A Research on Fast Multipole Method for Molecular Simulation

Jiamian Huang
School of Computing
Tokyo Institute of Technology
Tokyo, Japan
jiamian@rio.gsic.titech.ac.jp

***Abstract*—Predicting the motions of a group of objects interacting with each other is called n-body problem. In an n-body system with N objects, the number of interactive pairs becomes N^2, thus a direct method leads to $O(N^2)$ complexity. In a system with a large N like molecular dynamic application in which millions of particles are simulated, the direct method is regarded impractical. The Fast Multipole Method deals with this issue by converting the object-object interactions into cell-cell interactions, which results in an $O(N)$ complexity. However, the FMM also faces problems including slow execution and insufficiency of accuracy compared to traditional methods like PME. This paper intends to give a brief review on the Fast Multipole Method, and introduce some methods to deal with these drawbacks.**

***Index Terms*—N-body problem, Fast multipole method, Molecular dynamics, High performance computing, Parallel algorithm**

I. Introduction

The simulation of multiple objects interacting with each other is a critical task in many field, e.g. biomolecular simulation, astronomy and fluid simulation. In molecular dynamic(MD) simulation field, the *Coulomb force* and the *van der Waals force* contribute the most part of non-bonded intermolecular interaction. When a molecule is absorbed or rejected by the intermolecular forces from the other molecules, its motion is changed according to Newton's law of motion. Therefore, we can predict its motion using time integration. Since the van der Waals force decays fast in the far field in which a cutoff can be applied, the most significant part becomes the Coulomb force. Coulomb force arises between two charged atoms and proportional to charges magnitude according to Coulomb's law. The Coulomb force appears between every atom pair, making it the most computational expensive part in a molecular simulation application.

In a system of N atoms, the number of interaction pairs is N^2. When compute each pair in a direct way, the computation complexity becomes $O(N^2)$. For a real application in which millions of atoms are simulated this method is impractical since the computation time increases quadratically along with the object number. Many researches are conducted to achieve a fast evaluation of Coulomb force.

The Particle Mesh Ewald(PME) method [Ess+95] is a prevalent method for molecular simulation. PME splits the interaction list into short-range and long-range. In short-range, a direct method is applied. In long-range, charges are interpolated into a uniformed grid and the Fast Fourier Transform(FFT) is applied on it in reciprocal space. This method achieves an $O(NlogN)$ complexity and is well-implemented in many molecular simulation software like GROMACS [Abr+23]. However, PME faces with bottleneck for parallelization when large number of particles are simulated, due to the all-to-all communication requirement of the underlying FFT algorithm [KKG20]. This drawback limits the applicability of PME to small and medium simulation.

The Fast Multipole Method(FMM) [GR87] was proposed to accelerate the evaluation in an alternative way. In FMM, particles are split into near-field and far-field. In near-field the direct evaluation is performed. In far-field the particle-particle interaction is converted into cell-cell interaction using multipole expansion. Since the FMM does not face the limitation as PME and achieves a even lower $O(N)$ complexity, it is more suitable for higher parallelized application simulating large number of particles. Due to its efficiency, many practical parallel implementation are proposed. On the other hand, FMM faces force discontinuity problem when particles go through the cell border. This leads to the violation of energy conservation, thus a energy drift which is crucial for a MD simulation [EP16; CDF10] in a long time period. To deal with it, regularization methods were proposed for target [CDF10] and both [Sha+19]. The regularization alleviates the energy drift but therefore increases the computation cost. This research aims at accelerating the additional computing in regularization using parallelization methods.

II. Fast Multipole Method

Given a system of N charged particles located at position $\{x_1, x_2, \ldots, x_n\}$ with charges $\{q_1, q_2, \ldots, q_n\}$, the Coulomb potential for an arbitrary position y becomes

$$\phi(y) = -\sum_{i=1}^{N} q_i \log |x_i - y| \tag{1}$$

in two dimension and

$$\phi(y) = \sum_{i=1}^{N} \frac{q_i}{|x_i - y|} \tag{2}$$

in three dimension, where $|x_i - y|$ indicates the Euclidean distance between x_i and y. Generally, x_i is called source

and y is called target. In the case a particle with charge q locates at y, the Coulomb force receiving is represented as the multiplication of charge and gradient of potential

$$f(y) = -q\nabla\phi(y). \tag{3}$$

For each particle in the system this computation will be repeated, thus leading to an $O(N^2)$ complexity. For simplicity, we use complex **x** and **y** to deal with the problem. Therefore, the potential function becomes

$$\phi(\boldsymbol{y}) = -\sum_{i=1}^{N} q_i \log|\boldsymbol{x_i} - \boldsymbol{y}| \tag{4}$$

in two dimension and

$$\phi(\boldsymbol{y}) = \sum_{i=1}^{N} \frac{q_i}{|\boldsymbol{x_i} - \boldsymbol{y}|} \tag{5}$$

in three dimension, thus $\phi(y) = \Re(\phi(\boldsymbol{y}))$.

A. Multipole Expansion and Local Expansion

The interaction between target and source can be described in a more generalized way using kernel function G

$$\phi(\boldsymbol{y}) = \sum_{i=1}^{N} q_i G(\boldsymbol{x_i} - \boldsymbol{y}). \tag{6}$$

If we can find an expansion for $G(\boldsymbol{x_i} - \boldsymbol{y})$ that

$$G(\boldsymbol{x_i} - \boldsymbol{y}) = \sum_{k=0}^{\infty} T_k(\boldsymbol{x_i}) H_k(\boldsymbol{y}), \tag{7}$$

then we can separate source and target into two separated parts A and B

$$\phi(\boldsymbol{y}) = \underbrace{\sum_{k=0}^{P-1} H_k(\boldsymbol{y}) \underbrace{\sum_{i=1}^{N} q_i T_k(\boldsymbol{x_i})}_{\text{A}}}_{\text{B}} \tag{8}$$

as we should truncate after the former P items in a real implementation. Equation 8 converts the particle-particle interaction into particle-cell interaction as we can regard the A part as a cell, and B part is an interaction to the cell.

Furthermore, we can convert the particle-cell interaction into cell-cell interaction. Notice that

$$\boldsymbol{x} - \boldsymbol{y} = (\boldsymbol{x} - \boldsymbol{x'}) + (\boldsymbol{x'} - \boldsymbol{y'}) + (\boldsymbol{y'} - \boldsymbol{y}). \tag{9}$$

In the two dimension case, the potential function becomes

$$\phi(y) = \sum_{k=0}^{P-1} \frac{1}{k!} (\boldsymbol{y} - \boldsymbol{y'})^k L_k^{'} \tag{10}$$

where

$$L_k^{'} = \sum_{n=0}^{P-k-1} \nabla^{(n+k)} G(\boldsymbol{y'} - \boldsymbol{x'}) M_n^{'} \tag{11}$$

and

$$M_n^{'} = \sum_{i=1}^{N} \frac{q_i}{n!} (\boldsymbol{x'} - \boldsymbol{x})^n \tag{12}$$

Equation 12 is called the **particle to multipole expansion(P2M)**, Equation 11 is called the **multipole Expansion to local expansion(M2L)** and Equation 10 is called the **local expansion to particle(L2P)**. These equations are derived out using Taylor expansion and binary expansion. In the three dimension, spherical harmonics is a more proper expansion method [GR88; CGR99]. Figure 1 shows such cell-cell interaction. As shown in Figure 1, for different y in the same cell,

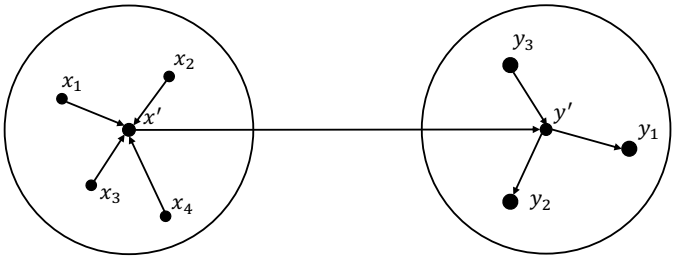

Fig. 1. The cell-cell interaction

only we should recompute is the **L2P** operator.

B. Translation Operators

In FMM cells are constructed hierarchically to reduce redundant computing. Therefore, we need to translate the expansions from one center to another, as shown in Figure 2.

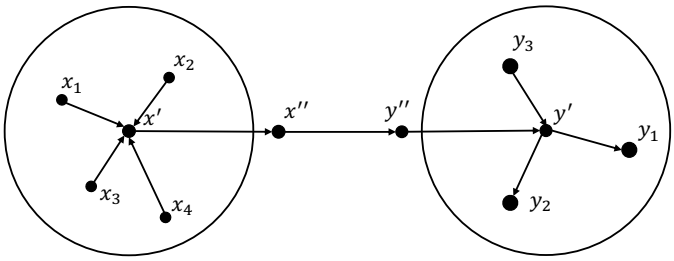

Fig. 2. Translation of multipole and local expansion

Notice that

$$\boldsymbol{x''} - \boldsymbol{x} = (\boldsymbol{x''} - \boldsymbol{x'}) + (\boldsymbol{x'} - \boldsymbol{x}), \tag{13}$$

therefore, for multipole expansion $M_n^{''}$ centered at x'', the **multipole to multipole translation(M2M)** operator holds that

$$M_n^{''} = \sum_{m=0}^{n} \frac{(\boldsymbol{x''} - \boldsymbol{x'})^{n-m}}{(n-m)!} M_m^{'}. \tag{14}$$

Similarly, local expansion $L_l^{'}$ centered at y' can be represented using local expansion $L_k^{''}$ centered at y'' using **local to local translation(L2L)** operator

$$L_l^{'} = \sum_{k=l}^{P-1} \frac{(\boldsymbol{y'} - \boldsymbol{y''})^{k-l}}{(k-l)!} L_k^{''}. \tag{15}$$

Figure 3 shows how these operators work collaboratively to break down the direct calculation into five parts.

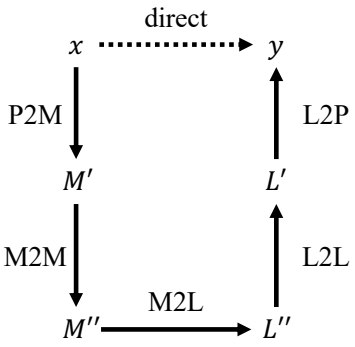

Fig. 3. Operators of FMM

C. The FMM Algorithm

Based on these five operators we show the Fast Multipole Method. Notice that only the P2M operator concerns is the difference of centers. Therefore, we can gather these small M' together to form a larger M'' using the M2M operator. Similarly, we can distribute the local expansion from a large L'' to several small L' using the L2L operator then add to each particle using L2P. On the other hand, since the M2L operator meets addition law for different M'', we can gather multipole expansion from different cells to form a single L''. This leads to a hierarchical cell structure.

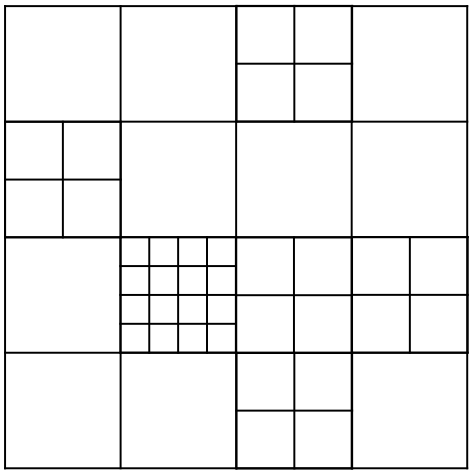

Fig. 4. Hierarchical cell structure

To form this tree-like structure, we divide the simulation box uniformly into four cells, then recursively divide each cells into four children cells until each cell hold a constant number of particle. Cells that do not have child cells are called leaf cells. Figure 4 shows the hierarchical structure. We define that two non-overlapping cells are well-separated that regard each other as far-field, otherwise near-field. The Fast Multipole Method algorithm consists of three steps.

- **Upward Pass :** For each leaf cell calculate the multipole expansion using the P2M operator. For the non-leave cells, gather multipole expansions from the corresponding children cells recursively using the M2M operator.
- **Horizontal Pass :** For each cell in the coarsest level, traverse all the cells in the same level. When meet well-separated cells, calculate and gather local expansion with all well-separated cells using M2L. When meet near-field cells, go to the finer level to perform M2L recursively. When both near-field cells have no child, perform direct calculation(**P2P**) for the internal particles.
- **Downward Pass :** For each cell in the coarsest level, distribute the local expansion to its children cell using L2L operator. When the cell has children then go to the finer level to perform the downward pass recursively, otherwise convert the local expansion into potential or/and force using L2P operator.

Since every operator holds a $O(kN)$ complexity, the overall complexity is $O(N)$.

III. FMM WITH PBC

In a typical MD software periodic boundary condition(PBC) is employed to avoid surface artifacts. FMM can be extended to PBC by duplicating the simulation box and placing around the original box recursively [LDBJ96]. These duplicate box are called images. As mentioned before, multipole expansion only concerns the difference between particle and center. Therefore, the multipole of cells in images are completely same as the cells in original box. Figure 5 shows the multi-level images structure for constructing a PBC. The l_0 indicates the original

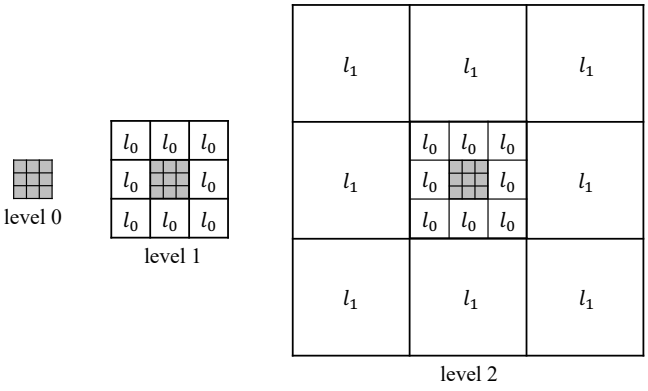

Fig. 5. FMM with PBC

simulation box. In level 1, l_0 box is copied 8 times and placed around the original box. Since some cells in the images are not well-separated to the original box, M2L or P2P should be performed. In level 2, l_1 box is copied 8 times and placed around the l_1 box. Since all images are well-separated from this level, only M2L will be performed. This procedure will be repeated until the expected depth.

IV. REGULARIZED FMM

When a target particle passes through the cell border the center it receives force from changed, resulting a surging in force. Similarly, a source particle passing through the cell

border will suddenly change the force the target receives. Figure 6 shows the discontinuity when cell passed cell border. These discontinuities in force violates the conservation of

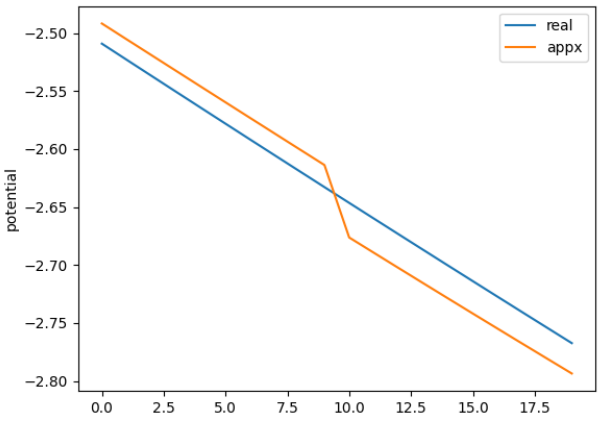

Fig. 6. Discontinuity in potential

energy, leading to an unphysical drift of energy in a long time period. To deal with it, a regularization approach can be applied to interpolate between cells [Sha+19]. Regularization is achieved by extending every geometry cell a certain size to a virtual cell. As shown Figure 7, position x is shared by four virtual cells. When calculating P2M for each virtual cell, particle in x will be included. On the other hand, when performing L2P potential in x will be interpolated from four virtual cells using

$$\phi(x) = w_1\phi_1(x) + w_2\phi_2(x) + w_3\phi_3(x) + w_4\phi_4(x), \quad (16)$$

where

$$w_1 + w_2 + w_3 + w_4 = 1. \quad (17)$$

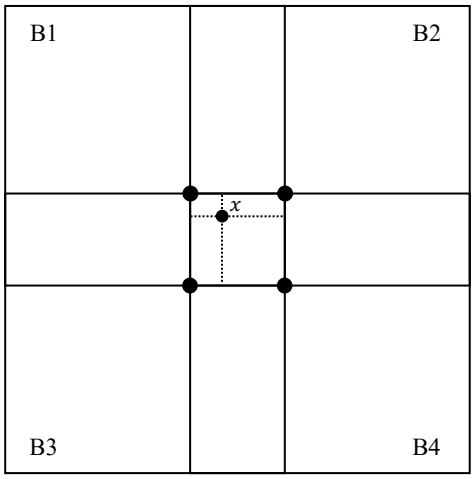

Fig. 7. Virtual cells

By employing this method, potential become smooth when particles pass cell border, as shown in Figure 8.

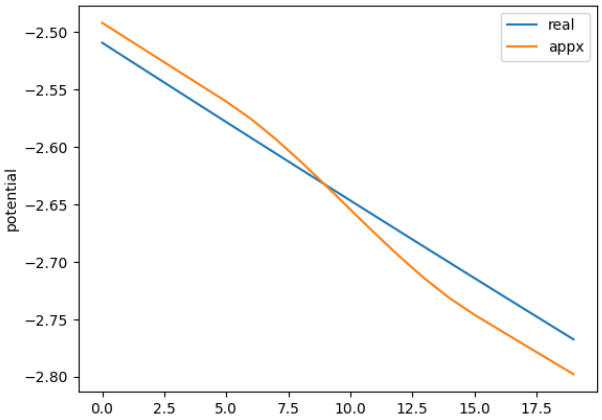

Fig. 8. Smooth change of potential by regularized FMM

V. Parallel Implementation of FMM

Although FMM decrease the complexity to $O(N)$, it also runs slowly when N is large enough. In a typical biomolecular simulation application N will be up to 10^9, by which FMM can be slow enough using single core. GPU is a powerful tool to make full use of the computational hardware. Many researches on the GPU parallelization of FMM have been conducted. Yokota et al. proposed a practical single GPU implementation of FMM called GemsFMM [YB11], which runs 150 times faster than the parallel direct method proposed by GPU Gem 3 [Fer+04] for $N = 10^7$. Kohnke et al. proposed an improved implementation that runs 8 times faster than the previous one for N from 10^4 to 10^7, and guarantees a same level of energy drift compared to the PME method. Moreover, parallel implementation using multiple GPUs are also proposed [Yok+09; Las+09; BY11].

VI. Conclusion

In this paper we reviewed on the Fast Multipole Method and its usage in molecular simulation. Although FMM runs faster than the widely-used PME for large N, the drift of energy and relatively lower accuracy limits its use. To deal with it, regularization method was proposed, and successfully alleviated the drift of energy. To further accelerate the regularized FMM, GPU-based parallelization method is considered to be a proper method given the successes in the previous GPU implementations of FMM.

References

[GR87] Leslie Greengard and Vladimir Rokhlin. “A fast algorithm for particle simulations”. In: *Journal of computational physics* 73.2 (1987), pp. 325–348.

[GR88] Leslie Greengard and Vladimir Rokhlin. “The rapid evaluation of potential fields in three dimensions”. In: *Vortex methods* 1360 (1988), pp. 121–141.

[Ess+95] Ulrich Essmann et al. “A smooth particle mesh Ewald method”. In: *The Journal of chemical physics* 103.19 (1995), pp. 8577–8593.

[LDBJ96] Christophe G Lambert, Thomas A Darden, and John A Board Jr. “A multipole-based algorithm for efficient calculation of forces and potentials in macroscopic periodic assemblies of particles”. In: *Journal of Computational Physics* 126.2 (1996), pp. 274–285.

[CGR99] Hongwei Cheng, Leslie Greengard, and Vladimir Rokhlin. “A fast adaptive multipole algorithm in three dimensions”. In: *Journal of computational physics* 155.2 (1999), pp. 468–498.

[Fer+04] Randima Fernando et al. *GPU gems: programming techniques, tips, and tricks for real-time graphics*. Vol. 590. Addison-Wesley Reading, 2004.

[Las+09] Ilya Lashuk et al. “A massively parallel adaptive fast-multipole method on heterogeneous architectures”. In: *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*. 2009, pp. 1–12.

[Yok+09] Rio Yokota et al. “Fast multipole methods on a cluster of GPUs for the meshless simulation of turbulence”. In: *Computer Physics Communications* 180.11 (2009), pp. 2066–2078.

[CDF10] Philippe Chartier, Eric Darrigrand, and Erwan Faou. “A regular fast multipole method for geometric numerical integrations of Hamiltonian systems”. In: *BIT Numerical Mathematics* 50.1 (2010), pp. 23–40.

[BY11] Lorena A Barba and Rio Yokota. “ExaFMM: An open source library for Fast Multipole Methods aimed towards Exascale systems”. In: *Boston: Boston University. Retrieved from barbagroup: http://barbagroup. bu. edu* (2011).

[YB11] Rio Yokota and Lorena A Barba. “Treecode and fast multipole method for N-body simulation with CUDA”. In: *GPU Computing Gems Emerald Edition*. Elsevier, 2011, pp. 113–132.

[EP16] Peter Eastman and Vijay S Pande. “Energy conservation as a measure of simulation accuracy”. In: *bioRxiv* (2016), p. 083055.

[Sha+19] DS Shamshirgar et al. “Regularizing the fast multipole method for use in molecular simulation”. In: *The Journal of Chemical Physics* 151.23 (2019), p. 234113.

[KKG20] Bartosz Kohnke, Carsten Kutzner, and Helmut Grubmuller. “A GPU-accelerated fast multipole method for GROMACS: performance and accuracy”. In: *Journal of Chemical Theory and Computation* 16.11 (2020), pp. 6938–6949.

[Abr+23] Mark Abraham et al. “GROMACS 2023.1 Manual”. In: (2023). DOI: 10 . 5281 / ZENODO . 7588710. URL: https : / / zenodo . org / record / 7588710.