Novel Force Matrix Transformations with Optimal Load-Balance for 3-body Potential based Parallel Molecular Dynamics in a Heterogeneous Cluster Environment

Sumanth J.V  
*University of Nebraska-Lincoln*, sumanth@cse.unl.edu

David Swanson  
*University of Nebraska-Lincoln*, dswanson@cse.unl.edu

Hong Jiang  
*University of Nebraska-Lincoln*, jiang@cse.unl.edu

Follow this and additional works at: [https://digitalcommons.unl.edu/csetechreports](https://digitalcommons.unl.edu/csetechreports)

Part of the Computer Sciences Commons

J.V, Sumanth; Swanson, David; and Jiang, Hong, "Novel Force Matrix Transformations with Optimal Load-Balance for 3-body Potential based Parallel Molecular Dynamics in a Heterogeneous Cluster Environment" (2007). *CSE Technical reports. 67.*  
[https://digitalcommons.unl.edu/csetechreports/67](https://digitalcommons.unl.edu/csetechreports/67)

This Article is brought to you for free and open access by the Computer Science and Engineering, Department of at DigitalCommons@University of Nebraska - Lincoln. It has been accepted for inclusion in CSE Technical reports by an authorized administrator of DigitalCommons@University of Nebraska - Lincoln.
Novel Force Matrix Transformations with Optimal Load-Balance for 3-body Potential based Parallel Molecular Dynamics in a Heterogeneous Cluster Environment

Sumanth J.V, David Swanson, Hong Jiang
Department of Computer Science and Engineering
University of Nebraska-Lincoln, U.S.A
{sumanth, dswanson, jiang }@cse.unl.edu

Abstract

Evaluating the Force Matrix constitutes the most computationally intensive part of a Molecular Dynamics (MD) simulation. In three-body MD simulations, the total energy of the system is determined by the energy of every unique triple in the system and the force matrix is three-dimensional. The execution time of a three-body MD algorithm is thus proportional to the cube of the number of atoms in the system. Fortunately, there exist symmetries in the Force Matrix that can be exploited to improve the running time of the algorithm. While this optimization is straight forward to implement in the case of sequential code, it has proven to be nontrivial for parallel code even in a homogeneous environment.

In this paper, we present two force matrix transformations that are capable of exploiting the symmetries in a 3-body force matrix in both a homogeneous and a heterogeneous environment while balancing the load among all the participating processors. The first transformation distributes the number of interactions to be computed uniformly among all the slices of the force matrix along any of the axes. The transformed matrix can be scheduled using any well known heterogeneous slice-level scheduling technique. The second transformation distributes interactions to be computed uniformly over the entire volume of the force matrix allowing us to perform a block decomposition of the force matrix. The transformed force matrix can be scheduled by any block level scheduling algorithm. We also derive theoretical bounds for efficiency and load balance for prior work in the literature. We then prove some interesting and useful properties of our transformations and evaluate their advantages and disadvantages. A loop reordering optimization for our transformations is also described. The performance of an MPI implementation of the transformations is studied in terms of the Step Time Variation Ratio (STVR) in a homogeneous and heterogeneous environment.

1 Introduction

Molecular Dynamics (MD) is a powerful technique used to obtain static or dynamic properties of liquids and solids. It can be more formally defined as a computer simulation technique where the time evolution of a set of interacting atoms is followed by integrating their equations of motion [1]. From the motion of the ensemble of atoms, a variety of useful microscopic and macroscopic information can be extracted such as transport coefficients, phase diagrams, and structural properties. The physics of the model is contained in a potential energy functional for the system from which the individual force equations for each atom are derived. There are numerous applications for MD simulations in diverse fields of science and technology such as chemistry, astronomy, biophysics, solid-state physics, material science and fluid dynamics, to mention a few.

MD simulations are not very memory intensive. Their space complexity grows linearly with the number of atoms being simulated. However, their time complexity grows cubically with the number of atoms being simulated (assuming a 3-body potential is being used). Being a very computationally intensive application [10], various solutions to improve execution times have been investigated. The most common methods for improving performance are parallelization [3] and using custom-designed special purpose hardware [13] [12].
In this paper, we describe two force matrix transformations that allow for the parallelization of a 3-body MD simulation and taking advantage of symmetries in the force matrix in a homogeneous as well as a heterogeneous cluster environment while ensuring an optimal load balance. Our first transformation, the Slice-Symmetric transformation is to be used in conjunction with the atom-decomposition MD algorithm [9]. Our second transformation, the Volume-Symmetric transformation is to be used with the force-decomposition MD algorithm [9]. To the best of our knowledge, the Volume-Symmetric transformation is the only way to exploit the symmetries in the force-matrix of a parallel 3-body MD simulation. We focus on the 3-body component of the Webber-Stillinger potential [11]. The load balancing properties of our transformations are mathematically analyzed. A detailed analysis of previous work in this area by Li et al. [8] is performed and closed form efficiency upper bounds for their approach is determined. However, their cyclic distribution works only with the atom-decomposition MD algorithm on a homogeneous cluster. Our technique can be used in a heterogeneous and homogeneous environment. Our first transformation can be used only with the atom-decomposition technique and the second transformation can work with both the atom- and force-decomposition MD algorithms. We also, discuss loop optimization techniques that can be used to implement our transformation. Performance is evaluated using the Step Time Variation Ratio (STVR)[8]. In the homogeneous case, we compare our technique to the cyclic distribution technique.

The rest of this paper is organized as follows: in Section 2, we give an overview of the computational aspects of an MD simulation and describe various parallelization techniques. In Section 3, we describe prior work in this area. Section 4 reviews an existing technique and derives theoretical efficiency bounds for it. In Section 5, our Slice-Symmetric transformation and some of its useful properties are described and proved. Section 6 describes our Volume-Symmetric transformation and derives its load balancing properties. Section 7 describes a loop optimization technique for the implementation of our transformation. We evaluate the performance of our technique and discuss the pros and cons of our approach in Section 8 and conclude the paper in Section 9.

2 Molecular Dynamics

2.1 Computational Aspects

The computational task in an MD simulation is to perform the time integration of a set of coupled differential equations (Newton’s equations) given by

\[ m_i \frac{\partial^2 \vec{r}_i}{\partial t^2} = \sum_j F_2(\vec{r}_i, \vec{r}_j) + \sum_j \sum_k F_3(\vec{r}_i, \vec{r}_j, \vec{r}_k) + \ldots \]

where, \( m_i \) is the mass of atom \( i \), \( \vec{r}_i \) is its position vector, \( F_2 \) is a two-body force function and \( F_3 \) is a three-body force function. In this paper, we perform the time-integration of Equation 2.1 using the velocity-verlet algorithm [15]. The force functions \( F_2 \) and \( F_3 \) are computed as the negative gradients of potential functions in which the energy of atom \( i \) is typically written as a function of the positions of itself and other atoms.

For concreteness, for the 3-body form we use the popular Weber-Stillinger potential. [11]. The total potential is expressed as two sums, one for unique pair interactions, and another for unique triplet interactions.

\[ U = \sum_{i<j} v_2(r_{ij}) + \sum_{i<j<k} v_3(\vec{r}_i, \vec{r}_j, \vec{r}_k) \]

The two-body component models the bonds and is given by:

\[ v_2(r) = \begin{cases} 
\epsilon A (B r^{-p} - c r^{-q}) \exp \left[ \left( \frac{r - a}{\lambda} \right)^{-1} \right], & r < a \\
0, & r \geq a
\end{cases} \]

where, \( A, B, p, q, a, \lambda \) and \( \gamma \) are experimentally determined parameters. See [11] for experimentally determined values for Silicon.

It is very much like a LJ potential [1], only with different exponents and a smooth cutoff. The three body component models the angles, and is the sum of functions of each of the three angles of a triplet, \( ijk \):

\[ v_3(\vec{r}_i, \vec{r}_j, \vec{r}_k) = h_{ijk} + h_{ijk} + h_{ikj} \]
where, \( h_{ijk} = h(r_{ij}, r_{ik}, \theta_{ijk}) \), \( \theta_{ijk} \) is the angle subtended at \( r_i \) and \( \cos \theta_{ijk} = -\frac{1}{2} \). If \( r \geq a \), \( h_{ijk} = 0 \). \( h_{ijk} \) is determined by equation 1 when \( r < a \).

\[
h_{ijk} = \epsilon \lambda \left( \frac{\gamma}{r_{ij} - a} + \frac{\gamma}{r_{ik} - a} \right) \left( \cos \theta_{ijk} + \frac{1}{3} \right)^2
\]

(1)

\( h_{ijk} \) and \( h_{ikj} \) can be computed by permuting the indices appropriately.

The force on any particle \( i \) is computed as:

\[
f_i = -\nabla_{\vec{r}_i} U
= -\sum_{j \neq i} \nabla_{\vec{r}_i} v_2(r_{ij}) - \sum_{j \neq i, k \neq i, j} \nabla_{\vec{r}_i} v_3(\vec{r}_i, \vec{r}_j, \vec{r}_k)
\]

where,

\[
\nabla_{\vec{r}_i} v_2(r_{ij}) = \frac{\partial}{\partial r_{ij}} \{ \epsilon A (B r_{ij}^{-p} - r_{ij}^{-q}) \exp [(r_{ij} - a)^{-1}] \}
\]

\[
\nabla_{\vec{r}_i} v_3(\vec{r}_i, \vec{r}_j, \vec{r}_k) = \left( \frac{\partial h_{ijk}}{\partial r_{ij}} + \frac{\partial h_{ijk}}{\partial r_{ik}} + \frac{\partial h_{ikj}}{\partial r_{ij}} \right)
\]

\( \frac{\partial h_{ijk}}{\partial r_{ij}} \) is computed by the following equation.

\[
\frac{\partial h_{ijk}}{\partial r_{ij}} = -\gamma h_{ijk} \left[ \frac{\vec{r}_{ij}}{r_{ij}} \frac{1}{r_{ij}^2} + \frac{\vec{r}_{ik}}{r_{ik}} \frac{1}{r_{ik}^2} \right]
+ 2\lambda \exp \left( \frac{\gamma}{r_{ij} - a} + \frac{\gamma}{r_{ik} - a} \right)
\times \left( \cos \theta_{ijk} + \frac{1}{3} \right)
\times \left( \frac{\vec{r}_{ij}}{r_{ij} r_{ik}} + \frac{\vec{r}_{ik}}{r_{ik} r_{ij}} - \frac{\vec{r}_{ij} + \vec{r}_{ik}}{r_{ij} r_{ik}} \right) \cos \theta_{ijk}
\]

\( \frac{\partial h_{ikj}}{\partial r_{ij}} \) and \( \frac{\partial h_{ikj}}{\partial r_{ik}} \) can be similarly computed. When using a three-body potential, the three-dimensional force matrix is symmetric since \( F_{ijk} = F_{ikj} \), where \( F_{ijk} \) is the \((i, j, k)\)th element of the force matrix \( F \) representing the force exerted on atom \( i \) by atoms \( j \) and \( k \).

For a system with \( N \) particles, the total number of unique triples to be evaluated is \( N(N-1)(N-2)/6 \). A triple \((i, j, k)\), where \( i \neq j \neq k \) is contained in 6 force elements. Each single particle can be involved in \( (N-1)(N-2)/2 \) triples, therefore the evaluation of the total 3-body force acting on a particle requires a sum over all the \( (N-1)(N-2)/2 \) triples. However, due to the symmetry described above, only three independent force elements are actually evaluated for each triple.

### 2.2 Classification

Many-body simulations can also be classified by whether they use direct or indirect methods. Simulations that compute each interaction of Equation 2.1 explicitly use what are known as direct methods. Conversely, if the simulation approximates some interactions, it is known as an approximate method. Approximate methods such as particle-mesh algorithms, hierarchical algorithms and fast multipole methods are typically used with long-range forces. However, approximate methods are much harder to implement than direct methods, particularly in parallel machines and systems [6]. Because of this complexity, approximate methods are typically not faster than direct methods until the number of atoms reaches a certain threshold value which can be quite large. In parallel implementations, the performance of approximate methods can suffer further from the fact that the work load can be difficult to balance among processors when the particle density is spatially and/or temporally non-uniform leading to underutilization of available resources.
2.3 Parallelization

The most common parallelization techniques of the direct methods are atom-, force- and spatial-decomposition. Atom-decomposition involves assigning the force computations of a subgroup of atoms to each processor. Force-decomposition generalizes this approach by assigning a subset of the force loops to each processor. Both of these decompositions are analogous to Lagrangian gridding in a fluids simulation where the grid cells move with the fluid. Further, in the above two techniques, the assignments of atoms to processors remain fixed throughout the entire simulation. Atom-decomposition requires the entire position array to be present at all the processors. Force-decomposition requires only a subset of the position array to be present at each processor leading to better scaling of the communication requirements.

On the other hand, spatial-decomposition works by assigning a portion of the physical simulation domain to each processor. Processors exchange atoms as they move from one domain into the other. This is analogous to Eulerian gridding in a fluids simulation where the grid remains fixed in space as fluid moves through it. Spatial-decomposition performs best when used with a short-range potential, a close-to-uniform spatial distribution of atoms and moderate to low densities. Atom- and force-decomposition perform the same irrespective of the above conditions. Plimpton [9] describes these methods in more detail and compares their relative performance in the case of a 2-body potential function. The techniques developed in this paper applies to short-range and long-range potentials evaluated using the atom- and force-decomposition techniques.

3 Related Work

Plimpton et al [9] [6] have determined a force matrix transformation that exploits the symmetry in a 2-body force matrix. The two-body component of the Webber-Stillinger potential can use this transformation to exploit symmetries in the 2D force matrix. Our technique is to be used for the 3-body component of the potential.

Li et al. [8] perform a cyclic decomposition of the force matrix when performing Atom-Decomposition. Li et al. refer to their technique as force decomposition, however they perform a slice level decomposition. In this paper, we use the terminology used by Plimpton et al. [9], hence consider Li et al.’s work to be an Atom-Decomposition technique instead of Force-Decomposition. We analyze their transformation and determine closed form theoretical upper bounds for efficiency and relative load imbalance when using their technique.

4 Cyclic Distribution

The cyclic distribution technique [8] decomposes the 3D force matrix into slices along one dimension. A triple \((i, j, k)\) is evaluated only if \(i < j < k\). This gives rise to a force matrix that looks like Figure 1. The interactions to be computed in the force matrix are in the shape of a tetrahedron. It is evident from the figure that lower slices of the matrix evaluate a larger number of tuples than higher slices. In a homogeneous cluster with \(P\) processors, the \(i^{th}\) slice is computed by the processor with rank \(\text{mod}(i, P)\). This strategy attempts to balance the load among the available processors in a homogeneous cluster environment.

It is obvious that the cyclic distribution technique assigns processors with lower id’s a slightly larger load than processors with higher id’s. We quantify this imbalance in section 4.1. This imbalance can be seen in Figure 2.

4.1 Analysis of the Cyclic Distribution

We now derive some interesting properties of the cyclic distribution. For the rest of this paper, we assume that \(P\) divides \(N\).

**Lemma 4.1** If the cyclic distribution is used with the atom-decomposition MD algorithm to schedule an \(N \times N \times N\) 3-body force matrix onto a collection of \(P\) homogeneous processors, \(I_P\), the number of interactions evaluated by processor \(X, 0 \leq X < P\), is given by \(K(N, P) - \frac{XN}{2P} (N + P - 3 - X)\), where \(K(N, P) = \frac{N}{12P} (N(2N - 9) + P(P + 3N - 9) + 12)\).

**Proof** \(n_m\), the number of interactions in slice \(m, 0 \leq m < P\) is given by equation 2

\[
n_m = \frac{(N - m - 1)(N - m - 2)}{2}
\] (2)
Each processor receives exactly \( N/P \) slices. Processor \( X, 0 \leq X < P \) receives slices with indices \( X, X + P, X + 2P, \ldots, X + (N/P - 1) P \). Combining this with Equation 2, \( I_P \), the number of interactions computed by processor \( X, 0 \leq X < P \) is

\[
I_P = \sum_{j=0}^{(N/P) - 1} \frac{(N - 1 - X - jP)(N - 2 - X - jP)}{2} \\
= \frac{N}{P} \left[ N(N - 2X - 3) + X(X + 3) + 2 \right] \\
+ \left[ 3P + 2PX - 2NP \right] \cdot \sum_{j=0}^{(N/P) - 1} j \\
+ P^2 \cdot \sum_{j=0}^{(N/P) - 1} j^2 \\
= \frac{N}{12} \cdot \frac{P}{N} \left[ (N(2N - 9) + P(P + 3N - 9) + 12) \\
- \frac{XN}{2P} (N + P - 3 - X) \right] \\
= K(N, P) - \frac{XN}{2P} (N + P - 3 - X)
\]

**Definition** The Absolute Load Imbalance (ALI) is defined as the difference between the maximum number of interactions assigned to any processor and the minimum number of interactions assigned to any processor.

**Definition** The Relative Load Imbalance (RLI) is defined as the ALI divided by the optimal load assignment. The optimal load assignment for a system with \( N \) atoms executing on \( P \) processors using a 3-body potential is \( \frac{N(N - 1)(N - 2)}{6P} \).

**Theorem 4.2** *If the cyclic distribution is used with the atom-decomposition MD algorithm to schedule an \( N \times N \times N \) 3-body force matrix onto a collection of \( P \) homogeneous processors, the RLI is \( \frac{3(P - 1)}{N} \).*

**Proof** When using the cyclic distribution, the maximum load is at processor 0 and the minimum load is at processor \( P - 1 \). Using lemma 4.1, the ALI can be determined as

\[
\text{ALI} = I_0 - I_{P-1} \\
= \left[ K(N, P) - 0 \right] \\
- \left[ K(N, P) - \frac{N}{2P} (P - 1)(N - 2) \right] \\
= \frac{N}{2P} (P - 1)(N - 2)
\]

The RLI can now be determined as

\[
\text{RLI} = \frac{6 \cdot P \cdot \text{ALI}}{N(N - 1)(N - 2)} \\
= \frac{3(P - 1)}{N - 1}
\]

Figure 3 depicts the effect of varying \( N \) and \( P \) on the RLI. It can be seen that for a fixed number of atoms, the RLI increases with an increase in number of processors. On the other hand, for a fixed number of processors, an increase in the number of atoms causes a decrease in the RLI.
Definition  Efficiency of a parallel homogeneous system is defined as

\[ \eta = \frac{t_{\text{seq}}}{P \cdot t_{\text{par}}} \]

where, \( t_{\text{seq}} \) is the execution time of the entire simulation on a single processor and \( t_{\text{par}} \) is the execution time of the entire simulation using \( P \) processors.

Theorem 4.3  If the cyclic distribution is used with the atom-decomposition MD algorithm to schedule an \( N \times N \times N \) 3-body force matrix onto a collection of \( P \) homogeneous processors, the upper bound for efficiency (\( \eta' \)) due to load imbalance is given by \( \frac{N(N-1)(N-2)}{6K(N,P)} \), where \( K(N,P) = \frac{N}{12P}(N(2N-9)+P(3N-9)+12) \).

Proof  Assuming that the time it takes to evaluate a single interaction on any of the processors is \( t_{\text{exec}} \), the sequential execution time can be determined as

\[ t_{\text{seq}} = \frac{N(N-1)(N-2)}{6} \cdot t_{\text{exec}} \]  

(3)

The parallel execution time is determined by the execution time on the processor that is assigned the maximum load since this forces the remaining processors to be idle toward the end of the time step. Processor 0 is the one with the maximum assigned load and its load is given by \( I_0 \). Hence, the parallel execution time can be determined as

\[ t_{\text{par}} = K(N,P) \cdot t_{\text{exec}} \]  

(4)

From equations 3 and 4, the upper bound for efficiency can be determined as

\[ \eta' = \frac{N(N-1)(N-2)}{6 \cdot P \cdot K(N,P)} \]  

(5)

It must be noted that the efficiency upper bound determined in Theorem 4.3 is only due to load imbalance. In practice, the bound will be lower due to communication overhead. Figure 4 depicts the effect of varying \( N \) and \( P \) on the above determined efficiency upper bound \( \eta' \). It can be seen that for a fixed number of processors, increasing the number of atoms improves \( \eta' \). For a fixed number of atoms, increasing the number of processors decreases the efficiency.

5  Slice-Symmetric Transformation

In this section, we propose a novel transformation that assigns an equal number of interactions to each slice of the force matrix. The transformed matrix \( F' \) is constructed as follows

\[ F'_{ijk} = \begin{cases} F_{ijk}, & (i > j > k) \land (i + j + k \equiv 1 \text{ mod } 3) \\ F_{ijk}, & (j > k > i) \land (i + j + k \equiv 2 \text{ mod } 3) \\ F_{ijk}, & (k > i > j) \land (i + j + k \equiv 0 \text{ mod } 3) \\ 0, & \text{ otherwise} \end{cases} \]  

(6)

Only the non-zero elements of the transformed force matrix are to be evaluated in the force evaluation routine. The correctness of this transformation is proved in Theorem 5.1. The modulo component of the transformation acts like a 3D checkerboard pattern with three colors as illustrated in Figure 5. Assume that a cell \((i, j, k)\) is either blue(Bl), green(Gr) or brown(Br) depending on whether \( i + j + k \equiv 0, 1, 2 \text{ (mod } 3) \) respectively. Row 0 of Slice 0 of the board will be of the form BlGrBrBlGrBr..., Row 1 of Slice 0 will be of the form GrBrBlGrBr..., Row 3 of Slice 0 will be of the form BrBlGrBr..., Row 4 of Slice 0 will be of the form BlGrBrBl..., Row 0 of Slice 1 will be of the form GrBrBlGrBr..., and so on.

A cube can be packed with 6 suitably sized tetrahedrons. The inequalities in each of the cases of the transformation select 3 of these tetrahedrons as illustrated in Figure 8. The spacing between the cells constituting the tetrahedrons is due to the modulo function. Figures 9 and 6 illustrate the interactions that are computed in slices (parallel to the \( jk \) plane) of the transformed force matrix. Each case of Equation 6 is colored differently. The plane intersects two of the tetrahedrons such that the intersecting part of the plane with each of the tetrahedrons forms a triangle (case 1 and 2 of Equation 6). The third tetrahedron (case 3 of Equation 6) is cut such that the intersecting part of the plane forms a rectangle. The cutting of the tetrahedron to form a rectangle and triangle is shown in Figure 7. As we move along the \( i \) axis, the areas of the triangles and
squares change but the sum of their areas remains constant (Figure 6). This ensures that irrespective of where a slice is made, the number of interactions in the slice is constant. This property is true even if the slicing is performed along either of the other two axes. In Figure 9, the blue, green and red elements of the slices correspond to the first, second and third cases in Equation 6. From now on, we refer to the interactions as blue, green or red depending on whether they satisfy cases 1, 2 or 3 of Equation 6. The triangles and the rectangles get closest to each other at the focal point \(i = j = k\) i.e. (i, i) on the \(jk\) plane. As the height of the slice increases, the focal point moves along the \(jk\) diagonal toward the \((N - 1, N - 1)\) point on the slice (Figure 6). We refer to the intersection of the diagonal \(i = j = k\) of the cube with the slice as the focal point.

5.1 Slice-Symmetric Transformation Properties

**Theorem 5.1** Evaluating the non-zero elements of a Slice-Symmetrically transformed \(N \times N \times N\) 3-body force matrix results in computing exactly all the required force components of a 3-body potential.

**Proof** For any tuple \((i, j, k), 0 \leq i, j, k < N\) and all its 6 permutations \((i, j, k), (i, k, j), (j, k, i), (j, i, k), (k, i, j), (k, j, i)\), the value of \(i + j + k(\text{mod } 3)\) is the same. Combining \(i + j + k(\text{mod } 3)\) with the inequality allows us to select exactly one of the permutations as long as \(i, j\) and \(k\) are distinct.

The inequalities in the transformation Equation 6 ensures that no triples of the form \((i, j, b), (i, k, j)\) or \((j, k, i)\) are evaluated. This is essential since in a 3-body potential, \(F_{iij} = 0, 0 \leq i < N\). The inequalities also ensure that no triples of the form \((i, j, b), (i, j, i)\) or \((j, j, i)\) are evaluated. This is also essential since in a 3-body potential, \(F_{iij} = 0, F_{ji} = 0\) and \(F_{jii} = 0, 0 \leq i, j < N\). This ensures that \(i, j\) and \(k\) are distinct.

Thus, the symmetric transformation causes the evaluation of only the unique triples in the original force matrix.

**Lemma 5.2** In an \(N \times N \times N\) Slice-Symmetrically transformed 3-body force matrix, the number of blue, green and red interactions in slice \(m, 0 \leq m < N\) along the \(i\) axis are \(\left\lfloor \frac{m(m-1)}{6} \right\rfloor\), \(\left\lfloor \frac{(N-1-m)(N-2-m)}{6} \right\rfloor\) and \(\left\lfloor \frac{m(N-1-m)}{3} \right\rfloor\) respectively.

**Proof** The expressions in the Theorem are quite intuitive except for the floors and the ceilings. The number of blue interactions in slice \(m, 0 \leq m < N\) is determined by the area below the diagonal in the square with corners \((0, 0)\) and the focal point \((m, m)\) on the \(jk\) plane which is \(\frac{m(m-1)}{2}\). Since the modulo function considers only a third of the elements, the area is to be divided by 3 leading to the expression \(\frac{m(m-1)}{6}\). The expression for the other triangle can be similarly derived by considering the area below the square with corners \((0, N - 1)\) and the focal point \((m, m)\).

The number of red interactions in slice \(m, 0 \leq m < N\) along the \(i\) axis is determined by the area of the rectangle with the focal point as one corner and \((0, N - 1)\) as the other corner, which is \(\frac{m(N-1-m)}{2}\). After applying the modulo function, this becomes \(\frac{m(N-1-m)}{6}\).

To determine the exact number of each type of interactions, a more careful analysis is required. The number of blue interactions \(b_m\) can be determined considering the following three possible cases:

- **\(m \equiv 0 \text{ mod } 3\)**
  This case corresponds to \(jk\) planes that have checkerboard patterns such that row 0 is of the form BlGrBrBlGrBr....
  The number of blue interactions \(b_m\) in slice \(m, 0 \leq m < N\) is equal to the number of Gr squares below the diagonal in a square of side \(m\) with corners \((0, 0)\) and \((m, m)\). In this case, it is trivial to see that \(b_m = \frac{m(m-1)}{6}\).

- **\(m \equiv 1 \text{ mod } 3\)**
  This case corresponds to \(jk\) planes that have checkerboard patterns such that row 0 is of the form GrBrBlGrBrBl....
  We now count the number of Gr squares below the diagonal in a square of side \(m\) with corners \((0, 0)\) and \((m, m)\). The number of Gr squares in rows \(0, 3, 6, \ldots, (m-1)\) is \(\left\lfloor \frac{m}{3} \right\rfloor\), in rows \(1, 4, 7, \ldots, (m-3)\) is \(\left\lfloor \frac{m}{3} \right\rfloor\) and rows \(2, 5, 8, \ldots, (m-2)\) is \(\left\lfloor \frac{m}{3} \right\rfloor\). Let \(rid\) be the row id. The number of rows such that \(rid \equiv 0 \text{ mod } 3\) is \(\left\lfloor \frac{m}{3} \right\rfloor\), \(rid \equiv 1 \text{ mod } 3\) is \(\left\lfloor \frac{m}{3} \right\rfloor\), \(rid \equiv 2 \text{ mod } 3\) is \(\left\lfloor \frac{m}{3} \right\rfloor\). The total number of Gr squares is now given by

  \[
  Gr_{tot} = \left\lfloor \frac{m}{3} \right\rfloor^2 + 2 \left\lfloor \frac{m}{3} \right\rfloor + 2 \left\lfloor \frac{m-1}{3} \right\rfloor^2
  \]
For the current checkerboard pattern, the number of Gr squares along the diagonal \( G_{r_{\text{diag}}} = \left\lceil \frac{m}{3} \right\rceil = \frac{m+2}{3} \). The number of non-diagonal Gr squares \( G_{r_{\text{ndiag}}} \) is given by

\[
G_{r_{\text{ndiag}}} = G_{r_{\text{tot}}} - G_{r_{\text{diag}}}
= \frac{m(m-1)}{3}
\]

Since the blue interactions correspond to the elements below the diagonal,

\[
b_m = \frac{G_{r_{\text{ndiag}}}}{2} = \frac{m(m-1)}{6}
\]

- \( m \equiv 2 \mod 3 \)

Using the same approach as in the above case, it can be shown that

\[
G_{r_{\text{tot}}} = 2 \left\lfloor \frac{m}{3} \right\rfloor \cdot \left\lceil \frac{m}{3} \right\rceil + \left\lceil \frac{m}{3} \right\rceil^2
\]

\[
G_{r_{\text{diag}}} = \left\lceil \frac{m}{3} \right\rceil
\]

\[
G_{r_{\text{diag}}} = \frac{m(m-1) - 2}{3}
\]

\[
b_m = \frac{m(m-1) - 2}{6}
\]

Combining the above three cases, \( b_m = \left\lfloor \frac{m(m-1)}{6} \right\rfloor \). Using a similar counting technique, the number of red interactions, \( r_m \) and green interactions \( g_m \) in slice \( m \) can be determined.

**Theorem 5.3** In an \( N \times N \times N \) Slice-Symmetrically transformed 3-body force matrix, the number of interactions to be computed are uniformly distributed among the slices of the force matrix along any axis.

**Proof** The total number of interactions \( n_m \) in any slice \( m, 0 \leq m < N \) along the \( i \)-axis is the sum of the number of blue, green and red interactions in that slice i.e. \( n_m = b_m + g_m + r_m \). The floors in the expressions for \( b_m, g_m \) and \( r_m \) can be eliminated and written as follows

\[
b_m = \begin{cases} 
\frac{m(m-1)}{6}, & m \not\equiv 2 \mod 3 \\
\frac{m(m-2)}{6}, & \text{otherwise}
\end{cases}
\]

\[
r_m = \begin{cases} 
\frac{m(N-1-m)+1}{3}, & (N \not\equiv 1 \mod 3) \\
\frac{m(N-1-m)+2}{3}, & [(N \not\equiv 0 \mod 3) \\
\quad \land (m \not\equiv 0 \mod 3)] \\
\frac{m(N-1-m)}{3}, & \land [(N \not\equiv 0 \mod 3) \\
\quad \land (m \not\equiv 1 \mod 3)]
\end{cases}
\]

\[
g_m = \begin{cases} 
\frac{(N-1-m)(N-2-m)-2}{6}, & N \not\equiv i \mod 3 \\
\frac{(N-1-m)(N-2-m)}{6}, & \text{otherwise}
\end{cases}
\]

Adding the above conditions gives us Equation 7.

\[
n_m = \begin{cases} 
\frac{(N-1)(N-2)}{6}, & N \not\equiv 0 \mod 3 \\
\frac{N(N-3)}{6}, & (N \equiv 0 \mod 3) \\
\frac{N(N-3)}{6} + 1, & (N \equiv 0 \mod 3) \\
\quad \land (m \not\equiv 1 \mod 3)
\end{cases}
\]

Equation 7
This demonstrates that the number of interactions in any slice is independent of the slice number when \( N \not\equiv 0 \mod 3 \). When \( N \equiv 0 \mod 3 \), every third slice beginning with slice 1 has exactly one interaction more than the slices above and below it. Given the divisibility constraints, this is the optimal uniform distribution of interactions among the slices for all values of \( N \). Similarly, it can be shown that the interactions are uniformly distributed among the slices along the other two axes.

**Corollary 5.4** In an \( N \times N \times N \) Slice-Symmetrically transformed 3-body force matrix, the total number of blue \((N_b)\), green \((N_g)\) and red interactions \((N_r)\) is given by

\[
N_b = \begin{cases} 
\frac{N(N-2)^2}{18}, & N \equiv 0 \mod 3 \\
\frac{N(N-1)(N^2-2N+2)}{18}, & N \equiv 1 \mod 3 \\
\frac{(N+1)(N-2)^2}{18}, & N \equiv 2 \mod 3 
\end{cases}
\]

\[
N_g = \begin{cases} 
\frac{N(N^2-3N+6)}{18}, & N \equiv 0 \mod 3 \\
\frac{(N-1)(N^2-2N+4)}{18}, & N \equiv 1 \mod 3 \\
\frac{(N-2)(N^2-N+4)}{18}, & N \equiv 2 \mod 3 
\end{cases}
\]

\[
N_r = \begin{cases} 
\frac{N(N^2-3N+6)}{18}, & N \equiv 0 \mod 3 \\
\frac{(N-1)(N^2-2N+4)}{18}, & N \equiv 1 \mod 3 \\
\frac{(N-2)(N^2-N+4)}{18}, & N \equiv 2 \mod 3 
\end{cases}
\]

**Proof** The total number of blue interactions can be determined by summing the number of blue interactions in each slice.

\[
N_b = \sum_{i=0}^{N-1} \left\lfloor \frac{i(i-1)}{6} \right\rfloor 
\]

\[
= \left( \frac{0^2}{6} - \frac{0}{6} \right) + \left( \frac{1^2}{6} - \frac{1}{6} \right) + \frac{2^2}{6} - \frac{2}{6} + \frac{1}{3} \ldots 
\]

\[
= \frac{1}{6} \sum_{i=0}^{N-1} i^2 - \frac{1}{3} \left( \sum_{i=0}^{N-1} i \right) 
\]

\[
= \frac{N(N-2)^2}{18}, \quad N \equiv 0 \mod 3 \\
\frac{(N-1)(N^2-2N+2)}{18}, \quad N \equiv 1 \mod 3 \\
\frac{(N+1)(N-2)^2}{18}, \quad N \equiv 2 \mod 3 
\]

By symmetry, \( N_b = N_g \). The total number of red interactions can be determined using the expression for the number of unique triples as

\[
N_r = \frac{N(N-1)(N-2)}{6} - 2 \cdot N_b 
\]

\[
= \begin{cases} 
\frac{N(N^2-3N+6)}{18}, & N \equiv 0 \mod 3 \\
\frac{(N-1)(N^2-2N+4)}{18}, & N \equiv 1 \mod 3 \\
\frac{(N-2)(N^2-N+4)}{18}, & N \equiv 2 \mod 3 
\end{cases}
\]

**5.2 Load distribution**

From Theorem 5.3, it is evident that when using a Slice-Symmetrically transformed force matrix with the atom-decomposition MD algorithm, the computational load assigned to each processor is perfectly balanced when using the atom-decomposition algorithm. This implies that when using a Slice-Symmetrically transformed force matrix, the primary factor affecting the parallel efficiency is communication overhead.

The cyclic distribution assigns slices that are separated by a distance of \( P \) slices to each processor, where \( P \) is the number of processors. It also assumes a homogeneous cluster environment. On the other hand, when using the symmetric distribution with the atom-decomposition algorithm, we can assign contiguous slices of the force matrix to the processors. This allows us to arbitrarily slice up the force matrix when using the symmetric distribution. This is an important property that allows us to use the symmetric distribution in a heterogeneous environment where the number of slices assigned to each processor is not always \( N/P \).


6 Volume-Symmetric Distribution

The Slice-Symmetric transformation described in Section 5 distributes the total number of interactions to be computed among the slices of the force matrix. However, the interactions to be computed in a slice are not uniformly distributed spatially as illustrated in Figure 6. Due to this, the interactions are not uniformly distributed over the volume of the force matrix. In this section, we extend our Slice-Symmetric distribution to distribute the interactions uniformly over the volume of the force matrix. The transformation is described by Equation 8

\[
F'_{ijk} = \begin{cases} 
F_{ijk}, & (i > j > k) \land (i + j + k \equiv 0 \text{ mod 6}) \\
F_{ijk}, & (i > k > j) \land (i + j + k \equiv 1 \text{ mod 6}) \\
F_{ijk}, & (j > k > i) \land (i + j + k \equiv 2 \text{ mod 6}) \\
F_{ijk}, & (j > i > k) \land (i + j + k \equiv 3 \text{ mod 6}) \\
F_{ijk}, & (k > i > j) \land (i + j + k \equiv 4 \text{ mod 6}) \\
F_{ijk}, & (k > j > i) \land (i + j + k \equiv 5 \text{ mod 6}) \\
0, & \text{otherwise}
\end{cases}
\]

(8)

Similar to the 3-color 3D checkerboard pattern used in the Slice-Symmetric transformation the Volume-Symmetric transformation uses a 6-color 3D checkerboard pattern. Figure 16 illustrates the first 6 slices of such a checkerboard. The same pattern is repeated in the higher slices. The color of a square \((j, k)\) in slice \(i\) is determined by the value of the expression \(i + j + k \text{ mod 6}\). It can be observed that the color pattern shift by one color between each consecutive slice.

The Volume-Symmetric transformation is similar to the Slice-Symmetric transformation (Equation 6) except that instead of using only 3 of the 6 tetrahedrons that pack a cube, all the 6 tetrahedrons are used. In order to compute the same number of interactions as before, the spacing between the individual elements in the transformed force matrix is doubled since all the arithmetic is now modulo 6 based. Figure 17 illustrates the elements of the force matrix after a Volume-Symmetric transformation has been applied to it. The elements of the matrix are colored depending on which case of Equation 8 they satisfy (Note: this coloring definition is different from that used in Figure 19). A pattern of parallel diagonal slices can be observed depending on the position of the camera. There is a slight change in the separation between the slices at the boundaries of each of the tetrahedrons. Figures 18 and 19 illustrate slices of the volume-symmetric force matrix. An uniform spatial distribution of interactions can be observed among the slices.

6.1 Volume-Symmetric Distribution Properties

The volume-symmetric transformation can be used with the atom-decomposition as well as with the force-decomposition MD algorithms. This is because it uniformly distributes the number of interactions to be computed among the slices of the force matrix as well as between similar sub-volumes of the force matrix.

Theorem 6.1 In an \(N \times N \times N\) Volume-Symmetrically transformed 3-body force matrix, the number of interactions to be computed are uniformly distributed among the slices of the force matrix along any axis.

Proof The number of interactions computed in any slice \(m, 0 \leq m < N\) of an \(N \times N \times N\) 3-body force matrix is given by Table 6.1

The number of interactions per slice can be determined using a technique similar to Theorem 5.3. Since modulo 6 arithmetic is performed, the number of exact solutions is large compared to the Slice-Symmetric transformation. It can be observed that in the worst case, the load imbalance between the slices is 5 interactions. However, given the divisibility constraints, this is the optimal distribution for any value of \(N\). Similarly, it can be shown that the number of interactions to be computed are uniformly distributed among the slices along the other two dimensions of the force-matrix.

When the Volume-Symmetric transformation is used with the force-decomposition MD algorithm, the force matrix is decomposed along all three dimensions instead of just one dimension as in the case of the atom-decomposition algorithm. Thus, a sub-cube of the \(N \times N \times N\) force matrix of side \(\frac{N}{\sqrt{6}}\) is assigned to each of the \(P\) processors. Figure 20 illustrates the number of interactions assigned to each of the 64 processors used with the force-decomposition algorithm and a Volume-Symmetrically transformed \(1680 \times 1680 \times 1680\) force-matrix. It can be seen that there are three distinct bands for the load assignment. The smallest assignment corresponds to the processors along the diagonal. These processors are required to
Table 1. Number of interactions in each slice of the force matrix when using the Volume-Symmetric transformation. \( N' = \frac{(N-1)(N-2)}{6} \) and \( N'' = \frac{N(N-3)}{6} \).

\[
\begin{array}{|c|c|c|c|c|c|c|}
\hline
& m \equiv 0 \text{ mod } 6 & m \equiv 1 \text{ mod } 6 & m \equiv 2 \text{ mod } 6 & m \equiv 3 \text{ mod } 6 & m \equiv 4 \text{ mod } 6 & m \equiv 5 \text{ mod } 6 \\
\hline
N' \equiv 0 \text{ mod } 6 & N'' & N'' + 1 & N'' - 2 & N'' + 1 & N'' + 2 & N'' \\
N' \equiv 1 \text{ mod } 6 & N' & N' & N' - 2 & N' + 1 & N' + 1 & N' \\
N' \equiv 2 \text{ mod } 6 & N' & N' & N' - 1 & N' + 2 & N' & N' - 1 \\
N' \equiv 3 \text{ mod } 6 & N'' & N'' + 1 & N'' & N'' + 1 & N'' & N'' \\
N' \equiv 4 \text{ mod } 6 & N' & N' + 1 & N' - 1 & N' & N' & N' \\
N' \equiv 5 \text{ mod } 6 & N' & N' + 1 & N' - 2 & N' & N' + 1 & N' \\
\hline
\end{array}
\]

ensure that all three co-ordinates of the interactions are distinct i.e. \( i \neq j \neq k \). Hence, the number of interactions computed by the processors that are assigned sub-volumes of the force-matrix that are along the diagonal is given by

\[
N_{\text{min}} = \frac{N}{\sqrt[3]{P}} \left( \frac{N}{\sqrt[3]{P}} - 1 \right) \left( \frac{N}{\sqrt[3]{P}} - 2 \right)
\]  

Equation 9 does not take into consideration the divisibility constraints since, if we did, there would be too many combinations of expressions as already seen in the case of Table 6.1. However, the expressions are very close to the exact solution and differ from the actual solutions by at most 5 interactions depending on the divisibility constraints. This is negligible since in most cases, we are dealing with systems where the number of interactions is much larger. For example, in a system with 1000 atoms, the number of unique interactions is roughly \( 1.7 \times 10^9 \). In typical simulations the number of atoms used is significantly much larger than 1000.

**Theorem 6.2** If the Volume-Symmetric distribution is used with the force-decomposition MD algorithm to schedule an \( N \times N \times N \) 3-body force matrix onto a collection of \( P \) homogeneous processors, the RLI is given by

\[
RLI = \frac{\sqrt[3]{P} \left( 3N - 2\sqrt[3]{P} \right)}{N (N - 3) + 2}
\]

**Proof** When using the Volume-Symmetric transformation, the maximum and minimum number of interactions computed by
any processor are \(N\text{max}\) and \(N\text{min}\) respectively. The ALI can now be determined as

\[
\text{ALI} = \frac{N\text{max} - N\text{min}}{6} = \frac{N}{\sqrt[3]{P}} \left(3 \cdot \frac{N}{\sqrt[3]{P}} - 2\right)
\]

The RLI can be determined from the ALI as

\[
\text{RLI} = \frac{6 \cdot P \cdot \text{ALI}}{N(N-1)(N-2)} = \frac{\sqrt[3]{P} \left(3N - 2\sqrt[3]{P}\right)}{N(N-3) + 2}
\]

Figure 21 plots the RLI determined by Theorem 6.2. It can be observed that the RLI decreases with increasing number of atoms for a fixed number of processors. For a fixed number of atoms, the RLI increases with an increase in the number of processors. However, the rate of increase in the RLI decreases with an increase in number of processors. The second plot in Figure 21 plots the RLI by executing the MD simulation with the bodies of the loops removed and counting the number of times the loops are entered. Both plots are identical implying that ignoring the divisibility conditions has virtually no effect. For all the points in the plot, we found that the maximum difference between the closed form solution and the real solution was 2 interactions. While counting the loops, we selected \(P\) such that it was a perfect cube to ensure optimal load balance.

**Theorem 6.3** If the Volume-Symmetric distribution is used with the force-decomposition MD algorithm to schedule an \(N \times N \times N\) 3-body force matrix onto a collection of \(P\) homogeneous processors, the upper bound for efficiency (\(\eta'\)) due to load imbalance is given by

\[
\eta' = \frac{(N - 1)(N - 2)}{N^2}
\]

**Proof** Assuming that the time it takes to evaluate a single interaction on any of the processors is \(t_{\text{exec}}\), the sequential execution time can be determined as

\[
t_{\text{seq}} = \frac{N(N-1)(N-2)}{6} \cdot t_{\text{exec}}
\]

The parallel execution time is determined by the execution time on the processor that is assigned the maximum load since this forces the remaining processors to be idle toward the end of the time step. The maximum load assigned to any processor is \(N\text{max}\). The efficiency upper bound due load balance can now be determined as

\[
\eta' = \frac{N(N-1)(N-2)}{6PN\text{max}} = \frac{N(N-1)(N-2) \cdot 6P}{N^3} = \frac{(N - 1)(N - 2)}{6}
\]

**Theorem 6.3** shows that efficiency upper bound due to load imbalance is greater than 99.95% for most typical MD simulations and is independent of the number of processors, implying that the Volume-Symmetric transformation is highly scalable. The efficiency bound is plotted in Figure 22. It can be observed that the efficiency is close to perfect and it improves with an increase in the number of atoms \(N\). Hence, it can be inferred that the slight load imbalance when using the Volume-Symmetric transform is insignificant for typical 3-body MD simulations.

In the case of a heterogeneous environment, the Volume-Symmetric transformation can be used in conjunction with any block level scheduling algorithm such as [2], [4], [7] etc. since the interactions to be computed are uniformly distributed over the entire volume of the force matrix. This allows for arbitrary sub-volumes of the force matrix to be assigned to processors such that the volume of the assigned portions of the force matrix is proportional to the compute power of the processor.
7 Implementation

7.1 Slice-Symmetric Transformation

We implemented the atom-decomposition algorithm using both the cyclic distribution and symmetric distribution techniques using the MPICH implementation [5] of MPI. The variable rank in the pseudocode of this paper refers to the processor rank assigned by MPICH. Figure 10 illustrates the loops in our implementation of the cyclic distribution force evaluation routine. It is optimized when compared to loops over all \( N \) tuples and using an if condition to determine which one satisfies the cyclic distribution property.

Figure 11 illustrates a naïve implementation of our symmetric distribution. Due to the symmetry in the force matrix, the if statement in the inner-most loop is taken only \( \frac{1}{3} \) of the time i.e. it is not taken 83.3\% of the time. To avoid this, we use loop unrolling and jamming techniques to re-order the loops and eliminate the need for a conditional statement in the inner-most loop. It is not possible for the compiler to perform this optimization since this optimization involves modulo arithmetic to determine the bounds of the loops.

The optimized loops are shown in Figure 12. The loop bounds are selected such that they satisfy the inequalities in Equation 6. Consider the implementation of Condition 1 of Equation 6. The loop bounds of the \( i \) and \( j \) loops are obvious. The initial value of the \( k \) loop \( k_0 \) can be either \( j-1 \), \( j-2 \) or \( j-3 \) depending on the divisibility constraints. We consider the three cases:

1. \( k_0 = j-1 \): If \( k_0 \) is supposed to be \( j-1 \), it means that \( i+j+(j-1) \equiv 1 (mod 3) \). Hence, if \( i+j+j \equiv 2 (mod 3) \), \( k_0 = j-1 \).
2. \( k_0 = j-2 \): If \( k_0 \) is supposed to be \( j-2 \), it means that \( i+j+(j-2) \equiv 1 (mod 3) \). Hence, if \( i+j+j \equiv 0 (mod 3) \), \( k_0 = j-2 \).
3. \( k_0 = j-3 \): Similarly, if \( i+j+j \equiv 1 (mod 3) \), \( k_0 = j-2 \).

To combine the above three cases, we need to make the remainders in the derived expressions increase monotonically. This can be done by adding one to both sides of the equivalence equations. Hence, if \( i+j+j+1 \equiv 0 (mod 3) \) then \( k_0 = j-1 \), if \( i+j+j+1 \equiv 1 (mod 3) \) then \( k_0 = j-2 \) and if \( i+j+j+1 \equiv 2 (mod 3) \) then \( k_0 = j-3 \). Thus, \( k_0 = j-t \), where \( t = (i+j+j+1) mod 3 \). Similarly, the loop bounds for the other conditions in Figure 12 can be determined.

We have found that the optimized version of the Slice-Symmetric distribution performs on average 13.8\% faster than the unoptimized version.

7.2 Volume-Symmetric Transformation

The Volume-Symmetric transformation with the force-decomposition algorithm was implemented on a homogeneous cluster. The optimized loops of the Volume-Symmetric transformation with the force-decomposition algorithm for a processor assigned the sub-cube of the force matrix with corners \((N_iL, N_jL, N_kL), (N_iU, N_jU, N_kU)\) are illustrated in Figure 23. For case 1 of Equation 8, the bounds of the \( j \) loop can be determined as follows

\[
\begin{align*}
    j & \in [0, i) \land j \in [N_jL, N_jU) \\
    & \equiv j \in [\max(0, N_jL), \min(i, N_jU)] \\
    & \equiv j \in [N_jL, \min(i, N_jU)] 
\end{align*}
\]

Similarly, it can be shown that \( k \in [N_kL, \min(i, N_kU)] \). However, when taking into consideration the divisibility constraints, the initial value of the innermost loop \( k_0 \) can be either \( N_kL, N_kL+1, N_kL+2, N_kL+3, N_kL+4 \) or \( N_kL+5 \). Similar to the Slice-Symmetric transformation, it can be shown that \( k_0 = N_kL+t \), where \( t = 5 - [(i+j+N_kL+5) mod 6] \). Similarly, the loop bounds of the other cases of Equation 8 can be determined.

The Volume-Symmetric transformation is more computationally intensive than the Slice-Symmetric transformation due to the additional cases to be tested. Hence, it is preferable to use the Slice-Symmetric transformation instead of the Volume-Symmetric transformation with the atom-decomposition MD algorithm although the Volume-Symmetric transformation also uniformly distributes the number of interactions to be evaluated among the slices of the force matrix. However, in the case
of the force decomposition MD algorithm, the Volume-Symmetric transformation is the only way to take advantage of the symmetries in the force matrix.

The optimized loops of Figure 23 perform roughly 2.7% slower than the Slice-Symmetric transformation due to the increased number of cases to be evaluated. In Section 8, we discuss situations where the Volume-Symmetric transformation might be useful.

8 Results and Discussion

To evaluate the load balance properties of the transformations, we use the Step Time Variation Ratio (STVR) [8]. The STVR for processor \( i, 0 \leq i < P \) is given by

\[
\text{STVR}_i = \left| \frac{T_i - \frac{1}{P} \sum_{j=0}^{P-1} T_j}{\frac{1}{P} \sum_{j=0}^{P-1} T_j} \right|
\]

where, \( P \) processors are used to evaluate an \( N \times N \times N \) force matrix and \( T_i \) is the average time to perform one time-step of the MD simulation on processor \( i, 0 \leq i < P \). The average step time \( T_i \) corresponds to the execution time of the parallelized force routine on each of the processors. It ignores the sequential execution time since the \( O(N^3) \) computational intensity of the force evaluation is much greater than the linear time sequential portions of the code. The \( T_i \) values do not consider the communication time. The obtained STVR is thus an index of the imbalance in the force transformation algorithm.

To evaluate the performance of the Cyclic and the Slice-Symmetric distributions in a homogeneous cluster environment, the following tests were performed on an Opteron 275 cluster with a Myrinet backplane. Figure 13 plots the STVR for the cyclic and symmetric distributions. The STVR in the case of the cyclic distribution is in the shape of a ‘V’. The ‘V’ shape is more pronounced as the number of processors used increases. This is due to the load slightly decreasing with increasing processor rank as proved in Lemma 4.1 and illustrated in figure 2. It can also be seen that the STVR for the Slice-Symmetric distribution is lower than that of the cyclic distribution since it optimally distributes the load among the processors.

The typical execution time for the 3-body force calculation is so much greater than the time required for the communication at the end of the time-step that we observed efficiencies greater that 99% for both the distributions. For a system with 8000 atoms running on 64 processors, the computation time was about 270 seconds. The size of the force vector to be communicated at the end of the time step is only about 187 kilobytes when a system with 8000 atoms is simulated. It takes only about 20 milliseconds to perform the required communication tasks at the end of the time-step on a Myrinet backplane. For larger number of atoms, this difference between the computation and communication times reduces further since the communication time scales as \( O(N) \) while the computation time scales as \( O(N^3) \).

Since the cyclic distribution works only in a homogeneous environment, we evaluate the STVR in a heterogeneous environment only for the Slice-Symmetric distribution using the atom-decomposition MD algorithm. For this test, we used a 4 node cluster comprising of two Athlon XP 2000+’s, an Athlon XP 2400+ and a Pentium III 800 MHz. We used the initial force calculation step as a benchmark for all the available processors and used the initial step execution time of each processor \( t_i, 0 \leq i < P \) to determine the number of slices to be assigned to processor \( i \) for the future time-steps as

\[
N_i = \frac{N}{t_i \sum_{j=0}^{P-1} \frac{1}{t_j}}
\]

where, \( N \) is the number of atoms in the simulation. Figure 14 illustrates that the number of slices assigned to each of the processors is proportional to the CPU performance. It can be seen from Figure 15 that the STVR is below 0.5% even in the case of a heterogeneous environment.

However, it must be noted that our Slice-Symmetric distribution is more complex in terms of computational intensity than the cyclic distribution. We have found that when using the optimized versions of both the distributions described in Section 7 the execution time of the cyclic distribution is 5.1% lower than that of the Slice-Symmetric distribution.

The bounds that we derived in sections 4.1 and 5.2 can help in determining which distribution to use for a given number of atoms \( N \) and number of processors \( P \). If the efficiency bound for the cyclic distribution is less than 95%, the Slice-Symmetric distribution is a better choice since at this point, the overhead of the Slice-Symmetric distribution is compensated for by its better load balancing properties. However, in most typical scenarios, the cyclic distribution is preferable in a homogeneous environment due to the disadvantage of poorer load balance being compensated for by the reduced computational overhead. In a heterogeneous environment, the symmetric distributions are the only option. Since the Slice-Symmetric distribution...
distributes the number of interactions evenly among the slices of the transformed force matrix, it can be used in conjunction with any slice-level scheduling algorithm. It can also be used in a distributed environment using scheduling techniques that we developed in [14].

The STVR of the Volume-Symmetric transformation with the force-decomposition MD algorithm in a homogeneous cluster is shown in Figure 24. It can be observed that the STVR is worse than the STVR of the Slice-Symmetric transformation. This is due to the slight load imbalance inherent in the Volume-Symmetric transformation (Section 6.1). Hence, in most typical scenarios, the Slice-Symmetric transformation is the optimal choice. The Volume-Symmetric transformation can be useful in massively parallel systems where the abundance of processors causes the communication time become significant. It can also be used in distributed systems with very low bandwidth links between the processors since the force-decomposition algorithm requires less communication than the atom-decomposition technique [9].

9 Conclusion

In this paper, symmetric transformations for a 3-dimensional force matrix are described. A Slice-Symmetrically transformed force matrix has the property that the total number of unique interactions in the system is optimally distributed among the slices of the 3-body force matrix along any dimension. This allows any heterogeneous slice level scheduling algorithm to be used on the transformed force matrix. A Volume-Symmetrically transformed force matrix has the property that the interactions to be computed are uniformly distributed over the entire volume of the force matrix. This allows any heterogeneous block level scheduling algorithm to be used on the transformed force matrix. Theoretical upper bounds for efficiency and relative load imbalance when using these distributions have been established. A detailed mathematical analysis of a priori technique in the literature known as the cyclic distribution has also been performed. We have also proved some interesting properties about the these distributions. The cyclic distribution can be used only in a homogeneous environment with the atom-decomposition MD algorithm. The Slice-Symmetric distribution can be used in a homogeneous and heterogeneous environment with the atom-decomposition MD algorithm. The Volume-Symmetric transformation can be used in a homogeneous and heterogeneous environment with the atom- and force-decomposition MD algorithms. An efficient loop reordering optimization for the force calculation routine in a 3-body potential when using the symmetric transformations has been described.

We have also evaluated the performance of an MPI implementation of the transformations. When using the cyclic distribution, the execution time is on average 5.1% lower than that of the Slice-Symmetric distribution. When using the Volume-Symmetric transformation, the overhead of the increased complexity is a 2.7% increase in execution time when compared to the Slice-Symmetric transformation. In the case of a homogeneous system, the bounds described in this paper can be used to determine which transformation technique is optimal for a given number of processors and number of atoms. For a heterogeneous system, the Symmetric transformations are the only option. The Slice-Symmetric transformation is to be used with the atom-decomposition MD algorithm and the Volume-Symmetric transformation is to be used with the force-decomposition MD algorithm. Nonetheless, only the symmetric distributions provides a viable option for the processing power found in heterogeneous systems such as the computational grid.

10 Acknowledgement

This work was completed utilizing the Research Computing Facility with the associated USCMS Tier-2 site at the University of Nebraska-Lincoln.

References

11 Figures

Figure 1. Transformed force matrix when using the cyclic distribution. The dark triangular surfaces represent the number of interactions in that slice.
Figure 2. Assignment of interactions among 64 processors when using the cyclic distribution technique. The dark points represent the load assigned to each of the 64 processors. The horizontal line represents the optimal load assignment.

Figure 3. Relative Load Imbalance when using the cyclic distribution.
Figure 4. Efficiency Upper Bound due to load imbalance when using the cyclic distribution.

Figure 5. The first three slices of a $60 \times 60 \times 60$ 3-color 3D checkerboard. This sequence repeats for the remaining slices. The pattern shifts by one color for each consecutive slice.

Figure 6. Three slices of the force matrix when using a Slice-Symmetric transformation.
Figure 7. The first two figures illustrate how a tetrahedron can be sliced such that the cut surface is either a triangle or a square. The next two figures illustrate the cutting of a tetrahedron along all three axes.
Figure 8. Various views of the transformed force matrix when using the Slice-Symmetric distribution.
Figure 9. Two slices of the transformed force matrix when using the symmetric distribution. The two cut planes depict the interactions that are computed in the corresponding slices. Only the visible parts of the plane are computed. The first figure cuts along the $jk$ plane. The second figure cuts along the $ij$ plane. The elements of the matrix have been made translucent to enable better visibility of the cut slices. The co-ordinates of the force matrix are normalized.

```c
for (i=rank; i<N; i+=size) {
    for (j=i+1; j<N; j++) {
        // compute $r_{ij}$
        for (k=j+1; k<N; k++) {
            compute $r_{ik}$, $r_{jk}$
            compute potential
            compute forces
        }
    }
}
```

Figure 10. Optimized force evaluation loops for the cyclic distribution.
for (i=rank*N/P; i<(rank+1)*N/P; i++) {
    for (j=0; j<N; j++) {
        // compute r_ij
        for (k=0; k<N; k++) {
            t = (i + j + k) % 3;
            if ((i>j && j>k && t==1) ||
                (j>k && k>i && t==2) ||
                (k>i && i>j && t==0)) {
                compute r_ik, r_jk
                compute potential
                compute forces
            }
        }
    }
}

Figure 11. Unoptimized force evaluation loops for the Slice-Symmetric transformation.
for (i=rank*N/P; i<(rank+1)*N/P; i++) {
    for (j=i-1; j >= 0; j--) {
        // compute r_ij
        t = 1 + ( (i+j+j+1) % 3 );
        for (k=j-t; k>=0; k-=3) {
            // Condition 1
            compute r_ik, r_jk
            compute potential
            compute forces
        }
    }

    for (k=i+1; k<N; k++) {
        // compute r_ik
        t = 3 - ( (i+k+k+1) % 3 );
        for (j=k+t; j<N; j+=3) {
            // Condition 2
            compute r_ij, r_jk
            compute potential
            compute forces
        }

        t = 1 + ( (i+i+k+5) % 3 );
        for (j=i-t; j>=0; j-=3) {
            // Condition 3
            compute r_ij, r_jk
            compute potential
            compute forces
        }
    }
}

---

**Figure 12.** Optimized force evaluation loops for the Slice-Symmetric transformation.
Figure 13. Comparison of STVR (in percentage) for Cyclic Distribution and Slice-Symmetric Distribution in a homogeneous cluster environment. A system with 8000 atoms was used for this experiment.

Figure 14. Number of slices assigned to each processor in a heterogeneous cluster using the Slice-Symmetric distribution. The horizontal line in the plot corresponds to an equal $N/P$ distribution. Processor P1 and P2 are Athlon XP 2000+’s, P2 is an Athlon XP 2400+ and P4 is a Pentium III 800MHz.
Figure 15. STVR for the heterogeneous cluster used in figure 14 using the Slice-Symmetric distribution technique.

Figure 16. The first six slices of a 6-color $60 \times 60 \times 60$ 3D checkerboard. This sequence repeats for the remaining slices. The pattern shifts by one color for each consecutive slice. Notice that the spacing between the diagonal lines is twice of that in Figure 6.
Figure 17. Various views of the transformed force matrix when using the Volume-Symmetric distribution for 50 atoms. Notice that the spacing between the parallel diagonal slices is twice of that in Figure 8.

Figure 18. A Slice of the Volume-Symmetric transformed Force Matrix in Figure 17.
Figure 19. Three slices of a $60 \times 60 \times 60$ force matrix when using the Volume-Symmetric transformation.

Figure 20. Load assignment when using the Volume-Symmetric transformation along with a force decomposition MD algorithm for a system of 1680 atoms with 64 homogeneous processors. The horizontal line in the plot corresponds to the optimal load i.e. $\frac{N(N-1)(N-2)}{6P}$. 
Figure 21. Relative Load Imbalance when using Volume-Symmetric transformation with the Force Decomposition MD algorithm. The first figure uses the closed form expressions derived in Theorem 6.2. The second figure was generated by running an MD simulation with empty loops and counting the number of times they were executed.

Figure 22. Efficiency upper bound due to load imbalance when using the Volume-Symmetric transformation with the force-decomposition algorithm in a homogeneous environment.
for (i=NiL; i<NiU; i++) {
  // i>j
  for (j=NjL; j<MIN(i, NjU); j++) {
    t=5-(i+j+NkL+5) % 6;
    for (k=NkL+t; k<MIN(j, NkU); k+=6) {
      // Condition 1
    }
    t=1+(i+j+NkU+1) % 6;
    for (k=NkU-t; k>=MAX(i+1, NkL); k-=6) {
      // Condition 5
    }
  }
  // i<k
  for (k=NkL; k<MIN(i, NkU); k++) {
    t=5-(i+k+NjL+4) % 6;
    for (j=NjL+t; j<MIN(k, NjU); j+=6) {
      // Condition 2
    }
    t=1+(i+k+NjU+2) % 6;
    for (j=NjU-t; j>=MAX(i+1, NjL); j-=6) {
      // Condition 4
    }
  }
  // k>i
  for (k=MAX(NkL, i+1); k<NkU; k++) {
    t=1+(i+k+NjU+3) % 6;
    for (j=NjU-t; j>=MAX(NjL, k+1); j-=6) {
      // Condition 3
    }
  }
  // j>i
  for (j=MAX(NjL, i+1); j<NjU; j++) {
    t=1+(i+j+NkU) % 6;
    for (k=NkU-t; k>=MAX(NkL, j+1); k-=6) {
      // Condition 6
    }
  }
}

Figure 23. Optimized force evaluation loops for the Volume-Symmetric transformation. Each processor is assigned the sub-volume defined by the cube with corners (NiL, NjL, NkL) and (NiU, NjU, NkU).
Figure 24. STVR for a system with 5832 atoms using the Volume-Symmetric transformation with the force-decomposition algorithm on 27 homogeneous processors.