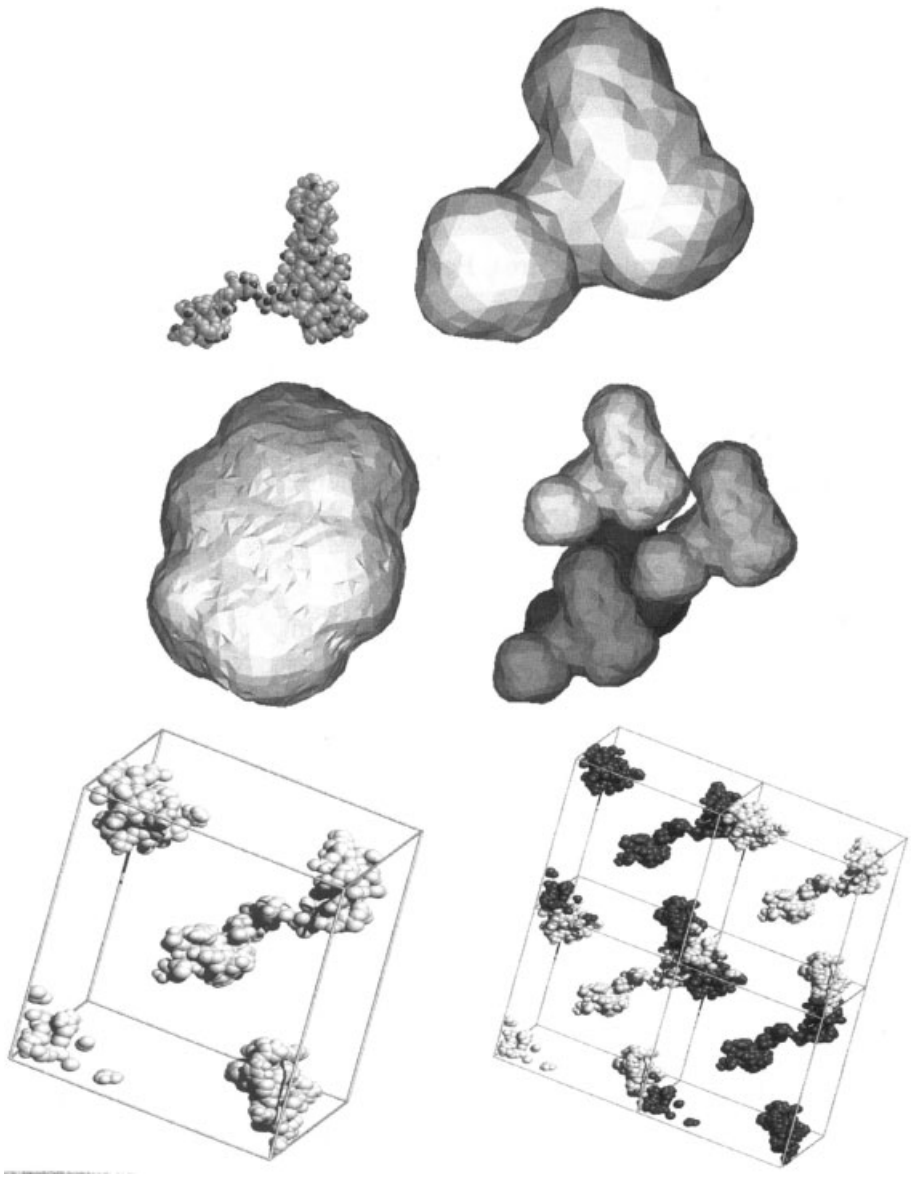

Figure 6. The NDLP method applied to molecule 1*A*32 from the PDB, consisting of 1102 atoms. The molecule is shown as a set of atoms, and M and N are shown as polyhedrals. The scaling and orientation of the figures vary. Top left: The molecule m. Top right: The body M, obtained by dilating m with a layer of width 10 Å. Middle left: The contact body N, defined as $N \equiv M \oplus -M$. Middle right: The NDLP of M. The four copies of M in the NDLP configuration touch each other. Bottom left: the NDLP box containing the fragmented molecule. The molecule is not centered in the box. Bottom right: four boxes fitted together, resulting in the reconstruction of the molecule. For simplicity only four of the eight boxes are shown. Although the molecule is not centered in the box whole molecules are formed, which shows that the location of the molecule in the box does not matter.

by an inscribed polyhedron with 326 vertices. Also, for the regular tetrahedron the NDLP performs very well. The theoretical maximal lattice packing density of the regular tetrahedron is known to be $\frac{18}{49} \approx 0.3673$, the NDLP method gave 0.3672.

Then the NDLP method was tested on 19 macromolecules randomly chosen from the PDB[7] (see Table I). The shape of these molecules ranged from almost spherical to complex (see Fig. 6). For every molecule three different boxes were constructed. A rectangular box and a regular rhombic dodecahedron were constructed by using current methods, and a triclinic box was constructed by the NDLP method. For every box $r_{lw} = 10$ Å. The rectangular box was constructed by the editconf procedure from

Table 1. Three Different Boxes Were Calculated for 19 Molecules, That Is, Each Molecule Was Placed in a Rectangular Box, a Regular Rhombic Dodecahedron, and a Triclinic Box Calculated by the NDLP Method.

Macro-Molecules			Rectangular Box		Dodecahedron		NDLP Triclinic Box		Speedup	
Nr.	PDB Code	Nr. of Atoms	Volume [nm^3]	Simul. Time [min]	Volume [nm^3]	Simul. Time [min]	Volume [nm^3]	Simul. Time [min]	Factor 1	Factor 2
1	1A32	1102	398.58	288	577.82	411	118.93	85	3.38	4.83
2	1A6S	805	141.25	97	142.43	105	80.08	58	1.67	1.81
3	1ADR	763	126.45	91	167.25	119	80.73	55	1.65	2.16
4	1AKI	1321	188.46	134	233.54	168	93.99	67	2.00	2.50
5	1BW6	595	144.18	99	130.27	89	66.32	45	2.20	1.97
6	1HNR	485	110.32	77	124.31	89	59.30	41	1.87	2.17
7	1HP8	686	133.17	94	177.10	129	77.57	53	1.77	2.43
8	1HQ1	982	201.62	143	218.77	158	103.71	72	1.98	2.19
9	1NER	768	170.14	118	147.91	105	85.35	58	2.03	1.81
10	1OLG	1808	297.63	210	468.93	337	203.44	145	1.44	2.32
11	1PRH	11676	1031.30	759	1337.80	987	611.67	467	1.62	2.11
12	1STU	668	130.79	91	190.32	136	73.41	50	1.82	2.72
13	1VCC	833	141.12	100	152.69	108	69.77	49	2.04	2.20
14	1VII	389	95.42	66	99.74	68	46.96	32	2.06	2.12
15	2BBY	767	155.59	109	159.26	113	80.78	56	1.94	2.01
16	1D0G*	1052	255.45	183	645.92	462	112.78	79	2.31	5.84
17	1D4V*	3192	1184.47	859	1319.21	951	451.23	329	2.61	2.89
18	1AAB	898	264.91	190	402.38	292	167.93	116	1.63	2.51
19	2ORC	720	197.03	139	230.64	161	125.85	88	1.58	1.82
			5369	3945	6925	4988	2707	1945	1.98	2.54

The boxes were calculated with r_{lw} = 10 Å. On average, the volume of the NDLP box is 50% of the rectangular box and 39% of the dodecahedron. Subsequently, the molecules were simulated for 25,000 timesteps in triclinic boxes, that is, the rectangular boxes and the dodecahedron boxes were transformed into their corresponding triclinic boxes. The simulations were performed with r_{co} = 14 Å. For each molecule and box type the simulation time is given. The last two columns show the speedup factors of the simulation time in the NDLP triclinic boxes relative to the simulation time in triclinic boxes derived from the rectangular boxes (factor 1) and relative to the simulation time in the triclinic boxes derived from the dodecahedron boxes (factor 2). The average of factor 1 is 1.98 and the average of factor 2 is 2.54. *Thus, due to the volume reduction, simulations in the NDLP box are about a factor of two faster.* The bottom line shows the total volume and time of the corresponding columns, and the average of factor 1 and factor 2.

GROMACS.[11] Editconf first determines the longest axis of the molecule and aligns the longest side of the box with this axis. GENBOX was also used to construct the regular rhombic dodecahedron box. GENBOX scales both boxes so that they fit tightly around the enlarged molecule. The NDLP triclinic box was constructed by the NDLP method. The volumes of the resulting boxes are listed in Table I. *On average, the volume of the NDLP box is 50% of the volume of the rectangular box and 39% of the volume of the dodecahedron.*

Subsequently we examined the simulation speed. For this purpose we transformed the rectangular and the dodecahedron boxes to their corresponding triclinic boxes, so every molecule was simulated in three different triclinic boxes. For every simulation, a 14 Å cut off with reaction-field was used, while some simulations were repeated using PME for the treatment of long range electrostatics, as a consistency check. The simulations in the NDLP triclinic box were performed using rotational constraining. Every simulation consisted of 25,000 timesteps of 2 fs. The simulations were performed on a single AMD Athlon 600 Mhz processor. In Table I for every molecule and box the simulation time is given. The last two columns show the speedup factors of the simulation time in the NDLP triclinic boxes relative to the simulation time in triclinic boxes derived from the rectangular boxes (factor 1) and relative to the simulation time in the triclinic boxes derived from the dodecahedron boxes (factor 2). The average of factor 1 is 1.98 and the average of factor 2 is 2.54. *Thus, due to the volume reduction, simulations in the NDLP box are about a factor of two faster.*

In ref. 3 it is shown that the overhead introduced by rotational constraining is negligible. This is confirmed in Table I. There it can be seen that 25,000 timesteps take $\approx$ 0.72 min/nm^3, independent of the use of rotational constraining. In Table I it can also be seen that the reduction of the volume results in a proportional increase of the simulation speed.

On a 600 Mhz AMD Athlon the time taken by the editconf procedure to calculate the rectangular and dodecahedron box is negligible. The time taken by the NDLP algorithm to calculate the NDLP triclinic box consists of a fixed and a variable part. The time to construct M from m varies and depends on the number of atoms in m. When m consists of up to 5000 atoms this time is negligible,

for 10,000 atoms it takes ≈ 5 min, for 25,000 atoms it takes ≈ 30 min, and for 60,000 atoms it takes ≈ 3 h. That is because the α-hull algorithm is quadratic in the number of points. The time to construct N plus the time taken by the actual search process does not depend on the number of atoms and is 15–30 min.

The 19 protein molecules used to test the NDLP method ranged in size from about 40 to about 1100 amino acids. Except for 17 all of the proteins were monomeric or tightly bound multimers. Structure 17 is a trimer consisting of three separate subunits. The first 15 structures were directly taken from their corresponding PDB-entries. For structures solved by NMR spectroscopy the first model was used. Structures 16 and 17 are modifications of the original PDB-entries. The protein used for 16 was a single receptor monomer, whereas for 17 a complete receptor trimer was used without the protein ligand, which was deleted from the file.

Structures 18 and 19 were solved by NMR, and an ensemble of 33 and 32 structures, respectively, were included in the PDB file. The shape of M should be chosen such that all possible conformational changes of m are taken into account, thus, for these two proteins the complete ensembles were used to calculate the NDLP. The rationale behind this is that all of these structures represent conformations that potentially could be sampled during the molecular simulations. Using such an ensemble of structures to determine the NDLP anticipates possible conformational changes during the simulation. Extra space to allow for these motions is considered in the packing. This could also be achieved by taking r_{lw} larger, but that would result in more space around the whole molecule instead of only around the regions involved.

The following procedure was used to determine the NDLP for ensembles of structures. First, all of the structures were superimposed as good as possible by performing a least squares fit on all atoms. This resulted in a set of $n \times na$ points, where na is the number of atoms in the molecule and n is the number of structures in the PDB file. Of this point set the NDLP was calculated in the usual way. So, $33 \times 898 = 29634$ and $32 \times 720 = 23040$ points were used to calculate the packings in 18 and 19, respectively.

Discussion and Conclusion

In this article we have described a method to determine the minimal computational box for the simulation of a macromolecule under periodic boundary conditions. The essence of the method is that the near-densest lattice packing of the dilated molecule is calculated and that from this lattice a computational box is derived. The fragmentation of the molecule in this box follows in a straightforward way from the lattice vectors. This differs from current methods. There, the unfragmented molecule is packed in the smallest box from a family of boxes with a known lattice packing. The reason that the volume of the boxes calculated with the NDLP method is smaller than the volume of boxes calculated with current methods is two-fold. First, in current methods only a small subset of the complete family of space filling boxes is considered, thus, only a subset of all possible lattices is considered. Second, in current methods the molecule is not fragmented in the box. As a result, the volume of the boxes calculated with the NDLP method is always less than or equal to the volume of the boxes calculated with current methods (see Table I). Moreover, assuming that the packing calculated by the NDLP method is the densest packing, the NDLP method gives the smallest possible box.

Before discussing the simulation results we should consider how rotational constraints may influence the results. In ref. 3 it is shown that, although rotational constraints modify the details of the dynamics, as long as the determinant of the mass tensor of the solute is approximately constant the statistical mechanics and thermodynamics of the constrained and unconstrained systems are identical. In the same article it is shown how the simulation results can be corrected when large deviations of the mass tensor occur, although typically these corrections are within the noise margins. For our simulations we did not apply the latter corrections. We compared the results of the simulations in the conventional box without rotational constraining with the results in the NDLP box with rotational constraining. Within the noise margin the temperature, pressure, energy, and conformational behavior are identical. Further validation is ongoing and we intend to report on that in a later article.

Until now, in our discussions we assumed that for a simulation in an NDLP box rotational constraints are used. However, for short simulations in an NDLP box, no rotational constraining is required. The simulation may continue as long as the molecule does not interact with its own images. How long this will be the case depends amongst others on the layer width r_{lw}, but is in general hard to predict. For that reason, for simulations in an NDLP box without using rotational constraints, it has to be monitored whether the molecule is interacting with its own images. When this happens the simulation should be stopped.

Let us finally see to what extent the NDLP method is compatible with existing MD methods. A box constructed with the NDLP method is an ordinary box. The only difference is that, because on average the box is smaller than current boxes, sometimes the molecule has to be fragmented to fit in the box. However, by applying the nearest image convention, during the actual simulation the molecule is reconstructed, thus in the infinite system only complete molecules are simulated. A fundamental property of MD simulation is that behavior of the infinite system does not depend on the shape of the unit cell, it does not matter whether it is a triclinic cell, a general truncated octahedron, or a nonconvex space filling cell. Nor does it matter how the molecule is stored in the cell, fragmented or unfragmented. The behavior of the simulation only depends on the configuration of the infinite system. For that reason, all simulation techniques that work for an infinite MD system with a lattice structure also work for an infinite system constructed from NDLP boxes. These techniques include Ewald summation, Particle-Mesh Ewald summation, scalar pressure scaling, per dimension pressure scaling and pressure scaling the complete pressure tensor, constraint dynamics, essential dynamics methods, and so forth. Thus, simulations in an NDLP box may be combined with all existing MD techniques that are valid for an infinite system with a lattice structure.

The NDLP method introduced in this article was already outlined in ref. 2. There it is described as an optimization problem in three parameters, aimed at finding the three lattice vectors of the densest lattice packing. It is suggested there that some general numerical optimization method is to be used to find the densest packing, but no actual implementation is presented. The NDLP method as presented in the article at hand is, however, not based on

a general optimization method. Instead, we use concepts and methods from computational geometry. This results in well-defined geometrical considerations and a robust implementation.

The NDLP method will be made available via `md.chem.rug.nl`. The program accepts both PDB files and files only containing atom coordinates. On the same internet page a tool is provided that has to do with the orientation of the lattice vectors. The orientation of the lattice vectors calculated by the NDLP method is unpredictable. However, some simulation packages require that the longest lattice vector lies along the positive x axis, and one of the remaining two lies in the x, y plane. We provide a tool to perform this rotation. We also provide a tool to translate all atoms of the molecule outside the box into the box.

The main conclusion of this article is that, due to the box volume reduction, for single macro-molecules the speed of molecular simulations using boxes constructed with the NDLP method, and using rotational constraints, is about two times the speed of simulations using boxes constructed with current methods.

Acknowledgments

We would like to thank Alan Mark for his support and valuable comments, and the reviewers for their constructive remarks.

Appendix A: The Minkowski Sum and Its Applications

In this section we develop some feeling for the Minkowski sum and its applications. The examples are given in 2D because conceptually and practically Minkowski sums and operations are completely analogous in 2D and higher dimensions. The Minkowski sum takes as input two point sets P and Q, and returns another point set R defined as

$$R \equiv P \oplus Q \equiv \{\mathbf{a} + \mathbf{b} : \mathbf{a} \in P, \mathbf{b} \in Q\} \tag{2}$$

This definition does not give geometrical insight into how R is formed from P and Q. To get some feeling for that we separately look at two properties of R, its shape and its position in the plane. The shape of R may be defined by a sweep process as follows. Choose some point $\mathbf{p}$ in P, and sweep with translates of P the plane such that $\mathbf{p}$ stays in Q. R consists of all points of the plane that are swept by P (see Fig. 7). The same shape R results when P and Q are exchanged. The position of R is, roughly speaking, the vectorial sum of the positions of P and Q. More precise, the rightmost coordinate of R is the sum of the rightmost coordinates of P and Q. The same holds for the leftmost, uppermost, and lowermost coordinates of R.

The Minkowski sum may be used to construct around some figure m a layer of width r_{lw}, giving a dilated figure M. It is calculated by taking the Minkowski sum of a circle S with radius r_{lw} and m, that is, $M \equiv S \oplus m$.

The figure $-M$ is defined as

$$-M \equiv \{-\mathbf{a} | \mathbf{a} \in M\} \tag{3}$$

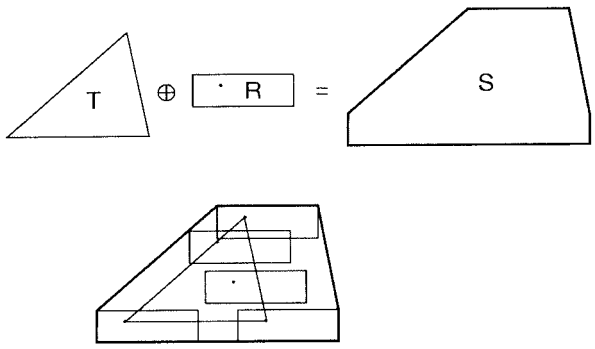

Figure 7. Upper figure: The Minkowski sum S of a triangle T and a rectangle R, that is, $S = T \oplus R$. In R some point $\mathbf{p}$ is chosen. The shape of S is the result of a sweep process. The plane is swept with translates of R in such a way that $\mathbf{p}$ stays in T. S consists of those points of the plane that are swept by R. In the lower figure some allowed sweep positions of R are shown. The same shape S results when R and T are exchanged.

that is, $-M$ is the figure M inverted in the origin. A figure N, called the *contact body* of M, may be constructed as

$$N \equiv M \oplus -M \tag{4}$$

thus, N is the set of points $\{\mathbf{a} - \mathbf{b} | \mathbf{a} \in M, \mathbf{b} \in M\}$. It can be shown easily that N is symmetric, and centered at the origin. Denoting by $M_{\mathbf{c}}$ the body M translated over the vector $\mathbf{c}$, N has the following important property. *The boundary of* N *consists of all points* $\mathbf{c}$ *for which it holds that* M *and* $M_{\mathbf{c}}$ *touch without overlapping.* This is not easily verified geometrically, however, the algebraic proof is simple. Proof: $N \equiv \{\mathbf{a} - \mathbf{b} | \mathbf{a} \in M, \mathbf{b} \in M\}$, thus for every point $\mathbf{c} \in N$ there are $\mathbf{a} \in M$, $\mathbf{b} \in M$ such that $\mathbf{c} = \mathbf{a} - \mathbf{b}$. Thus, $\mathbf{a} = \mathbf{b} + \mathbf{c}$. Because $\mathbf{a} \in M$ and $\mathbf{b} + \mathbf{c} \in M_{\mathbf{c}}$, it holds that M and $M_{\mathbf{c}}$ have at least one point in common, namely the point $\mathbf{a} = \mathbf{b} + \mathbf{c}$. Obviously, when $\mathbf{c}$ lies on the boundary of N, M and $M_{\mathbf{c}}$ only have boundary points in common, that is, M and $M_{\mathbf{c}}$ touch without overlapping. This property of N makes it useful to construct a dense packing of M.

Appendix B: Some Remarks on Lattice Packings of Nonconvex Bodies in 3D

In the algorithm explained in the section Calculating the NDLP of M we run through all all-contact situations of $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$. For every combination of $\mathbf{a}$, $\mathbf{b}$, and $\mathbf{c}$ giving minimal $|\det(\mathbf{a}, \mathbf{b}, \mathbf{c})|$ we check whether there is a nearby lattice point $\mathbf{d} = i \times \mathbf{a} + j \times \mathbf{b} + k \times \mathbf{c}$ i, j, k integer for which M and $M_{\mathbf{d}}$ overlap. That $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$ do not overlap is obvious. How can it be then that possibly M and $M_{\mathbf{d}}$ overlap? We will try to develop some feeling for that. Unfortunately, this cannot be explained in 2D—there it does not happen—so we have to explain it in 3D.

Let us assume that we have a body M consisting of spheroidal part and a long tail connected rigidly to it. Let us assume that we have been able to find an all-contact situation of the spheroidal

parts of the bodies $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$, such that the bodies $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$ (including their tails) do not intersect. We concentrate on a specific tail, say the tail of M. Now let us form one layer of bodies around $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$ by locating bodies at lattice points, where the layer only consists of bodies touching one or more of the set $\{M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$, but not M. Let us concentrate on a body $\tilde{M}$ in this layer. The lattice points have been calculated for touching bodies $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$, but M and $\tilde{M}$ do not touch. Therefore, M and $\tilde{M}$ possibly intersect, that is, $\tilde{M}$ and the tail of M may very well intersect. This conflicts with intuition. Intuition tells us that $\tilde{M}$ and M will not intersect because they are separated by a layer of bodies. Probably for convex bodies intuition is right, but for nonconvex bodies certainly not. So, the only way to be sure that M and $\tilde{M}$ do not intersect is to test explicitly for it. For this reason, for all lattice points within the range 1/2 diam(N) of the origin it has to be tested whether they fall in the interior of N. Such packings have to be rejected.

Now that we have seen that packings possibly have to be rejected we have to consider whether this can result in nonconvergence of the NDLP algorithm, that is, do bodies M exist for which it holds that, for every all-contact of $\{M, M_{\mathbf{a}}, M_{\mathbf{b}}, M_{\mathbf{c}}\}$, body $M_{\mathbf{d}}$ intersects body M for some lattice vector $\mathbf{d}$? This is an open mathematical problem. It does, however, not worry us very much because in our tests we did not encounter a single molecule for which the NDLP method did not converge, and we tested many more molecules than shown in Table I. Nevertheless, in our implementation it is signaled that not a single packing passes the lattice test. In that case the NDLP method would fail and the box would have to be constructed in the conventional way.

References

1. Tóth, L. F. Regular Figures; Pergamon Press: London, 1964.
2. Bekker, H. J Comput Chem 1997, 18, 1930.
3. Amadei, A.; Chillemi, G.; Ceruso, M. A.; Grottesi, A.; Di Nola, A. J Chem Phys 2000, 112, 9.
4. Betke, U.; Henk, M. Comput Geom 2000, 16, 157.
5. de Berg, M.; van Krefeld, M.; Overmars, M.; Schwarzkopf, O. Computational Geometry, Algorithms and Applications; Springer Verlag: Berlin, 2000.
6. Eisenhaber, F.; Lijnzaad, P.; Argos, P.; Sander, C.; Scharf, M. J Comput Chem 1995, 16, 273.
7. Berman, H. M.; Westbrook, J.; Feng, Z.; Gilliland, G.; Bhat, T. N.; Weissig, H.; Shindyalov, I. N.; Bourne, P. E. The Protein Data Bank. http://www3.oup.co.uk:80/nar/Volume_28/Issue_01/html/gkd090_gml.html
8. Edelsbrunner, H.; Muecke, E. P. ACM Trans Graph 1994, 13, 43.
9. Gottschalk, S.; Lin, M. C.; Manocha, D. Computer Graphics, Annual Conference Series 1996, 30, 171.
10. Computational Geometry Algorithms Library. http://www.cgal.org/Manual/doc_html/index.html
11. GROMACS: The GROningen MAchine for Chemical Simulations. http://www.gromacs.org/