A Fast, Scalable Method for the Parallel Evaluation of Distance-Limited Pairwise Particle Interactions

DAVID E. SHAW
D. E. Shaw Research and Development, LLC and Center for Computational Biology and Bioinformatics, Columbia University, 120 W. 45th St., 39th floor, New York, New York 10036

Received 31 December 2004; Accepted 17 February 2005
DOI 10.1002/jcc.20267
Published online in Wiley InterScience (www.interscience.wiley.com).

Abstract: Classical molecular dynamics simulations of biological macromolecules in explicitly modeled solvent typically require the evaluation of interactions between all pairs of atoms separated by no more than some distance R, with more distant interactions handled using some less expensive method. Performing such simulations for periods on the order of a millisecond is likely to require the use of massive parallelism. The extent to which such simulations can be efficiently parallelized, however, has historically been limited by the time required for interprocessor communication. This article introduces a new method for the parallel evaluation of distance-limited pairwise particle interactions that significantly reduces the amount of data transferred between processors by comparison with traditional methods. Specifically, the amount of data transferred into and out of a given processor scales as $O(R^{3/2}p^{-1/2})$, where p is the number of processors, and with constant factors that should yield a substantial performance advantage in practice.

 J Comput Chem 26: 1318–1328, 2005

Key words: molecular simulation; molecular dynamics; parallel computing; n-body problem; pairwise particle interactions

Introduction

The ability to routinely perform classical molecular dynamics (MD) simulations of biological macromolecules in explicitly modeled solvent for periods on the order of a millisecond would be of considerable value from both a scientific and a pharmaceutical perspective. Single simulations of this duration (by contrast with multiple simulations of the same molecular system whose *aggregate* simulation time approaches the millisecond time scale[1]) have, however, been considered to fall well outside the reach of current computational capabilities. Although efficient methods have been devised to approximate the aggregate effects of distant electrostatic interactions,[2–10] such simulations typically also require the evaluation of electrostatic and van der Waals interactions between all pairs of atoms separated by no more than some distance R, referred to in this article as the *interaction radius.* (In the case of the fast multipole methods, each atom actually interacts with all atoms contained within a fixed-size rectangular parallelepiped. For simplicity, however, this article assumes a fixed interaction sphere, as is typically used, for example, in the various Ewald-based methods.) To execute millisecond-scale MD simulations using currently envisioned technologies, the computational burden associated with the calculation of such pairwise interactions during each MD time step would almost certainly have to be shared among a large number of processors.[11]

The principal challenge in applying massive parallelism to accelerate such simulations is the time required for the interprocessor communication, which is necessary: (1) to bring together within each processor the updated positions of some subset of all atom pairs at the beginning of each MD time step, allowing the calculation of the interatomic force associated with each such pair [although we will use the phrase "position data" in the interest of brevity, in some simulations, certain other information (e.g., partial charges that vary over time) will be imported along with the atom's current position]; and (2) to combine the resulting partial force vectors with other such vectors at the end of each MD time step to obtain the net force acting on each atom.

In this article, we will refer to the first form of communication as *importing* and to the second as *exporting,* and will refer to the amount of data transferred during these processes as the *import load* and *export load,* respectively. The efficiency of a massively parallel MD simulation depends in large part on the way in which the import and export loads vary as a function of the number of atoms, n, and the number of processors, p. An efficient algorithm would take advantage of the fact that each atom need interact with only those atoms falling within a surrounding sphere of radius R to limit the import and export loads to a quantity dependent only on

Correspondence to: D. E. Shaw; e-mail: david@deshaw.com

R, and not on n. In addition, for a molecular system of fixed size, such an algorithm would ideally allow the import and export loads to be made arbitrarily small with a sufficient increase in the number of processors.

The methods traditionally used to parallelize MD simulations, however, have lacked either one or both of these desirable properties. In this article, we introduce a new technique, called the *NT* (for "neutral territory") *method,* that significantly reduces the amount of data transferred between processors by comparison with traditional methods. The method offers $O(R^{3/2}p^{-1/2})$ scaling, satisfying both of the properties outlined above. Moreover, the associated constant factors are such that the NT method should provide substantial performance advantages in most cases of practical significance.

This article describes the NT method and compares it with other techniques traditionally used for the parallel execution of MD simulations. In the interest of concreteness, a particular example of one of these standard techniques, which we will refer to as the *HS* (for "Half-Shell") *method,* is chosen for more detailed comparison with the NT method. The asymptotic behavior of the NT method is analyzed and compared with that of the HS method, and the practical performance of the two methods on problems and machines of realistic size are also compared. In a later section, we compare the NT method with another new method developed independently by Marc Snir[12] that also achieves $O(R^{3/2}p^{-1/2})$ scaling, although with a less favorable multiplicative constant. We also describe a set of modifications to Snir's method, based on techniques analogous to those we employ in the NT method, that allow his method to achieve performance approaching (although not equaling) that of the NT method for problems and machines of practical size.

Although the NT method should be applicable to other particle-based simulation techniques as well, we will describe the method in the context of MD simulation in the interest of concreteness. Throughout the article, we will concern ourselves only with pairwise interactions between nonbonded atoms, and not with the evaluation of either bonded or aggregated distant interactions. With regard to the computational environment, we will assume a communication model in which the same amount of time is required to communicate between any pair of processors. In addition, we will analyze only *bandwidth* requirements (determined by the *amount* of data transferred into a given processor) and not *latency* (a fixed time interval associated with the overhead entailed in transferring a packet of data from one processor to another, independent of the amount of data contained in that packet). In the interest of simplicity, we will also restrict most of our analyses to the import of atomic coordinates at the beginning of a given MD time step, with only brief discussion of the "mirror image" problem of exporting force data at the end of that time step.

Standard Methods for Parallelizing MD Simulations

Because the techniques used to parallelize a single MD simulation have already been surveyed by several authors,[13,14] we will provide here only a brief summary of some of the salient characteristics of the three major categories of traditional approaches to this problem: (a) atom decomposition methods, (b) force decomposition methods, and (c) spatial decomposition methods.

These methods may be usefully distinguished according to which processor is responsible for (a) calculating the force between a given pair of atoms and (b) updating the position of a given atom. In the *atom decomposition methods*[15–22] (also referred to as *particle decomposition* or *replicated data* methods), each processor is responsible for updating the positions of a fixed subset of atoms throughout the course of the simulation, and for calculating the forces attributable to the interaction of each of these atoms with all other atoms. Because each processor must import the updated positions of all atoms in the biomolecular system being simulated during every time step, the import load does not decrease with an increasing number of processors.

In the *force decomposition methods,*[13,23–34] each processor is again responsible for updating the positions of a fixed subset of atoms throughout the simulation, and we may thus speak of these atoms as *residing within* that processor. In contrast with the atom decomposition methods, however, each processor is responsible for calculating the forces attributable to a fixed set of atom *pairs.* In most cases, neither of the atoms in a given atom pair will reside within the processor that is responsible for their interaction, so both atoms will have to be imported. Atom pairs, however, are allocated to processors in a manner that allows each processor to import data from only $2p^{1/2}$ other processors during every time step. This allows the import load to be made arbitrarily small by increasing the number of processors. Although this property makes force decomposition methods attractive in the context of an *all-atom* simulation, such methods are not able to take advantage of a limited interaction radius to reduce the import load in simulations that use fast methods to approximate the aggregate effect of distant interactions.

In the *spatial decomposition methods*[13,16,17,35–54] (also referred to as *domain decomposition* or *geometric* methods), each processor is associated with a fixed block of *space* throughout the course of the simulation. During a given MD time step, the processor is responsible for calculating the forces experienced by all atoms currently residing within its own block and updating their respective positions, but atoms may move from one block (and thus one processor) to another over the course of the simulation. At the beginning of each MD time step, each processor imports data from a distance-limited region of space surrounding its own block whose volume is proportional to R^3. Spatial decomposition methods thus take advantage of a limited interaction radius to limit the import load to a quantity dependent only on that interaction radius, and not on the number of atoms in the system being simulated. Unlike the force decomposition methods, however, spatial decomposition methods do not allow the import and export loads to be made arbitrarily small by increasing the number of processors. Instead, import time approaches a fixed, nonzero asymptotic limit as the number of processors approaches infinity.

Although the choice between a force or spatial decomposition entails a significant tradeoff, the NT method offers the asymptotic advantages of both methods, and overcomes their most important limitations. Table 1 provides an informal comparison of the NT method with each of the three standard categories of parallel MD simulation methods with respect to the asymptotic characteristics discussed above.

Table 1. Comparison of Asymptotic Characteristics.

	Exploitable range limitation	Scaling with number of processors
Atom decomposition methods	None	No scaling
Force decomposition methods	None	$O(p^{-1/2})$ scaling
Spatial decomposition methods	$O(R^3)$ neighbors	No scaling
NT method	$O(R^{3/2})$ neighbors	$O(p^{-1/2})$ scaling

Simplified Two-Dimensional Analog

The basis for NT's advantage is most easily understood by first considering a simplified, two-dimensional analog of the algorithm that captures some (though not all) of its most salient characteristics, and by comparing this simplified variant with a similar analog of the HS method. In the context of the simplified model, atoms are positioned at various points within a plane, and the plane is divided into a grid of squares, each with side length b. Each square is associated with one processor, and may be identified by the coordinates of its low-coordinate corner (the *base coordinates* of that square). The square in which a given atom lies is referred to as the *home square* of that atom. The *import region* is defined as that region of the plane from which a given processor must import position data. Finally, the square associated with the processor in which two atoms interact is referred to as the *interaction square* for that interaction.

Within the context of this model, we may define a highly simplified, two-dimensional analog of the HS method (the *HS analog*) in which the processor associated with square S imports position data for all particles that lie within a distance R of S. The import region of this two-dimensional HS analog thus consists of the blue portion of Figure 1a, which has an area of $4bR + \pi R^2$. After this import process has been completed, the processor associated with S is able to interact each atom A in S with all atoms that lie within a distance R of A. (The simplified technique we have just described in fact allows two atoms to interact within *both* of their home squares, and thus includes a larger import region than is actually necessary; in the case of the two-dimensional analogs of the HS and NT methods, however, we will ignore this issue in the interest of simplicity.)

For purposes of comparison, we define a simplified, two-dimensional analog of the NT method (the *NT analog*) in which any pair of atoms A and B separated by no more than a distance R interact within that square whose x base coordinate is equal to the x base coordinate of A, and whose y base coordinate is equal to the y base coordinate of B. The import region of the processor associated with square S thus consists of that portion of the "row" and "column" containing S that lies within a distance R of S, as illustrated by the blue area in Figure 1b. Each interaction occurring within S will thus involve (a) one atom from either S itself or the horizontal bar extending a distance R from it in both directions, and (b) one atom from either S itself or the corresponding vertical bar.

It will be noted that the import region of the two-dimensional (though not the three-dimensional) NT analog is a proper subset of that of the corresponding HS analog, with an area of only $4bR$. Thus, the import requirements of the 2D NT analog are always lower than those of the HS analog, resulting in a reduction in the amount of time required to transfer data into and out of each processor. Moreover, as the number of processors grows large, the NT analog offers an asymptotic advantage over the HS analog: As the molecular system is partitioned into an increasingly large number of boxes with a progressively decreasing side length b, the area of the HS analog's import region approaches that of a circle of radius R, while that of the NT analog approaches zero.

It is worth noting that, in contrast with the HS analog, the NT analog interacts most pairs of atoms within a processor in which *neither* atom resides, as is the case for the force decomposition methods. Because it seems counterintuitive that such a method would result in a *lower* import load than a method that never needs to import more than one of the two atoms in any interacting pair, an intuitive discussion of the underlying basis of this advantage may be in order. We begin by noting that in both algorithms, we may identify two sets of atoms such that each interaction involves one atom from each set. In the case of the HS analog, the first set consists of only those atoms residing within a single square, while the second is the much larger set consisting of all atoms that lie within its entire import region. In the case of the NT analog, on the other hand, the two sets are much better balanced in size, each consisting of all atoms lying within a bar of area $b(2R + 1)$.

The number of interactions to be calculated within the processor is roughly (and as $p \to \infty$, exactly) proportional to the *product* of the number of atoms in each set, which should be approximately

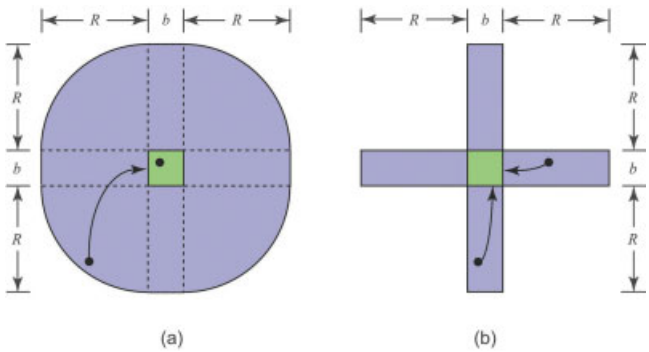

Figure 1. Import regions of the simplified two-dimensional HS and NT analogs. The import region of a simplified two-dimensional analog of the HS method is illustrated in (a), while that of a comparable 2D NT analog is depicted in (b). The spatial relationship between the two particles is the same in both subfigures, but interaction occurs within a different square. In both subfigures, the interaction square appears in green, while the import region appears in blue.

the same for the two methods, assuming a uniform distribution of atoms. The amount of data required to calculate these interactions, on the other hand, is proportional to the *sum* of the number of atoms in the two sets (excluding those already present in the interaction square), which is smaller for the NT analog than for the HS analog. For the same reason that the perimeter of a square (which is proportional to the *sum* of the length and width) is smaller than that of any other rectangle having the same area (which is equal to the *product* of the length and width), the better balance between the size of the two sets in the NT analog results in a smaller import load. Moreover, the relative advantage of the NT analog grows as the number of processors (and thus, the ratio of R to the side length of a single square) increases.

Although the simplified, two-dimensional "toy problem" outlined above provides some insight into one important aspect of NT's performance advantage, the situation becomes more complex in the case of a realistic three-dimensional MD simulation. First, although the NT analog may be generalized fairly easily to any *even* number of dimensions, doing so in the case of an odd number of dimensions is less straightforward. Whereas the two-dimensional NT analog interacts the atoms in two elongated bars, each aligned with one of the two axes, it is not immediately obvious how to partition *three* dimensions into two such sets in such a way as to minimize import volume. In addition, the import region of the actual NT algorithm will be "pruned" in such a way that each pair of atoms is brought together within only one processor, eliminating the duplication noted above. In contrast with the two-dimensional analog, the import region of the actual, three-dimensional NT method is not a strict subset of import region of the (3D) HS method. Indeed, although the actual NT method should allow a significant reduction in data transfer time in most cases of practical interest, when the number of processors is very small relative to the size of the molecular system being simulated, the NT method may actually have a *larger* import load than the HS method.

The actual (three-dimensional) HS and NT algorithms are specified, analyzed, and compared in the following three sections.

Locus of Interaction

Like the spatial decomposition methods, the NT method assigns each processor to a particular region of space throughout the course of the simulation. Atom pairs, however, are generally interacted within a processor in which neither atom resides, as in the case of the force decomposition methods. This section describes the specific rules used by the NT method to determine the processor in which a given pair of atoms is interacted (the *locus of interaction* for that atom pair), and compares these rules with those employed by the HS method. We begin with a description of the manner in which space is allocated among the processors in both methods.

For purposes of this article, we will assume that the molecular system being simulated resides within a box called the *global cell,* which serves as the unit cell of an infinite, spatially periodic system whose period in each dimension is equal to the side length of the global cell in that dimension, such that space is "tiled" with an infinite number of identical copies of the molecular system being simulated. The periodic boundary conditions imposed by such a tiling will simplify our analysis, and will thus be assumed for purposes of this article, although the NT method should also be applicable to simulations in which distant interactions are handled using fast multipole techniques or other approaches involving the evaluation of pairwise interactions within a locally circumscribed region of space.

Both the HS and NT methods divide the global cell into a regular three-dimensional rectangular grid of smaller boxes, each having the dimensions $b_x \times b_y \times b_z$. During a given MD time step, each processor is responsible for updating the positions of all atoms currently residing within one such box, which is referred to as the *home box* of both that processor and those atoms. Over the course of an MD simulation, however, a given atom may move from one box to another, at which point a different processor will assume responsibility for updating its position. Exploiting the one-to-one relationship between processors and boxes to justify a form of informal shorthand, we will sometimes apply terminology to one that is technically applicable to the other—referring, for example, to the interaction of a pair of atoms within a given box rather than within its associated processor. The *base coordinates* of a given box (or by extension, of the processor associated with that box, or of any atom in that box) are defined as the coordinates of the low-coordinate corner of that box.

In the HS method, a given pair of atoms is always interacted within the home box of one of the two atoms. Specifically, the locus of interaction for the HS method is defined by the following *interaction rules:*

1. If the two atoms have different x base coordinates, the atoms are interacted within the box with the smaller base coordinate.
2. If the two atoms have the same x base coordinate but different y base coordinates, the atoms are interacted within the box with the smaller y base coordinate.
3. If the two atoms have the same x and y base coordinates but different z base coordinates, the atoms are interacted within the box with the smaller z base coordinate.
4. If all three base coordinates of the two atoms are identical, the two atoms share the same home box, and the atoms are interacted within that box.

In the NT method, the locus of interaction for a given pair of atoms need not be the home box of either of those atoms. Instead, the atoms are interacted within a box (the *interaction box*) whose base coordinates are a mixture of the base coordinates of the two atoms. To specify the particular set of interaction rules employed by the NT method, it is useful to distinguish one of the two atoms as the *tower atom,* and the other as the *plate atom,* according to the following criteria:

1. If the two atoms have different x base coordinates, the one with the smaller x base coordinate is the tower atom.
2. If the two atoms have the same x base coordinate but different y base coordinates, the one with the smaller y base coordinate is the tower atom.
3. If the two atoms have the same x and y base coordinates but different z base coordinates, the one with the smaller z base coordinate is the plate atom.

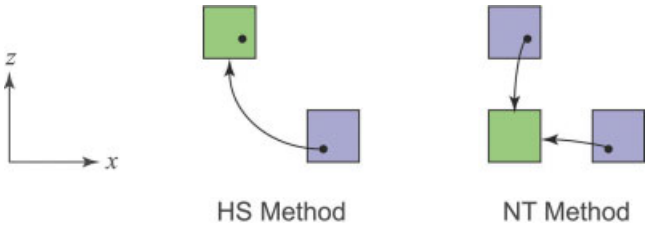

Figure 2. Locus of interaction, HS vs. NT. The green box within each subfigure represents the box in which a given pair of atoms (black dots) interacts, with the y axis omitted in the interest of visual clarity. When the two atoms have different base coordinates in all three dimensions, the HS method interacts the two atoms within the home box of the atom having the smaller x base coordinate, while the NT method interacts the two atoms within a box in which neither atom resides—specifically, that box whose x and y base coordinates are those of the atom with the smaller x base coordinate, and whose z base coordinate is that of the atom with the larger x base coordinate. When the two atoms share one or more base coordinates, the two methods break the resulting "ties" in specific ways, as detailed in the interaction rules that appear in the main text.

4. If all three base coordinates of the two atoms are identical, the tower and plate atoms may be chosen arbitrarily.

With these definitions, the NT method will always interact the tower and plate atoms within that processor whose x and y base coordinates are those of the tower atom, and whose z base coordinate is that of the plate atom.

Simplified versions of the interaction rules for the HS and NT methods are illustrated in Figure 2 for the case in which the two atoms have different base coordinates in all three dimensions (but with the y dimension omitted in the interest of visual clarity).

Import Region

As in the case of their two-dimensional analogs, our description of the HS and NT methods will make use of the term *import region,* which is defined in the three-dimensional case as that region of space from which a given interaction box must import data during each MD time step. The import region for a given interaction box I must encompass all atoms not already present within I that must be interacted within I (as specified by the interaction rules presented in the previous section) during the current time step. The import regions of both the HS and NT methods lie within a region called the *interaction neighborhood* of I, which includes all points outside of I that lie within a Euclidean distance R of some point in I. Depending on the number of processors, the shape of the interaction neighborhood may range from a box with slightly rounded corners and edges (when the number of processors is small) to a slightly extended and flattened sphere (when the number of processors is large).

The interaction neighborhood of I may be divided into the following 26 *interaction subregions,* which serve as the building blocks for the import regions of both the HS and NT methods:

1. Six *face subregions,* denoted γ_{-x}, γ_{+x}, γ_{-y}, γ_{+y}, γ_{-z}, and γ_{+z}, each associated with one of the six faces of I. Face subregion γ_{-x}, for example, is a $b_y \times b_z \times R$ box whose base is the low-x-coordinate face of I, and which extends outward from I in the $-x$ direction for a distance R.
2. Twelve *edge subregions,* denoted γ_{-x-y}, γ_{-x+y}, γ_{+x-y}, γ_{+x+y}, γ_{-x-z}, γ_{-x+z}, γ_{+x-z}, γ_{+x+z}, γ_{-y-z}, γ_{-y+z}, γ_{+y-z}, and γ_{+y+z}, each associated with one of the 12 edges of I. Edge subregion γ_{-x-y}, for example, is an "outward-pointing" quarter-cylinder of radius R and length b_z whose axis coincides with the low-x-, low-y-coordinate edge of I.
3. Eight *corner subregions,* denoted γ_{-x-y-z}, γ_{-x-y+z}, γ_{-x+y-z}, γ_{-x+y+z}, γ_{+x-y-z}, γ_{+x-y+z}, γ_{+x+y-z}, and γ_{+x+y+z}, each associated with one of the eight corners of I. Corner subregion γ_{-x-y-z}, for example, is an "outward-pointing" octant of a sphere of radius R whose center coincides with the low-x-, low-y-, low-z-coordinate corner of I.

Figure 3 shows an example of each type of subregion. It will be noted that the dimensions of the face and edge subregions are determined in part by the box dimensions. In the case of the HS method, the box is cubical, with side length $b_x = b_y = b_z = b$. In the NT method, the x and y dimensions of the box are equal, with side length $b_x = b_y = b_{xy}$, but its height, b_z, will not in general be equal to b_{xy}. Instead, the aspect ratio b_{xy}/b_z will be optimized to minimize the import load, in a manner discussed in the following section. Indeed, if it were not possible to vary the ratio of b_{xy} to b_z, the import load of the NT method would be asymptotically proportional to R^2 rather than $R^{3/2}$, as we shall see shortly.

The import region of the HS method is defined as the union of (a) the three face subregions γ_{+x}, γ_{+y}, and γ_{+z}; (b) the six edge subregions γ_{+x-y}, γ_{+x+y}, γ_{+x-z}, γ_{+x+z}, γ_{+y-z}, and γ_{+y+z}; and (c) the four corner subregions γ_{+x-y-z}, γ_{+x-y+z}, γ_{+x+y-z}, and γ_{+x+y+z}.

The import region of the HS method is depicted in Figure 4a. The fact that this region is (for the parameters assumed in creating the figure) roughly hemispherical rather than roughly spherical is attributable to the fact that a given pair of atoms need not interact within the home boxes of *both* atoms. Instead, a force vector representing, for example, the force exerted on atom p by atom q may be calculated within the home box of p and added into a vector sum representing all force components acting on p. A negated version of the same force vector, representing (by New-

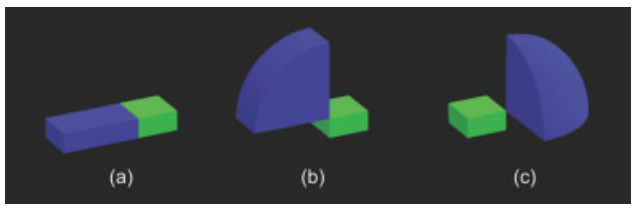

Figure 3. Interaction subregions. Sample interaction subregions are shown in blue, while the interaction box appears in green. One of the six face subregions is shown in (a), one of the 12 edge subregions in (b), and one of the eight corner subregions in (c).

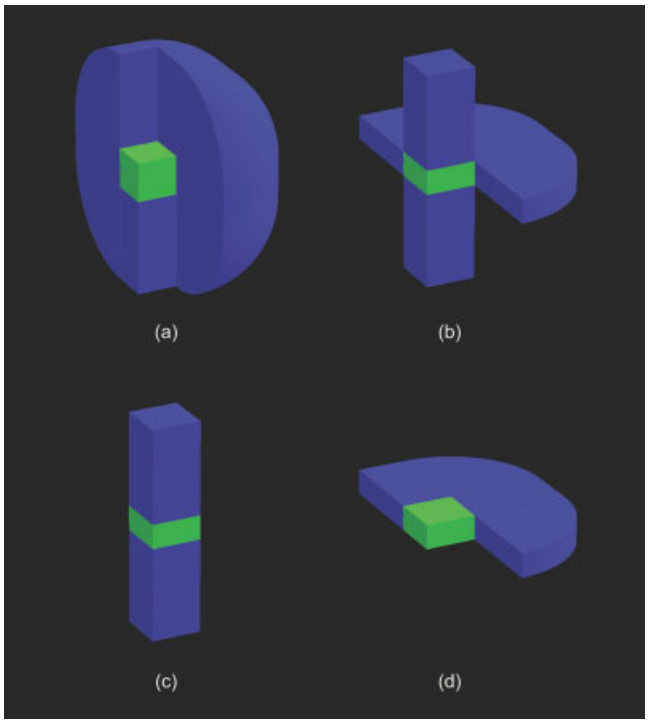

Figure 4. Import regions of the HS and NT methods. The import region of the HS and NT methods are illustrated in (a) and (b), respectively. In both cases, the interaction box appears in green, while the import region appears in blue. The tower and plate of the NT method are illustrated separately in (c) and (d), respectively, with the outer tower and outer plate appearing in blue.

ton's third law) the force exerted on q by p, may then be exported to the home box of q and combined with all other forces acting on q. It is easily verified that for each atom pair $\{p, q\}$ separated by no more than R, either:

1. p and q share the same home box,
2. q resides within the import region of the home box of p, or
3. p resides within the import region of the home box of q.

Thus, the import region of the HS method is sufficient to allow each atom pair to be interacted within some processor.

The import region of the NT method, on the other hand, consists of the union of (a) the four face subregions γ_{+x}, γ_{+y}, γ_{-z}, and γ_{+z}, and (b) the two edge subregions γ_{+x-y} and γ_{+x+y}, but includes no corner subregion—a feature that is responsible in part for its asymptotic advantage over the HS method. For convenience, certain portions of the NT method's import region are given special names. In particular, the union of γ_{+z} and γ_{-z} is referred to as the *outer tower,* while the union of the outer tower and the interaction box is referred to simply as the *tower.* The union of subregions γ_{+y}, γ_{+x+y}, γ_{+x}, and γ_{+x-y} is referred to as the *outer plate,* while the union of the outer plate and the interaction box is referred to simply as the *plate.* It should be noted that the interaction box belongs to both the tower and the plate. Intuitively, the tower is the box that would be obtained by stretching the interaction box by a distance R in both the positive z and negative z directions. The plate includes the interaction box and an adjacent "terrace" of thickness b_z extending (roughly speaking) a distance R away from the tower, but continuing only about halfway around it. The tower and plate are depicted in Figures 4c and d, respectively, while the full import region of the NT method is illustrated in Figure 4b.

Referring to the interaction rules presented in the previous section, it is easily verified that for each pair of atoms, there exists some box I such that one of the atoms lies within the tower of I while the other lies within its plate. (One or both of these atoms, however, may lie within the interaction box I itself, and may thus not need to be imported.) Thus, the import region of the NT method is sufficient to allow each pair of atoms separated by no more than a distance R to be interacted within some processor. After a given pair of atoms has been interacted within I, oppositely oriented copies of the resulting force vector may be exported to the home boxes of the two atoms.

Import Volume

Assuming uniform density (measured in atoms per cubic Ångström), the import load (the amount of atomic coordinate data that must be transferred into each processor during each MD time step) is proportional to the volume of the import region. In this section, we thus examine the volume of the import region for the HS and NT methods.

Before proceeding to a quantitative analysis, it may be useful to review the first two rows of Figure 5, which illustrate the manner in which the interaction box and import region of the HS method and the NT method, respectively, scale with an increasing number p of processors. The volume of each box is inversely proportional to p for both the HS and NT methods, and for a given value of p, is the same for both methods. With the HS method, the box remains cubical as p increases. In the NT method, however, the box becomes shorter and fatter with increasing p. As is apparent in the figure, the relative advantage of the NT method over the HS method grows as the number of processors increases. In the limit as $p \to \infty$, the import region of the HS method closely approximates a hemisphere with radius R, while that of the NT method becomes a collection of regions having a much smaller aggregate volume.

We now provide a quantitative analysis of the import volumes of the two methods, in both absolute and asymptotic terms. The import region of the HS method includes: (a) three face subregions, each of volume Rb^2, (b) six edge subregions, each of volume $\pi R^2 b/4$, and (c) four corner subregions, each of volume $\pi R^3/6$.

Thus, the total import volume of the HS method is

$$V_i = 3Rb^2 + \frac{3}{2}\pi R^2 b + \frac{2}{3}\pi R^3$$

and the limit as $p \to \infty$ is

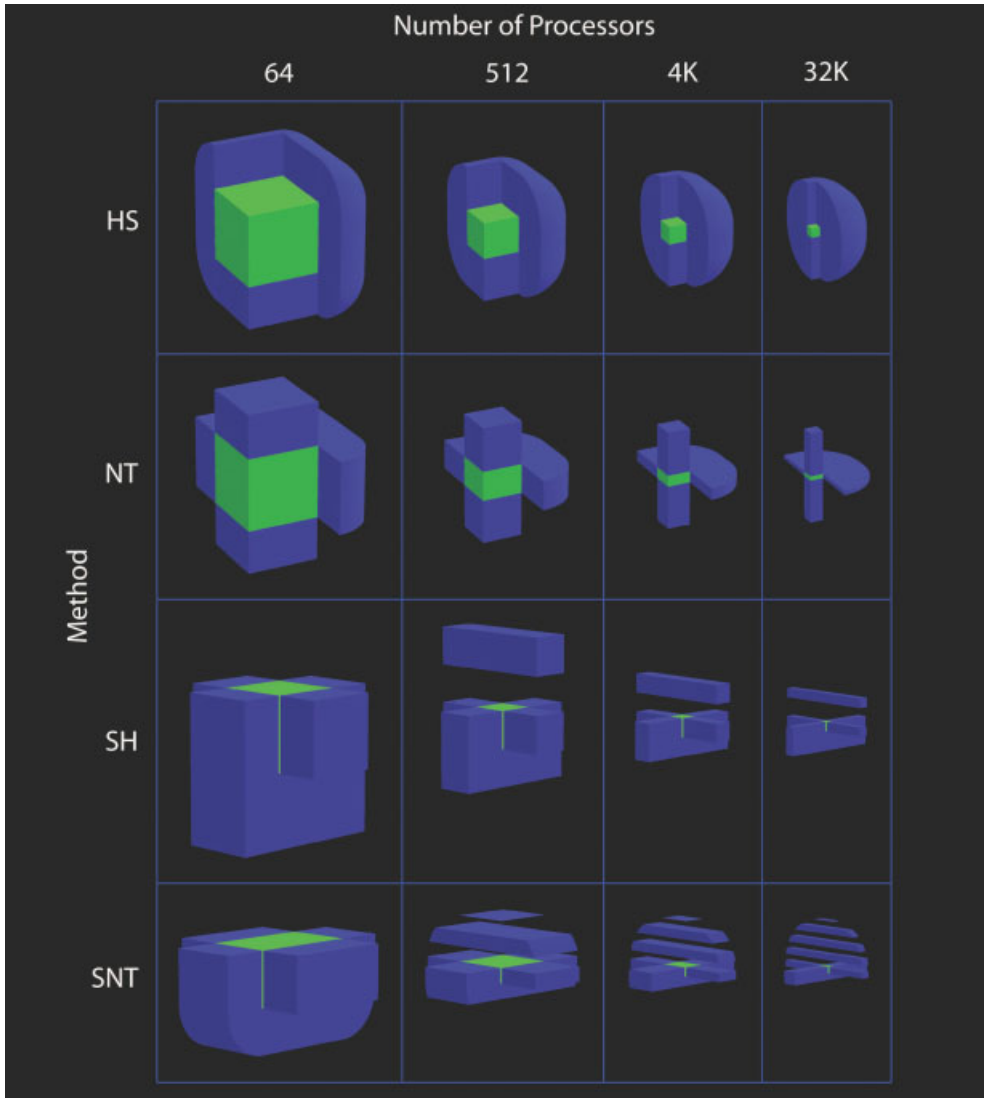

Figure 5. Scaling of import regions. The interaction box and import region of the HS, NT, SH, and SNT methods are shown for a machine containing 65, 512, 4K, and 32K processors, assuming a molecular system with 50,000 atoms, an interaction radius of 12 Å, and a density of 0.1 atom/Å^3.

$$V_{i\infty} = \frac{2}{3}\pi R^3.$$

The dimensions (and thus the volume) of the NT method's import region are determined in part by the box dimensions. Although the *volume* of the box is fully determined by the size of the molecular system and the number of processors, the aspect ratio b_{xy}/b_z is a free parameter that can be chosen in such a way as to minimize interprocessor communication. To do so, we express the import volume V_i in terms of b_{xy}, b_z, and R, then solve for the aspect ratio that minimizes this quantity. The import region of the NT method consists of:

1. The outer tower, which in turn consists of:
   a. Two face subregions, each with volume Rb_{xy}^2, and
2. The outer plate, which in turn consists of:
   a. Two face subregions, each with volume $Rb_{xy}b_z$, and
   b. Two edge subregions, each with volume $\pi R^2 b_z/4$.

Thus, the volume of the import region of the NT method is

$$V_i = 2Rb_{xy}^2 + 2Rb_{xy}\,b_z + \frac{\pi R^2 b_z}{2}. \quad (1)$$

Note that b_{xy} and b_z are subject to the constraint $V_b = b_{xy}^2 b_z$, where $V_b = V/p$ is the box volume and V is the volume of the global cell (that is, of the molecular system being simulated). We can thus express b_z in terms of b_{xy} and V_b, yielding

$$V_i = 2Rb_{xy}^2 + \frac{2RV_b}{b_{xy}} + \frac{\pi R^2 V_b}{2b_{xy}^2}. \quad (2)$$

Although it might appear from this equation that V_i is proportional to R^2 for large p, the import-minimizing value of b_{xy} is itself a function of R, and turns out to be proportional to $R^{1/4}$ when p is large, resulting in an import volume that scales as $R^{3/2}$. In particular, by differentiating eq. (2) with respect to b_{xy}, setting the result equal to zero, and solving the resulting quartic equation for b_{xy}, we find that the import load of the NT method is minimized when

$$b_{xy} = \frac{\sqrt{c} + \sqrt{V_b c^{-1/2} - c}}{2}$$

where

$$c = \frac{d}{6} - \frac{2\pi R V_b}{d}$$

and

$$d = \{27V_b^2 - 3\sqrt{3V_b^3[(4\pi R)^3 + 27V_b]}\}^{1/3}.$$

Substituting this optimal value of b_{xy} into the equation $b_z = V_b/b_{xy}^2$ yields the corresponding (optimal) value of b_z, while substitution into eq. (1) yields the (optimized) import volume of the NT method. Although the functional forms of the optimal values of b_{xy}, b_z, and V_i are relatively cumbersome for finite p, a Taylor-series expansion around the point $V_b = 0$ reveals that as the number of processors grows large, these quantities approach the much simpler asymptotic values

$$b_{xy\infty} = \frac{(\pi R V_b)^{1/4}}{\sqrt{2}}$$

$$b_{z\infty} = 2\sqrt{\frac{V_b}{\pi R}}$$

$$V_{i\infty} = 2\pi^{1/2} R^{3/2} V_b^{1/2} \quad (3)$$

and because $V_b = V/p$,

$$V_i = O(R^{3/2} p^{-1/2}),$$

consistent with the asymptotic results summarized in Table 1. The $p^{1/2}$ factor in this asymptotic expression implies that the amount of data transferred into and out of a given processor approaches zero as the number of processors becomes very large, in contrast with the HS method, for which the import volume asymptotically approaches the volume of a hemisphere of radius R. Also of interest is the fact that as p grows large, the amount of transferred data becomes proportional not to the volume of the interaction neighborhood, but to the *square root* of this volume.

From a practical perspective, however, it is important to compare not only the asymptotic behavior of the HS and NT methods, but also their absolute performance for problems and machines of practically relevant sizes. The first two rows of Table 2 provide a comparison of the two methods for the case of a molecular system with 50,000 atoms, an interaction radius of 12 Å, and a density of 0.1 atom/Å^2, for a wide range of machine sizes. It will be noted that, under the assumptions of our analysis, NT should offer a nontrivial improvement over HS when running on a contemporary cluster of moderate size, and a rather dramatic improvement on a massively parallel supercomputer like IBM's BlueGene/L.[55] (BlueGene/L is designed to embody as many as 65,536 nodes, each containing two processors, and is expected to be applied to, among other things, the exploration of protein folding trajectories using MD simulation.) Figure 7 compares the two methods (along with two others that will be discussed in the following section) in a format that highlights their asymptotic behavior.

Table 2. Time Required to Import Atomic Coordinate Data.

	Number of Processors			
Method	64	512	4K	32K
HS	3126	1389	787	552
NT	2339	686	211	68
SH	5469	1758	317	81
SNT	2722	771	226	71

Limit of measurement is the amount of time required to import one atom. Assumptions: 50,000 atoms, interaction radius = 12 Å, density = 0.1 atom/Å^3.

Comparison with and Enhancements of Snir's Method

In recently published article,[12] Marc Snir described another method, that, like the NT method, achieves $O(R^{3/2}p^{-1/2})$ scaling, though with a somewhat less favorable multiplicative constant. Because his method, like NT, captures the principal advantages of both force-based and spatial decomposition methods, he referred to his technique as a "hybrid" method, and we shall thus refer to it here as the *SH* (for "Snir's Hybrid") *method.* Although the two methods were developed independently and are quite different in their general approaches and the nature of their import regions, they also share certain characteristics that account for their asymptotic advantages by comparison with traditional methods. In this section, we describe the essential elements of the SH method, compare its performance to that of the NT method, and describe the manner in which the performance of the SH method can be enhanced using techniques analogous to those we have employed in the NT method.

Employing the terminology used in this article, the SH method associates each processor with a cubical box of side length b. Defining $r = \lceil R/b \rceil$ and $\tilde{r} = \lceil \sqrt{r+1} \rceil$, the import region of the processor whose base coordinates are (i, j, k) consists of two subsets of its interaction neighborhood:

1. A contiguous region, which we will refer to as the *base,*

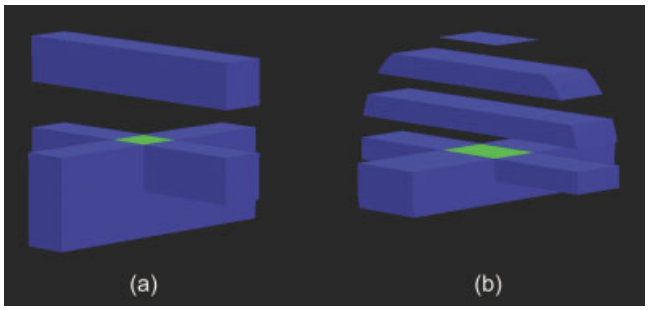

Figure 6. Import regions of the SH and SNT methods. The import region of the SH and SNT methods are illustrated in (a) and (b), respectively. In both cases, the interaction box appears in green, while the import region appears in blue.

consisting of the set of boxes with base coordinates $(i + u, j, k - w_1)$ where $u \in [-r, r]$ and $w_1 \in [0, \tilde{r} - 1]$.

2. A set of noncontiguous regions, which we will refer to individually as *bars* and collectively as the *comb,* consisting of the set of boxes with base coordinates $(i, j + v, k + w_2\tilde{r})$, where $v \in [-r, r]$ and $w_2 \in \lfloor 0, r/\tilde{r} \rfloor$].

The import region of the SH method is illustrated in Figure 6a, while the third row of Figure 5 illustrates the manner in which it scales with an increasing number of processors.

As in the case of the NT method, the SH method guarantees that for each pair of atoms separated by no more than a distance R, there exists some interaction box I such that one of the atoms lies within the base of I while the other lies within its comb. (Because the interaction box is included within both the base and the comb, however, one or both of these atoms may lie within I itself, and may thus not need to be imported.) Thus, the import region of the SH method is sufficient to allow each pair of atoms separated by no more than a distance R to be interacted within some processor.

To quantify the import load under the SH method, we note that the volume of the base is $V_b(2r + 1)\tilde{r}$, while each of the $(\lfloor r/\tilde{r} \rfloor + 1)$ bars in the comb has a volume $V_b(2r + 1)$. Because both the base and the comb include the interaction box, which is not part of the import region, we must subtract $2V_b$ from the sum of base and comb volumes, yielding the import volume

$$V_{i\mathrm{SH}} = V_b[(2r + 1)(\tilde{r} + \lfloor r/\tilde{r} \rfloor + 1) - 2],$$

where $V_b = b^3 = V/p$ is the box volume, and V is the volume of the system being simulated. Eliminating all "floors" and "ceilings" (the effect of which becomes negligible as V_b approaches zero) and performing a series expansion around $V_b = 0$, we find that as the number of processors approaches infinity, the import volume of the SH method approaches

$$V_{i\mathrm{SH}\infty} = 4R^{3/2}V_b^{1/2}$$

which is larger than the asymptotic import volume of the NT method by a factor of $2/\sqrt{\pi} \approx 1.13$.

Although the advantage of the NT method over the SH method is thus relatively modest in the limiting case of an infinite number of processors, the advantage for problems and machines of practical size is larger. The time required to import atomic coordinate data using the SH method (as described in Snir's article) appears in the third row of Table 2 for our example molecular system and for machines containing various numbers of processors. In reviewing Table 2 and its graphical counterpart, Figure 7, it will be noted that the SH method requires significantly more time for data import than the NT method for all of the sampled values of p. Indeed, for the cases of 64 and 512 processors, even the HS method appears to perform better than the SH method.

The advantage of the HS method over the SH method for systems of this size, however, does not reflect an intrinsic limitation of Snir's general approach. Snir's article, which was published in a theoretical computer science journal, focused largely (and, for that audience, quite appropriately) on the asymptotic behavior of his method as the number of processors approached infinity, and not on the optimization of his approach for machines of practical size. As we shall see, with certain modifications, Snir's method can, in fact, achieve performance approaching (although not equaling, for finite p) that of the NT method for problems and machines of reasonable size.

The first of these modifications is a relatively obvious one (which was presumably not incorporated in Snir's specification and analysis of the SH method only because of the asymptotic and theoretical focus of his article): eliminating from the import region all points lying more than a distance R from any point in the interaction box, as is the case for both the HS and NT methods. Our other modifications involve the optimization of certain parameters in a manner analogous to the import-minimizing optimization

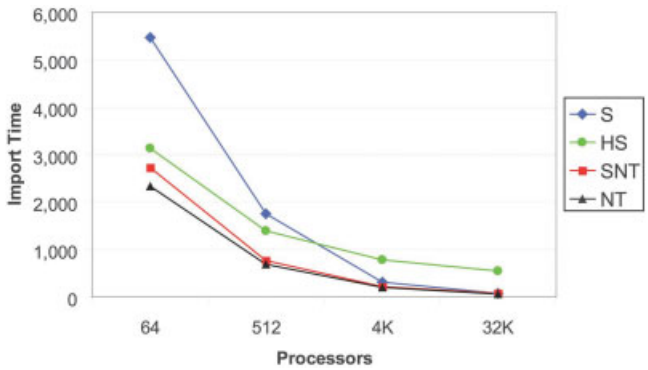

Figure 7. Time required to import atomic coordinate data. The four methods discussed in this article are compared with respect to the time required for data import, assuming a system with 50,000 atoms, an interaction radius $R = 12$ Å, and a density of 0.1 atom/Å^3. The unit of measurement is the amount of time required to import one atom. Although the import time of the NT, SH, and SNT methods may be made arbitrarily small by adding processors to the system, that of the HS method approaches a fixed, nonzero asymptote as its import region evolves toward a sphere of radius R. Although NT offers the best performance throughout the range of machine sizes shown in this figure, HS would outperform NT if the number of processors were very small ($\leq$4, under the assumptions of this figure).

performed in the NT method. In particular, we begin by removing the restriction of a cubical box, introducing the variable parameters b_x and b_y (which, together with V_b, determine the remaining box dimension b_z). In addition, we replace $\tilde{r}$, which in the SH method is always defined as $\lceil \sqrt{r+1} \rceil$, with a third adjustable parameter, s. (Like $\tilde{r}$, however, the parameter s will assume only integer values.) We then perform a global optimization over the parameters b_x, b_y, and s to find that combination of parameters that minimizes import volume.

The resulting variant of the SH method, which will be referred to in this article as *SNT* (for "Snir with NT-like modifications"), is illustrated in Figure 6b, while the last row of Figure 5 illustrates the manner in which its import region scales with an increasing number of processors. Expressed in terms of the (as yet unspecified) box dimensions, and defining $S = b_z s$, the volume of the import region for the SNT method is

$$V_{i\mathrm{SNT}} = b_y \left(2b_z R + b_x \min(S - b_z, R) + 2 \int_0^{\min(S-b_z,R)} \sqrt{R^2 - z^2}\, dz \right) + b_x \left[2b_z R + \sum_{w_2=1}^{\left\lfloor \left(\frac{R+b_z}{S}\right) \right\rfloor} \left(b_y \min(b_z, R - w_2 S + b_z) + 2 \int_{w_2 S - b_z}^{\min(w_2 S, R)} \sqrt{R^2 - z^2}\, dz \right) \right].$$

The last row of Table 2 shows the data import time for the SNT method under the assumptions of our example molecular system, as calculated by global sampling over a wide range of b_x and b_y values and all meaningful values of s, followed by local minimization. It will be noted that, although NT outperforms SNT for each of the four example machine configurations, the degree of this outperformance grows smaller as the number of processors increases. In the limit as $p \to \infty$, the base becomes a $2R \times S \times b_y$ rectangular parallelepiped, the comb becomes a finely diced half-disk with radius R and thickness b_z whose "effective density" approaches $1/s$, and the import volume of the SNT method approaches

$$V_{i\mathrm{SNT}\infty} = 2\pi^{1/2} R^{3/2} V_b^{1/2},$$

which is identical to that of the NT method.

Conclusions

We have introduced a new technique, called the NT method, for the parallel evaluation of distance-limited pairwise particle interactions. Although the technique should be applicable to other types of simulations as well, it was designed with the goal of accelerating the execution of lengthy biomolecular simulations using classical molecular dynamics techniques on parallel computer systems ranging from conventional clusters to massively parallel supercomputers. Because the time required for interprocessor communication has historically been the principal obstacle to the efficient utilization of a large number of processors for this purpose, the results presented here may help to extend the period of time over which it is feasible to simulate the atomic-level dynamics of explicitly solvated biological macromolecules using MD techniques.

We have shown that the amount of data transferred into and out of a given processor during execution of the NT method scales as $O(R^{3/2}p^{-1/2})$, where R is the interaction radius and p is the number of processors. The method takes advantage of the fact that each atom need interact with only those atoms falling within a surrounding sphere of radius R to limit the import and export loads to a quantity dependent only on R, and not on the number of atoms in the system. Indeed, as p grows large, the amount of transferred data becomes proportional not to the *volume* of a roughly spherical interaction neighborhood of radius R, but to the *square root* of this volume. Additionally, the amount of data transferred into and out of a given processor may be made arbitrarily small with a sufficient increase in the number of processors, allowing a large number of processors to be productively employed on a single MD simulation, and avoiding the rapidly diminishing returns associated with traditional spatial decomposition methods. Most importantly, the constant factors associated with these performance figures are such that the NT method should provide substantial performance advantages by comparison with traditional methods in most cases of practical significance.

We have also compared the NT method with another new technique developed independently by Marc Snir that also achieves $O(R^{3/2}p^{-1/2})$ scaling, although with a less favorable multiplicative constant, and have described a set of modifications to his method, based on techniques analogous to those we employ in the NT method, that allow it to achieve performance approaching (although not equaling) that of the NT method for problems and machines of practical size. Although our results indicate that the NT method performs better than either the original or modified version of Snir's method for machine sizes ranging from 64 to 32K processors, its relative advantage declines as the number of processors becomes large. Indeed, we have shown that in the limit as $p \to \infty$, our modified version of Snir's method has exactly the same performance as the NT method.

It should be emphasized that the analyses presented in this article have considered only the *amount* of data transferred into a given processor, and not the fixed latency associated with the transfer of a minimal packet of data. The burden imposed by communication latency may, in fact, be quite significant in practice—particularly in the case of machines with a large number of processors—and a detailed latency analysis for parallel machines with various communication networks would thus provide a useful complement to the results presented here.

Acknowledgments

The author gratefully acknowledges the assistance of Ron Dror, Kevin Bowers, and Michael Eastwood, who reviewed the results reported in this article and made a number of helpful comments and suggestions.

References

1. Pande, V. S. Biopolymers 2003, 68, 91.
2. Ewald, P. P. Annalen Phys 1921, 64, 253.
3. Greengard, L.; Rokhlin, V. J Comput Phys 1987, 73, 325.
4. Ding, H. Q.; Karasawa, N.; Goddard, W. A. Chem Phys Lett 1992, 196, 6.
5. Ding, H. Q.; Karasawa, N.; Goddard, W. A. J Chem Phys 1992, 97, 4309.
6. Darden, T.; York, D.; Pedersen, L. J Chem Phys 1993, 98, 10089.
7. Hockney, R. W.; Eastwood, J. W. Computer Simulation Using Particles; Adam Hilger: Bristol, 1988.
8. Deserno, M.; Holm, C. J Chem Phys 1998, 109, 7678.
9. Deserno, M.; Holm, C. J Chem Phys 1998, 109, 7694.
10. Shan, Y.; Klepeis, J. L.; Eastwood, M. P.; Dror, R. O.; Shaw, D. E. J Chem Phys 2005, 122, 54101.
11. Almasi, G. S.; Cascaval, C.; Castanos, J. G.; Denneau, M.; Donath, W.; Eleftheriou, M.; Giampapa, M.; Ho, H.; Lieber, D.; Moreira, J. E.; Newns, D.; Snir, M.; Warren, H. S. Int J Parallel Prog 2002, 30, 317.
12. Snir, M. Theory Comput Syst 2004, 37, 295.
13. Plimpton, S. J Comput Phys 1995, 117, 1.
14. Heffelfinger, G. S. Comput Phys Commun 2000, 128, 219.
15. Fox, G. C.; Otto, S. W. Phys Today 1984, 37, 50.
16. Fincham, D. Mol Simul 1987, 1, 1.
17. Smith, W. Comput Phys Commun 1991, 62, 229.
18. Brooks, B. R.; Hodoscek, M. Chem Design Automat News 1992, 7, 16.
19. Clark, T. W.; McCammon, J. A.; Scott, L. R. Proc 5th SIAM Conference on Parallel Processing for Scientific Computing, 1992, 338.
20. Knirsch, R.; Hofmann, K. Lecture Notes Comput Sci 1992, 634, 829.
21. Smith, W. Theor Chim Acta 1993, 84, 385.
22. Scott, W.; Gunzinger, A.; Baumle, B.; Kohler, P.; Muller, U. A.; Vonder Muhll, H. R.; Eichenberger, A.; Guggenbuhl, W.; Ironmonger, N.; Muller–Plathe, F.; van Gunsteren, W. F. Comput Phys Commun 1993, 75, 65.
23. Nguyen, H. L.; Khanmohammadbaigi, H.; Clementi, E. J Comput Chem 1985, 6, 634.
24. Boyer, L. L.; Pawley, G. S. J Comput Phys 1988, 78, 405.
25. Brunet, J. P.; Mesirov, J. P.; Edelman, A. Proc Supercomputing 90, 1990, p. 748.
26. Windemuth, A.; Schulten, K. Mol Simulat 1991, 5, 353.
27. Skeel, R. D. J Comput Chem 1991, 12, 175.
28. Shifman, M. A.; Windemuth, A.; Schulten, K.; Miller, P. L. Comput Biomed Res 1992, 25, 168.
29. Schreiber, H.; Steinhauser, O.; Schuster, P. Parallel Comput 1992, 18, 557.
30. Brunet, J. P.; Mesirov, J. P.; Edelman, A. Siam J Sci Comput 1993, 14, 1143.
31. Plimpton, S.; Hendrickson, B. Int J Mod Phys C 1994, 5, 295.
32. Plimpton, S.; Hendrickson, B. J Comput Chem 1996, 17, 326.
33. Taylor, V. E.; Stevens, R. L.; Arnold, K. E. J Parallel Distrib Comput 1997, 45, 166.
34. Murty, R.; Okunbor, D. Parallel Comput 1999, 25, 217.
35. Rapaport, D. C; Clementi, E. Phys Rev Lett 1986, 57, 695.
36. Rapaport, D. C. Comput Phys Rep 1988, 9, 1.
37. Bruge, F. Comput Phys Commun 1990, 60, 31.
38. Pinches, M. R. S.; Tildesley, D. J.; Smith, W. Mol Simulat 1991, 6, 51.
39. Liem, S. Y.; Brown, D.; Clarke, J. H. R. Comput Phys Commun 1991, 1991, 261.
40. Giles, R.; Tamayo, P. Proc Scalable High Performance Computing Conference, 1992 (SHPCC-92), 1992, p. 240.
41. Gupta, S. Comput Phys Commun 1992, 70, 243.
42. Brown, D.; Clarke, J. H. R.; Okuda, M.; Yamazaki, T. Comput Phys Commun 1993, 74, 67.
43. Esselink, K.; Smit, B.; Hilbers, P. A. J. J Comput Phys 1993, 106, 101.
44. Kalia, R. K.; Nakano, A.; Greenwell, D.; Vashishta, P. Supercomputer 1993, 54, 10.
45. Hedman, F.; Laaksonen, A. Int J Mod Phys C 1993, 4, 41.
46. Lomdahl, P. S.; Tamayo, P.; Gronbech–Jensen, N.; Beazley, D. M. Proc Supercomputing 93, 1993, p. 520.
47. Clark, T. W.; Hanxleden, R. V.; McCammon, J. A.; Scott, L. R. Proc Scalable High Performance Computing Conference (SHPCC 94), 1994, p. 95.
48. Beazley, D. M.; Lomdahl, P. S. Parallel Comput 1994, 20, 173.
49. Eisenhauer, G.; Schwan, K. J Parallel Distrib Comput 1996, 35, 76.
50. Abraham, F. F. IEEE Comput Sci Eng 1997, 66.
51. Stadler, J.; Mikulla, R.; Trebin, H.-R. Int J Mod Phys C 1997, 8, 1131.
52. Srinivasan, S. G.; Ashok, I.; Jonsson, H.; Kalonji, G.; Zahorjan, J. Comput Phys Commun 1997, 102, 28.
53. Putz, M.; Kolb, A. Comput Phys Commun 1998, 113, 145.
54. McCoy, R. A.; Deng, Y. J of High Perform Comput Appl 1999, 13, 16.
55. Adiga, N. R.; Almasi, G.; Almasi, G. S.; Aridor, Y.; Barik, R.; Beece, D. K.; Bellofatto, R.; Bhanot, G.; Bickford, R.; Blumrich, M. A.; Bright, A. A.; Brunheroto, J.; Cascaval, C.; Castanos, J.; Chan, W.; Ceze, L.; Coteus, P.; Chatterjee, S.; Chen, D.; Chiu, G.; Cipolla, T. M.; Crumley, P.; Desai, A.; Deutsch, A.; Domany, T.; Dombrowa, M. B.; Donath, W.; Eleftheriou, M.; Erway, C. C.; Esch, J.; Fitch, B. G.; Gagliano, J.; Gara, A.; Garg, R.; Germain, R. S.; Giampapa, M.; Gopalsamy, B.; Gunnels, J. A.; Gupta, M.; Gustavson, F. G.; Hall, S.; Haring, R. A.; Heidel, D.; Heidelberger, P.; Herger, L. M.; Hoenicke, D.; Jackson, R. D.; Jamal–Eddine, T.; Kopcsay, G. V.; Krevat, E.; Kurhekar, M. P.; Lanzetta, A. P.; Lieber, D.; Liu, L. K.; Lu, M.; Mendell, M.; Misra, A.; Moatti, Y.; Mok, L.; Moreira, J. E.; Nathanson, B. J.; Newton, M.; Ohmacht, M.; Oliner, A.; Pandit, V.; Pudota, R. B.; Rand, R. A.; Regan, R. D.; Rubin, B.; Ruehli, A. E.; Rus, S.; Sahoo, R. K.; Sanomiya, A.; Schenfeld, E.; Sharma, M.; Shmueli, E.; Singh, S.; Song, P.; Srinivasan, V.; Steinmacher–Burow, B. D.; Strauss, K.; Surovic, C.; Swetz, R. A.; Takken, T.; Tremaine, R. B.; Tsao, M.; Umamaheshwaran, A. R.; Verma, P.; Vranas, P.; Ward, T. J. C.; Wazlowski, M. E.; Barret, W.; Engel, C.; Drehmel, B.; Hilgart, B.; Hill, D.; Kasemkhani, F.; Krolak, D.; Li, C. T.; Liebsch, T.; Marcella, J.; Muff, A.; Okomo, A.; Rouse, M.; Schram, A.; Tubbs, M.; Ulsh, G.; Wait, C.; Wittrup, J.; Bae, M.; Dockser, K.; Kissel, L.; Seager, M. K.; Vetter, J. S.; Yates, K. Proc 2002 ACM/IEEE Conference on Supercomputing (SC 2002), Baltimore, MD, 2002, p. 1.