Stony Brook University

High Scalable Implementation of SPME using Parallel Spherical Cutoff 3D FFT on the 6D Torus QCDOC Supercomputer

A Dissertation Presented

By

Bin Fang

to

The Graduate School

in Partial Fulfillment of the

Requirements

for the Degree of

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

May 2007

Stony Brook University

The Graduate School

Bin Fang

We, the dissertation committee for the above candidate for the

Doctor of Philosophy degree,hereby recommend
acceptance of this dissertation.

Prof. Yuefan Deng – Dissertation Advisor
Department of Applied Mathematics and Statistics

Prof. John Reinitz – Chairperson of Defense
Department of Applied Mathematics and Statistics

Prof. David F. Green – Committee Member
Department of Applied Mathematics and Statistics

Dr. James W. Davenport – Outside Member
Computational Science Center
Brookhaven National Laboratory

This dissertation is accepted by the Graduate School

Lawrence Martin
Dean of the Graduate School

Abstract of the Dissertation

High Scalable Implementation of SPME using Parallel Spherical Cutoff 3D FFT on the 6D Torus QCDOC Supercomputer

by

Bin Fang

Doctor of Philosophy

in

Applied Mathematics and Statistics

Stony Brook University

2007

In order to model complex heterogeneous biophysical systems with non-trivial charge distributions such as globular proteins in water, it is important to evaluate the long range forces present in these systems accurately and efficiently. The Smooth Particle Mesh Ewald summation technique (SPME) is commonly employed to determine the long range part of electrostatic energy in large scale molecular simulations. While the SPME technique does not give rise to a performance bottleneck in a single processor or scalar computation, current implementations of SPME on massively parallel supercomputers become problematic at large processor numbers, limiting the time and length scales that can be reached. Here, two accomplishments have been made in this dissertation to give rise to both improved accuracy and efficiency on massively parallel computing platforms. First of all, a well designed parallel framework of 3D

complex-to-complex FFT and 3D real-to-complex FFT for the novel QCDOC supercomputer with its 6D-torus architecture is given. The efficiency of this framework was tested on up to 4096 processors. Secondly, a new modification of the SPME technique is exploited, which was inspired by the non-linear growth of the approximation error of Euler Exponential Spline interpolation function. This fine grained parallel implementation of SPME has been embedded into MDoC package. Numerical tests of package performance on up to 1024-processor QCDOC supercomputer residing at Brookhaven National Lab are presented for two systems of interest, β-hairpin solvated in explicit water, a system which consists of 1112 water molecules and a 20 residue protein for a total of 3579 atoms, and HIV-1 protease solvated in explicit water, a system which consists of 8793 water molecules and a 198 residue protein for a total of 29508 atoms.

Contents

List of Figures

List of Tables

Acknowledgments

First of all, I would like to thank my advisor, Professor Yuefan Deng, for his deep insights and ideas, lots of invaluable advices, many helpful discussions, endless encouragement and continuous support in my Ph. D. program. Thank you so much for introducing me to this great research area.

I thank my Ph. D. committee members, Professor James Glimm, Professor Brent Lindquist and Dr. James W. Davenport, for their time and valuable comments.

I also like to give my special thanks to Dr. Glenn J. Martyna, without whom my research would never be so smooth.

I will never forget all members in the molecular dynamics research group, especially, Peter Rissland, Guowen Han and Xin Chen, who ever provided me their kind help when I needed. I have enjoyed very much working with all of you.

I also thank Chulwoo Jung, Robert Bennett and David Stampf for their kind help with the access to BNL QCDOC supercomputer. All my numerical results were obtained on QCDOC.

Finally, I would like to express my deepest love and thanks to my wife, Ping Hao, for her supports and sacrifices, and my mom who has been keeping encouraging me every moment and everywhere.

This project is supported by the BNL LDRD grant #36930 entitled "Molecular dynamics on ultrascalable supercomputer" and IBM.

1 Introduction

It has recently become possible to determine the genome of an entire species in a reasonable time frame [77, 52]. Using this information, it is turn possible to determine the primary sequence of the proteome of an organism. It then remains by synergistic theoretical and experimental research to determine the structure and function of a large diverse set of proteins comprising the proteome, thereby, yielding important insights into the atomic details that underlie biophysical processes such as photosynthesis, respiration, and DNA replication.

One important theoretical tool in biophysical research is atomistic molecular dynamics (MD) simulation. Classical MD simulation were carried out in the hope of understanding the properties of assemblies of molecules in terms of their structures and the microscopic interactions between them. This serves as a complement to conventional experiments, enabling us to learn something new, something that is not easy or even cannot be found out through conventional experiments [18, 47, 48, 61, 70]. Of course, experiments play an essential role in validating the simulation methodology: comparisons of simulation and experimental data serve to test the accuracy of the calculated results and to provide criteria for improving the methodology. Computer simulations act as a bridge between microscopic length and time scales and the macroscopic world of the laboratory. Unfortunately, due to the intrinsic computation-intensive property of computer simulation, at present, atomistic simulation's impact has been limited by the mismatch between the length times and time scales that the method can currently access and those needed for success. As a direct result, most details of the macroscopic world still cannot be revealed. In order to improve the ability of MD simulation to accurately describe proteins requires synergistic advances in methods [8, 29, 67, 46, 42, 74, 59], computer architectures [12, 10, 39] and

fine grained parallel software [65, 64, 41]. Indeed, this route is yielding promising results [35, 15, 65].

In classical molecular dynamics, which is an N-body problem, the evolution of a group of interacting particles is determined through numerical integration of the equations of motion. At each time step, forces on particles are computed; and the equations of motion are integrated to update the velocities and positions of the particles. The force computation herein is based on an empirical potential function or force field [15, 20, 56]. A typical empirical-potential-function usually consists of bonded interactions and non-bonded interactions. The former usually include the bond potentials, the angle potentials and the torsion potentials; the latter include Lennard-Jones interaction and electrostatic potential energy. The first four are short-ranged interactions, while the last one is a long-ranged interaction.

The computation of the long ranged interactions is well recognized to be the most challenging task in computer MD simulations. Intuitively, the interaction between each particle and everyone else could be directly computed, which means a double loop over all the particles is necessary. However, this naive method would cause a complexity of $O(N^2)$, which is prohibitive in MD simulation, while the other four short ranged potentials scale linearly when a cutoff is employed. (N herein is the number of charged particles in the simulation box.) Because models of proteins have components with large partial charges, unlike the short ranged potentials, the long ranged electrostatic potential cannot be approximated simply by cutting off force interactions between pairs of particles further apart than some cut-off distance. This statement has been empirically proved valid in the simulation of DNA, when truncation was used on long-range electrostatic interactions [60].

The most commonly used technique for handling these long range interactions

is the Ewald summation method[30]. The idea of it is to split the very slowly (not even unconditionally) converging sum over the electrostatic potential into two sums which converge exponentially fast. Still, this method suffers from two deficits: First, it is computationally demanding, since one part of the problem is solved in reciprocal space, thereby Fourier transformations being needed. Second, the algorithm scales like N^2 or at best $N^{3/2}$, if cutoffs are optimized with respect to the splitting parameter [62]. Several methods have been proposed to tackle the first problem, e.g., tabulation of the complete Ewald potential [71] or the use of polynomial approximations [5, 62]. Regardless of the difficulty of a computational overhead which might strongly increase with the desired accuracy, all these techniques do not solve the second problem: the unfavorable scaling with particle numbers.

The essential idea is *not* to avoid the Fourier transforms but to modify the algorithm in such a way that the Fast Fourier transformation (FFT), with complexity of order $N \log N$, could be employed in the reciprocal-space part. In the mean while, cutoff could be employed in the real-space part, with linear complexity. Because of the intrinsic discrete property of FFT, there raises another problem on how to discretize the continuous potential and how to minimize the discretization errors. At present there exist several mesh-based techniques for the Ewald summation method, the original Particle-Particle-Particle-Mesh Ewald (P^3ME) method of Hockney and Eastwood [46], the Particle Mesh Ewald (PME) method of Darden *et al.* [22], and Smooth Particle Mesh Ewald (SPME) method first proposed by Essmann *et al.* [29]. In this dissertation, SPME is the one we are going to investigate and implement on parallel platforms.

At present, in order to improve the accuracy and efficiency of the treatment of long range forces in biophysical MD simulations, several synergistic approaches involving

methods developments, parallel software developments and the use of the state-of-the-art supercomputer architectures have been employed. For instance, NAMD [64] is an MD application that can exhibit high parallel performance on several different computer architectures. The highlight of NAMD is that the package combines force and spatial decompositions together and utilizes processor virtualization to solve load imbalance problems. Another example is Blue Matter [41], designed specifically for use on the BlueGene/L supercomputer, which has 3D torus communication network [39]. The novel architecture of BlueGene/L, especially the hardware-embedded global broadcast routing and the fast nearest neighbor communication ability, provides MD application an appropriate platform for obtaining high parallel performance. Inspired by NAMD and Blue Matter, our group intend to design and implement another fine grained MD application, MDoC, specifically for use on the QCDOC supercomputer which has a 6D torus communicational architecture. The framework of MDoC was designed and implemented by P.Rissland [69], and the multi-time stepping integral method was modified and embedded into the package by G.W.Han [43]. In this dissertation, a modification of the SPME method is introduced into MDoC package that reduces both the computation and communication associated with the parallel 3D-FFT even at small grid spacing.

In order to achieve a high scalability, the most challenging task of every parallel implementation of mesh-based Ewald sum methods is the communication optimization in 3D-FFT calculation, especially on thousands-of-node parallel platforms. This is all because 3D FFT is a global-scope computation and on parallel platforms, the most time-consuming global communication is unavoidable. People who ignore the importance of the communication optimization of 3D-FFT can definitely not achieve high scalability. In the third section of this dissertation, we will present a fine-grained parallel 3D-FFT framework which takes full advantage of the 24 off-node links of each

processor on QCDOC. The framework is designed with the capability of embedding any existing sequential FFT package, for instance FFTW[4], so that it can provide more flexibility. Not only the complex-to-complex 3D-FFTs, but also the real-to-complex 3D-FFTs are supported by this framework.

More optimization has been done when 3D-FFT acts as a bridge to connect the real-space interactions and the reciprocal-space interactions in SPME. It is not difficult to notice that the approximation error, which is produced when Euler exponential spline is employed to approximate the Fourier components of the charge density, does not vary uniformly with the reciprocal-space vector, but greatly increases as the vector approaches the edge of the simulation box (in reciprocal space). This means that given a big enough reciprocal-space lattice, it is simply not useful to include inaccurate high Fourier coefficients in the reciprocal space summation. Indeed, it can be shown to be computationally inefficient to do so [68], especially when communication is dominant in parallel 3D-FFT calculation. In our modified SPME, the spherical cutoff is implemented in 3D FFT itself, thereby reducing both the Flops of the scalar implementation and communication volume of the parallel implementation. The details of the theoretical analysis and the implementation will be introduced in Section 5. The accuracy analyses shown in Section 6 represent that, given a fixed spherical cutoff, our modified SPME with cutoff 3D-FFT can provide an approximation to the Ewald sum within desired accuracy. In the meantime, the efficiency analyses show that with the help of the spherical cutoff, 3D-FFT has no longer been the dominantly time-consuming part in SPME. Moreover, when moderate-sized lattices are being used, charge assignment and force interpolation become the bottleneck of SPME on the parallel platforms less than thousands of processors. Some schemes are also proposed at the end of Section 6 for future improvement.

The dissertation is organized as follows: In Section 2, the functional form of present generation Molecular Dynamics is reviewed, in conjunction, with several common methods to handle the long ranged electrostatic potential, among which the Smooth Particle Mesh Ewald method is discussed in detail with attention paid to improve the method scalar accuracy and efficiency. Intramolecular force terms and Lennard-Jones type forces are not of interest, here [66]. A brief description of the QCDOC architecture is then given in Section 3. The implementation details of parallel framework for 3D-FFT on QCDOC are presented in Section 4, followed by the efficiency analysis and the numerical results. SPME implementation, given in Section 5 is based on this framework, in addition, we added in a spherical cutoff in 3D-FFT itself. Results are presented in Section 6 for two protein/water systems, β-hairpin in streptococcal protein G and HIV-1 protease. Finally we make our conclusions and propose the future work in Section 7.

2 Long-range forces in Molecular Dynamics

In this section, the main task is to theoretically explain the underlying reason why spherical cutoff 3D-FFT could be introduced into SPME algorithm, followed by the implementation details in scalar computing platforms and the prediction of Flop reduction. Before this explanation, basic Molecular Dynamics simulation knowledge will be reviewed, including force fields, periodic boundary conditions and popular methods to compute long-range forces.

2.1 Review of Molecular Dynamics

Classical molecular dynamics solves Hamilton's equation of motion for a set of (generalized) coordinates $\mathbf{q}_i$ and momenta $\mathbf{p}_i$ which are given by

$$\begin{aligned} \dot{\mathbf{p}}_i &= -\frac{\partial H}{\partial \mathbf{q}_i} \\ \dot{\mathbf{q}}_i &= \frac{\partial H}{\partial \mathbf{p}_i} \\ H\left(\{\mathbf{p}_i, \mathbf{q}_i\}\right) &= \sum_i \frac{|\mathbf{p}_i|^2}{2m_i} + \phi\left(\{\mathbf{q}_i\}\right) \end{aligned}$$

where m_i is the mass of the ith particle and ϕ represents the potential energy of the whole system. In many cases the potential energy is conservative and Cartesian coordinates $\mathbf{r}_i$ and velocity $\mathbf{v}_i$ are usually used; Newton's equation of motion $m_i\ddot{\mathbf{r}}_i = \mathbf{F}_i(\mathbf{R})$ is exploited, where $\mathbf{R}$ is the union of the position vectors of all particles in the system and $\mathbf{F}_i$, for conservative force fields, is a function of the positions only,

$$\mathbf{F}_i = -\frac{\partial}{\partial \mathbf{r}_i}\phi(\mathbf{R})$$

The trajectories of the interacting particles are calculated through the approximate solution of the Hamiltonian equations of motion base on empirical potential function or force field. Current generation force fields such as harmm [56] and Amber [15, 20], take the form

$$\begin{aligned} \phi_{total} &= \phi_{inter} + \phi_{intra} \ , \\ \phi_{intra} &= \phi_{bonds} + \phi_{angles} + \phi_{dihedral} \ , \qquad (1) \\ \phi_{inter} &= \phi_{Coulomb} + \phi_{vdw,rep} \ . \qquad (2) \end{aligned}$$

The three terms in Equation 1 describe the stretching, bending, and torsional bonded interactions,

$$\begin{aligned} \phi_{bonds} &= \sum_{\text{bonds}\, i} k_i^{\text{bond}} (l_i - l_{0i})^2 \ , \\ \phi_{angles} &= \sum_{\text{angles}\, i} k_i^{\text{angle}} (\theta_i - \theta_{0i})^2 \ , \\ \phi_{dihedral} &= \sum_{\text{dihedral}\, i} k_i^{\text{dihedral}} (1 + \cos(n_i \omega_i - \gamma_i)) \ , \end{aligned}$$

where *bonds* count each covalent bond in the system, *bends* are the angles between each pair of covalent bonds sharing a single atom at the vertex, and *dihedral* describes atom pairs separated by exactly three covalent bonds with the central bond subject to the torsion angle ω.

The final two terms in Equation. 2 describes interactions between nonbonded atom

pairs,

$$\phi_{vdw,rep} = \sum_i \sum_{j>i} \left[\frac{A_{ij}}{|\mathbf{r}_{ij}|^{12}} - \frac{B_{ij}}{|\mathbf{r}_{ij}|^6} \right] , \quad (3)$$

$$\phi_{Coulomb} = \sum_i \sum_{j>i} \frac{q_i q_j}{4\pi\varepsilon_0 |\mathbf{r}_{ij}|} , \quad (4)$$

which correspond to the van der Waal's forces (approximated by a Lennard-Jones 6-12 potential) and electrostatic interactions, respectively.

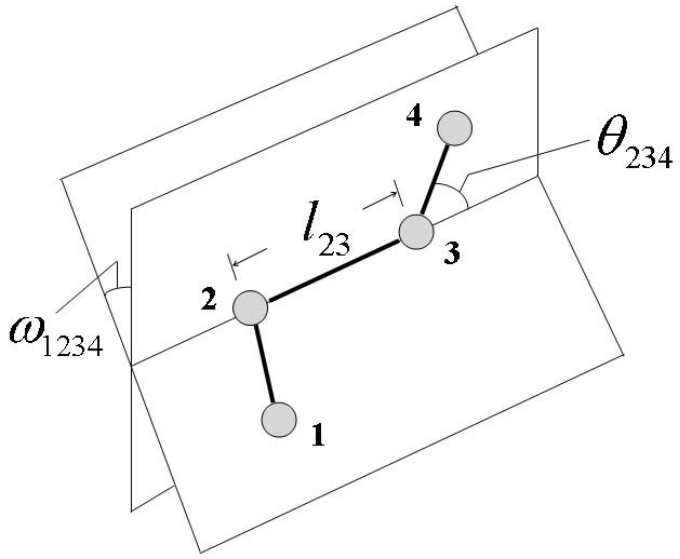


Figure 1: Geometry of single chain molecule.

This figure illustrates the bond length l_{23}, bend angle θ_{234} and torsion angle ω_{1234} (adapted from [7]).

All the parameters k_i^{bond}, θ_{0i}, etc., in the above interaction functions are determined through a combination of empirical techniques and quantum mechanical calculations. The force field is then tested for fidelity in reproducing the structural, dynamic, and thermodynamic properties of small molecules that have been well-characterized experimentally, as well as for fidelity in reproducing bulk properties. MDoC [69] is able to use the parameterizations from AMBER [15, 20] force field

specifications.

Assuming the force $\mathbf{F}_n = \mathbf{F}(\mathbf{r}_n)$ is already computed with the given force field, MDoC uses the Verlet method [36] for NVE ensemble simulations. The "velocity-Verlet" method obtains the position and velocity at the next time step $(\mathbf{r}_{n+1}, \mathbf{v}_{n+1})$ from the current one $(\mathbf{r}_n, \mathbf{v}_n)$ in the following way:

$$\begin{aligned}
\text{"half-kick I"}\ \mathbf{v}_{n+1/2} &= \mathbf{v}_n + M^{-1}\mathbf{F}_n \cdot \Delta t/2\ , \\
\text{"drift"}\ \mathbf{r}_{n+1} &= \mathbf{r}_n + \mathbf{v}_{n+1/2}\Delta t\ , \\
\text{"force compute"}\ \mathbf{F}_{n+1} &= \mathbf{F}(\mathbf{r}_{n+1})\ , \\
\text{"half-kick II"}\ \mathbf{v}_{n+1} &= \mathbf{v}_{n+1/2} + M^{-1}\mathbf{F}_{n+1} \cdot \Delta t/2\ .
\end{aligned}$$

where M is the mass of this particle. For a fixed time period, the method exhibits a error proportional to Δt^2. For the detail of the implementation of "velocity-Verlet" on QCDOC, please refer to P.Rissland's doctorate dissertation [69].

2.2 Long-range forces and Ewald summation

In a complete MD simulation package, every detail mentioned in the previous subsection needed to be taken care of. (For the details of MDoC package, please refer to P.Rissland's doctorate dissertation [69] for intramolecular potential calculation and "velocity-Verlet" integration method, and G.W.Han's doctorate dissertation [43] for multiple-time-step integration methodology.) However, in this dissertation, only long-range potential and forces are of interest, because this is the biggest challenge in every MD simulation, especially on parallel platforms [40, 41, 65, 69].

Numerically, to avoid surface effects at the boundary of the simulated systems,

periodic boundary conditions are often used in MD simulation. Dropping the prefactor $1/4\pi\varepsilon_0$, the two parts of the intermolecular potential (Equation. 4 and 3) are usually given by

$$\begin{aligned} \phi_{inter}^{Coul}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) &= \frac{1}{2} \sum_{\hat{\mathbf{S}}} {\sum_{ij}}' \left[\frac{q_i q_j}{|\, \mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}|} \right] \\ \phi_{inter}^{Vdw,Rep}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) &= \frac{1}{2} \sum_{\hat{\mathbf{S}}} {\sum_{ij}}' \left[\frac{A_{ij}}{|\, \mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}|^{12}} - \frac{B_{ij}}{|\, \mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}|^{6}} \right] \end{aligned} \tag{5}$$

where $\mathbf{r}_{ij}$ is the nonbonded interatomic distance vector, A_{ij} and B_{ij} are the Lennard-Jones parameters (short range repulsion and Van der Waals interactions, respectively), and the q_i's are the charges on the particles. The sum over replicas, $\hat{\mathbf{S}} = \{l, m, n\}$, indicates that periodic boundary conditions have been employed, the prime restricts the sum to $i \neq j$ when $\hat{\mathbf{S}} = 0$. The columns of the matrix, $\overleftrightarrow{\mathbf{h}}$, contain the Cartesian components of the three vectors, $\{\mathbf{a}, \mathbf{b}, \mathbf{c}\}$ which describe the parallelepided bounding the system. The volume is given by $V = \mathbf{a} \times \mathbf{b} \cdot \mathbf{c} = \det \overleftrightarrow{\mathbf{h}}$ in the usual way. (For a cubic cell of side, L, $\overleftrightarrow{\mathbf{h}}$ is diagonal and $V = L^3$.)

Periodic boundary conditions [8] are designed to treat systems at finite concentration with a minimum of finite size effects. It is not appropriate to employ this method to treat systems at infinite dilution. Therefore, it is important to note that NMR and ESR experiments of biomolecular systems are performed at fairly substantial concentration (millimolar) not at infinite dilution. In addition, extensive tests on small model systems have revealed that artifacts due to periodic images are small, particularly, in water solution, a medium with a high dielectric constant. Thus, the technique becomes a small approximation. Briefly, errors are dominated by spurious correlations. That is, the periodic images precisely mirror the molecules in the central cell, rather than taking on "independent" conformations. For instance, a two dimen-

sional version of such a periodic-boundary-condition system is shown in Figure 2. The central cell is labeled O, with all its replicas labeled A,B,C,... All the particle images, e.g. 1_A, 1_B,... mirror the position, the movement of their original copy, e.g. particle 1.

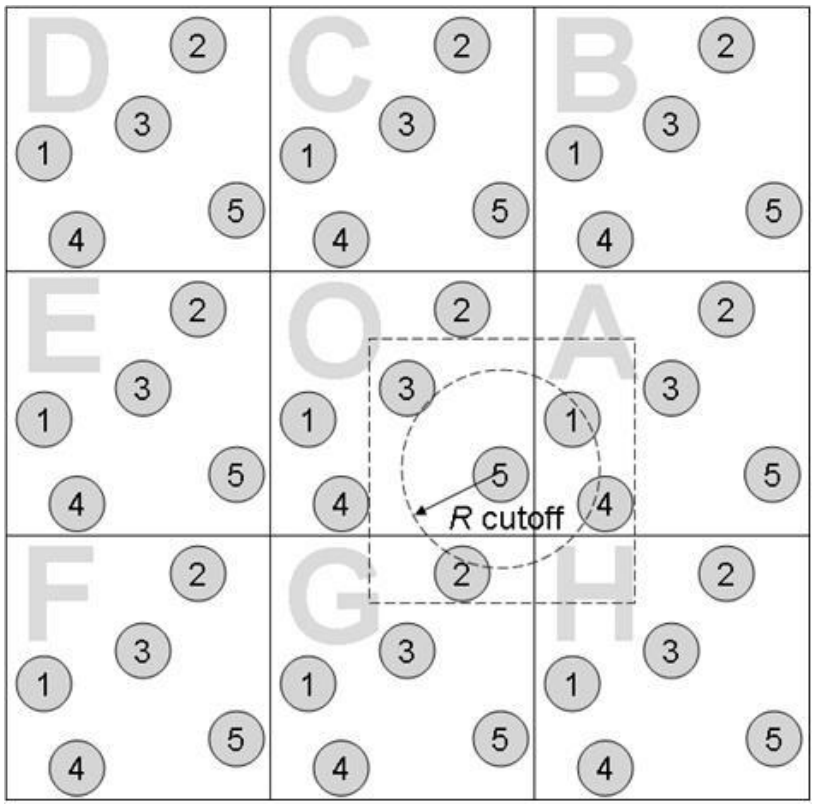


Figure 2: Demo of Periodic Boundary Condition.

A two-dimensional periodic system which represents the minimum image convention and cutoff scheme. The central box contains five particles; the dashed box constructed with particle 5 at its center also contains five particles. The dashed circle represents a potential cutoff (adapted from [73])

The electrostatic and Van der Waals forces under periodic boundary conditions are assumed to be summed over all replicas. Therefore, a force evaluation seems to scale like N^2 or greater. Fortunately, well-defined methodology exists that reduces

the computational burden to $N \log N$. Here, the commonly used Ewald and Smooth Particle Mesh Ewald summation techniques for the evaluation of the electrostatic energy are examined, in conjunction, with a spherical truncation of the VanderWaals or Lennard-Jones interaction.

First, since the Lennard-Johns potential is short range in nature (decaying fast than $1/r^3$), these terms can be treated using either a sharp or smooth spherical cutoff function within the first or nearest image approximation,

$$\phi_{inter}^{Vdw,Rep}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) \;=\; \frac{1}{2}\sum_{ij}{}' \chi(|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|, r_c) \left[\frac{A_{ij}}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|^{12}} - \frac{B_{ij}}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|^{6}} \right].$$

This reduces the scaling of the calculation to order N at fixed cutoff, r_c. In the first or nearest image approximation, the components of the integer vector, $\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}$, are selected such that the distance, $|\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|$, achieves its minimum value. For instance, in Figure 2, none of the particles or particle images, except of 3_{O}, 1_{A}, 4_{A} and 2_{G}, have non-zero influence on particle 5 after applying spherical cutoff function. A discussion of issues related to the use of switching functions, $\chi(r, r_c)$, in MD simulations, and mean field corrections to the truncated form of the potential energy are given in Appendix A in [32].

Unfortunately, the Coulomb interaction cannot be treated accurately using spherical truncation, since the infinite summation of charge-charge interactions for a charge-neutral system in Equation 5 is conditionally convergent, meaning that the result of the summation depends on the order in which it is taken. One solution suggested by Ewald [30], which is considered more reliable than a spherical truncation scheme [33, 54], involves rewriting the Coulomb energy using the identity, $1/r =$

$[\mathrm{erf}(\alpha_{ewd}r) + \mathrm{erfc}(\alpha_{ewd}r)]/r$,

$$\phi^{(\mathrm{Coul})}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = \phi^{(\mathrm{Coul})}_{\mathrm{dir}} + \phi^{(\mathrm{Coul})}_{\mathrm{rec}} + \phi^{(\mathrm{Coul})}_{\mathrm{self}} + \phi^{(\mathrm{Coul})}_{\mathrm{surface}} \ , \tag{6}$$

where

$$\phi^{(\mathrm{Coul})}_{\mathrm{dir}}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = \frac{1}{2}\sum_{\hat{\mathbf{S}}}{}' \sum_{ij} q_i q_j \frac{\mathrm{erfc}(\alpha_{ewd}|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|)}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|} \ , \tag{7}$$

$$\phi^{(\mathrm{Coul})}_{\mathrm{rec}}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = \frac{1}{2}\sum_{\hat{\mathbf{S}}} \sum_{ij} q_i q_j \frac{\mathrm{erf}(\alpha_{ewd}|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|)}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|} \ , \tag{8}$$

$$\phi^{(\mathrm{Coul})}_{\mathrm{self}}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = -\frac{\alpha_{ewd}}{\pi^{1/2}} \sum_i q_i^2 \ , \tag{9}$$

$$\phi^{(\mathrm{Coul})}_{\mathrm{surface}}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = \frac{2\pi}{(2\varepsilon_s + 1)V} \left| \sum_i q_i \mathbf{r}_i \right|^2 \ . \tag{10}$$

The Ewald summation in Equation 6 contains four terms: direct sum $\phi^{(\mathrm{Coul})}_{\mathrm{dir}}$, reciprocal sum $\phi^{(\mathrm{Coul})}_{\mathrm{rec}}$, self-energy $\phi^{(\mathrm{Coul})}_{\mathrm{self}}$ and surface energy $\phi^{(\mathrm{Coul})}_{\mathrm{surface}}$. The self-energy term is a trivial constant, while the surface term is usually neglected by assuming the "tin-foil" boundary condition $\varepsilon_s = \infty$, which partly due to the dielectric constant of water ($\varepsilon_s \approx 80$) being much larger than 1. The direct sum term, which involves the complementary error function, $\mathrm{erfc}(x) = \frac{2}{\sqrt{\pi}} \int_x^\infty \exp(-t^2)dt$, is short range and can be treated using a spherical cutoff as above. The reciprocal sum term (long range term) can be evaluated in reciprocal space as described in Appendix B. Introducing the Fourier components of the charge density,

$$\bar{\mathrm{n}}(\mathbf{g}) = \sum_k q_k \exp(-i\mathbf{g} \cdot \mathbf{r}_k),$$

and a spherical cutoff in real space, the potential energy becomes

$$\begin{aligned}
\phi^{(\mathrm{Coul})}(\mathbf{R},\overleftrightarrow{\mathbf{h}}) &= \frac{1}{2V}\sum_{\hat{\mathbf{g}}\neq 0}\left[|\bar{\mathrm{n}}(\mathbf{g})|^2 W(\mathbf{g})\right] - \frac{\alpha_{ewd}}{\pi^{1/2}}\sum_i q_i^2 \\
&+ \frac{1}{2}{\sum_{ij}}' q_i q_j \chi(|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|, r_c)\frac{\mathrm{erfc}(\alpha_{ewd}|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|)}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|} \qquad (11) \\
W(\mathbf{g}) &= \left[\frac{4\pi}{g^2}\right]\exp\left(-\frac{g^2}{4\alpha_{ewd}^2}\right), \qquad (12)
\end{aligned}$$

where

$$\mathbf{g} = 2\pi\hat{\mathbf{g}}\overleftrightarrow{\mathbf{h}}^{-1}.$$

Note, a spherical cutoff in reciprocal space, g_c, can be employed to truncate the sum over the (integer) reciprocal lattice vectors, $\hat{\mathbf{g}} = \{l, m, n\}$. This is reasonable since the corresponding convergence function, $\exp[-g^2/(4\alpha_{ewd}^2)]$, decays rapidly at large argument. The functional form of real space convergence function, $\mathrm{erfc}(\alpha_{ewd} r)$, and its reciprocal space counterpart, indicate that the scaled cutoffs, (g_c/α_{ewd}) and $(r_c\alpha_{ewd})$, are of interest. Thus, once r_c is selected, α_{ewd} and g_c are, effectively, determined for a fixed truncation error. (In a cubic box, it is useful to define $\hat{g}_c = g_c L/(2\pi)$). A more detailed discussion of reciprocal space cutoff effects on Ewald methods can be found elsewhere.

The Ewald method scales poorly with increasing number of atoms, N, at constant density ($\rho = N/V$). At fixed real space and reciprocal space cutoffs, $\{r_c, g_c\}$ obtained by taking α_{ewd} constant, the computational cost of the reciprocal space part of the calculations scales like N^2. Similarly, at fixed number of reciprocal space vectors, $\hat{\mathbf{g}}$ obtained by taking $\alpha_{ewd} V^{-1/3}$ to be constant, the computational cost of the real space part scales like N^2. Therefore, by judiciously adjusting the real space and reciprocal

space cutoff values (i.e. α_{ewd}), the Ewald method can be made to scale as $N^{3/2}$.

2.3 Mesh based Ewald summation

As we mentioned, the scalability of regular Ewald summation is poor with increasing number of atoms. The computation of order N^2 or even $N^{3/2}$ is prohibitive for heterogenous macromolecular systems. To make Ewald summation practical in actual simulation, several mesh based Ewald summation methods have been investigated, including Smooth Particle Mesh Ewald (SPME) [29], Particle Mesh Ewald (PME) [22, 63] and Particle-Particle-Particle Mesh Ewald (P^3ME) [46]. All these methods employ an interpolation scheme to generate an approximation to the atomic charge density that can be calculated in $\mathcal{O}(N \log N)$ at constant cutoff, $\{r_c, g_c\}$, using a three dimensional real-to-complex fast Fourier transform (3D-FFT). They share a similar formula,

$$\begin{aligned} \phi_{tot}^{(\mathrm{Coul})}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) &= \frac{1}{2V} \sum_{\hat{\mathbf{g}}} \left[|\bar{\mathrm{n}}^{(\mathrm{Interp})}(\mathbf{g}, n, L_{\mathrm{FFT}})|^2 W^{(\mathrm{Kern})}(\mathbf{g}) \right] - \frac{\alpha_{ewd}}{\pi^{1/2}} \sum_i q_i^2 \\ &+ \frac{1}{2} \sum_{ij}^{'} \chi(|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|, r_c) q_i q_j \frac{\mathrm{erfc}(\alpha_{ewd} |\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|)}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}} \hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|}. \end{aligned} \quad (13)$$

The differences between different methods are $\bar{\mathrm{n}}^{(\mathrm{Interp})}(\mathbf{g}, n, L_{\mathrm{FFT}})$, caused by different interpolation schemes, and $W^{(\mathrm{Kern})}(\mathbf{g})$. Briefly speaking, SPME employs Euler exponential spline interpolation [72, 19] to generate $\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, n, L_{\mathrm{FFT}})$ and uses the same $W(\mathbf{g})$ as in Equation 12. PME employs Lagrangian polynomial interpolation [51] to generate $\bar{\mathrm{n}}^{(\mathrm{Lagrange})}(\mathbf{g}, n, L_{\mathrm{FFT}})$ and uses the same $W(\mathbf{g})$ as SPME. P^3ME employs the same interpolation scheme as PME, however uses different $W^{(\mathrm{Kern})}(\mathbf{g})$, intending to make mesh calculation to be as close as possible to the original continuum problem.

In this dissertation, we focus on SPME and its implementation, so the derivative of the SPME formula is given in Appendix B in detail. Brief introductions of PME and P^{3}ME are given in Appendix D.

In SPME, the approximation is generated by simply substituting the SPME expression for the Fourier components of the charge density, $\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, n, L_{\mathrm{FFT}})$ into the potential energy to yield,

$$\begin{aligned}\phi_{tot}^{(\mathrm{Coul})}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) &= \frac{1}{2V}\sum_{\hat{\mathbf{g}}}\left[|\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, n, L_{\mathrm{FFT}})|^2 W(\mathbf{g})\right] - \frac{\alpha_{ewd}}{\pi^{1/2}}\sum_i q_i^2 \\ &+ \frac{1}{2}{\sum_{ij}}' \chi(|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|, r_c) q_i q_j \frac{\mathrm{erfc}(\alpha_{ewd}|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|)}{|\,\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}_{ij}^{(\mathrm{near})}|}. \quad (14)\end{aligned}$$

The same $W(\mathbf{g})$ is employed as Equation 12 and spherical truncation of reciprocal and real spaces is used. The required 3D-FFT is real to complex which saves a factor of 2x in scalar and 2x in communication cost in parallel. It is, therefore, clear that SPME simply approximates the reciprocal space part of the Ewald sum at cutoff $\hat{g}_c$.

As discussed in Appendix C, the error in the Euler exponential spline approximation to the charge density does not vary uniformly with $\hat{\mathbf{g}}$ but greatly increases as $2\hat{\mathbf{g}}$ approaches L_{FFT} (e.g. scaling as $[2\hat{\mathbf{g}}/\text{Ł}_{\mathrm{FFT}}]^n$). It is, therefore, necessary to select $L_{\mathrm{FFT}} > 2\hat{g}_c$ as it is simply not useful to include inaccurate high Fourier coefficients in the reciprocal space sum. Indeed, it can be shown to be computationally inefficient to do so [68].

In our new procedure, a spherical cutoff in reciprocal space of the SPME approximated reciprocal space part of the Ewald sum is introduced. The EES interpolation order, n, and the 3D-FFT size, L_{FFT} are then tuned to match the error of the directly computed, spherically truncated, reciprocal space part of the Ewald sum relative to

the directly computed, reciprocal space contribution of the Ewald sum evaluated at very large cutoff. That is, a direct evaluation of the reciprocal space part of the Ewald summation yields the exact result at the cutoff of interest. For example, it would not be sensible to tune the SPME parameters to match the results of truncated direct Ewald sum more closely then the results of the truncated direct Ewald sum, itself, match the exact result (.e.g the reciprocal space sum evaluated directly at very large cutoff). Of course, introducing the spherical cutoff and paying careful attention to accuracy critera would not, alone, increase the efficiency of the method. The spherical cutoff must be implemented within the 3D-FFT reducing both the Flops of the scalar computation and communication of the parallel implementation following the standard procedure used in plane-wave based DFT electronic structure codes [38, 75]. In this dissertation, we focus on the implementation and efficiency of modified SPME method on parallel computing platform, so before the implementation details are presented, some brief introductions of our parallel platform, QCDOC, will be made next.

3 QCDOC: A low power, large scale parallel supercomputer

QCDOC (Quantum Chromodynamics on a Chip)[12, 11] is a massively parallel supercomputer developed by a group of researchers at Columbia University, the RIKEN BNL Research Center (RBRC), IBM's Watson Research Laboratory, and the UKQCD collaboration. It utilizes the latest low-power micro-electronic architectural concepts including system-on-chip design. The use of system-on-chip technology has enabled the designers of QCDOC to create a massively parallel supercomputer that can be air cooled, cheaply. In detail, the QCDOC processing component consists of a single ASIC (application specific integrated circuit) and a standard DDR RAM module. Larger machines can be built by connecting many of the smaller processing components together. The QCDOC ASIC contains a PowerPC 440 running at the low clock speed 500MHz as its processing core, a 1Gflop 64-bit floating point unit(FPU), and 4MB of embedded DRAM, with a peak bandwidth of 8Gbyte per second transfer rate to the processor controlled by DMA for direct access to the embedded memory. The primary goal of the compute node is to run a small set of floating-point intensive scientific applications extremely efficiently at low power with low cooling cost.

In terms of parallel structure, QCDOC is a six dimensional torus platform. The highspeed serial communication between nearest neighbors has a bandwidth of 500 Mbits/s per link when it runs at 500MHz. The communication between the links is managed by the SCU (serial communication unit) in the ASIC. There are a total of 24 off-node links, 2 for each direction along each dimension, leading to a total off chip bandwidth of 12Gbit per second. QCDOC has implemented its own operating system called QOS to handle nearest neighbor communication calls, and global operations,

which is done via the pass through hardware.

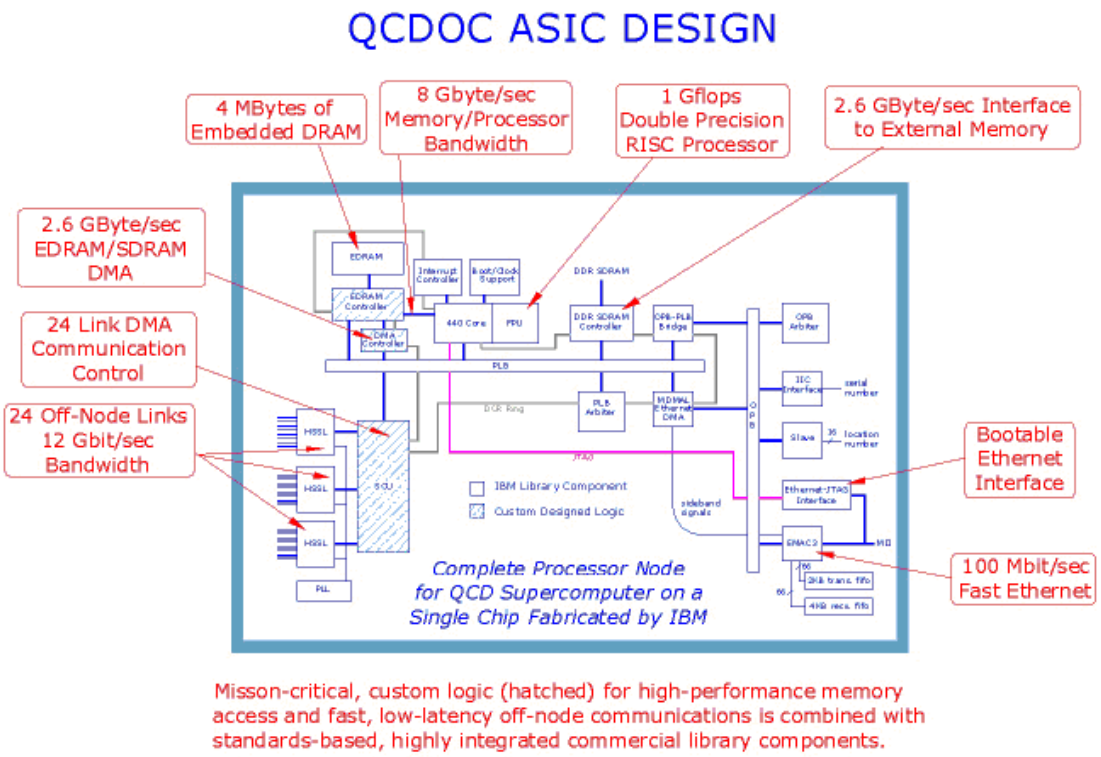


Figure 3: An overview of the QCDOC ASIC design.

With the ordinary inexpensive computing nodes, and the unique interconnection, QCDOC can deliver a price/performance of less than $1 per sustained Mflops. Originally, QCDOC was specifically designed and optimized to perform Quantum Chromodynamics (QCD) simulations which model the behavior of matter on sub-nuclear length scales. However, the versatility of the QCDOC hardware allows applications such as MD Simulation [23] to be ported and run at high efficiency. It will, of course, be interesting to see if the 6D-torus communication architecture can be efficiently employed to treat long range electrostatic interactions via SPME.

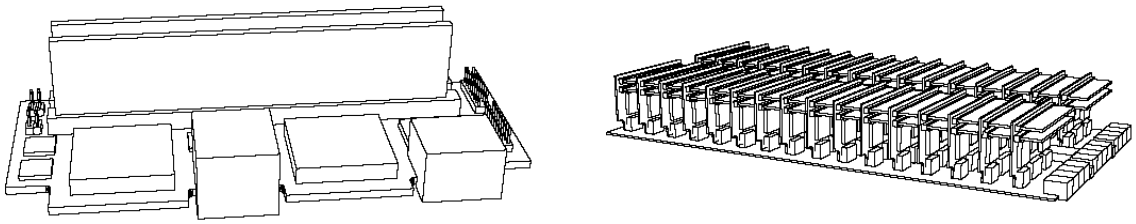


Figure 4: QCDOC daughter board and mother board.

A 2-node daughter board is shown on the left picture, with two QCDOC ASICs and two 128Mbyte DDR SDRAMs. The central connector carries 40 differential pairs making up the off-daughter board 6-dimensional serial communications network. A single mother board is shown on the right picture. Two rows of 16 daughter boards with two nodes each provide a total of 64 nodes.

4 3D FFT and its parallel implementation on QCDOC–6D torus supercomputer

No matter which mesh based Ewald summation is being employed, 3D FFT is a common tool acting the transferring between real space and reciprocal space in order to decrease the computational order to $N \log N$. Moreover, not only for handling long-range forces in MD simulation, but also for many other physical, chemical, engineering research fields, 3D FFT acts as a very popular and useful tool. In this dissertation, we designed a parallel framework for both complex-to-complex 3D FFT and real-to-complex 3D FFT for common use and a specific spherical cutoff real-to-complex 3D FFT to treat long-range forces in SPME. In this section, the basic FFT knowledge would be reviewed briefly, followed by the implementation detail and the evaluation of efficiency of the 3D complex-to-complex FFT framework on QCDOC. In order to retain the completeness of the description of SPME parallel implementation, we defer the description of the spherical cutoff 3D FFT to the next section.

4.1 Basic 3D FFT

Let $f(x, y, z)$ be a complex/real function taking integer-triplet argument, i.e. $x = 0, ..., N_x - 1$, $y = 0, ..., N_y - 1$, $z = 0, ..., N_z - 1$. We define its 3D FFT $\bar{f}(g_x, g_y, g_z)$

as,

$$\begin{aligned}
\bar{f}(g_x, g_y, g_z) &= \sum_{x=0}^{N_x-1} \sum_{y=0}^{N_y-1} \sum_{z=0}^{N_z-1} f(x,y,z) W_{N_x}^{x g_x} W_{N_y}^{y g_y} W_{N_z}^{z g_z}, \\
g_x &= 0, ..., N_x - 1, \\
g_y &= 0, ..., N_y - 1, \\
g_z &= 0, ..., N_z - 1,
\end{aligned}$$

where $W_{N_\alpha} = \exp(\frac{-2\pi i}{N_\alpha})$, $N_\alpha \in \mathbb{Z}$ and $\alpha = x, y, z$.

It consists of a series of three 1D-FFTs. First of all, one FFT is to transform along z-axis for all x, y to obtain $\hat{f}(x, y, g_z)$, i.e.

$$\begin{aligned}
\hat{f}(x, y, g_z) &= \sum_{z=0}^{N_z-1} f(x,y,z) W_{N_z}^{z g_z}, \\
x &= 0, ..., N_x - 1, \\
y &= 0, ..., N_y - 1, \\
g_z &= 0, ..., N_z - 1,
\end{aligned}$$

Secondly, one FFT is to transform along y-axis for all x, g_z to obtain $\check{f}(x, g_y, g_z)$, i.e.

$$\begin{aligned}
\check{f}(x, g_y, g_z) &= \sum_{y=0}^{N_y-1} \hat{f}(x,y,g_z) W_{N_y}^{y g_y}, \\
x &= 0, ..., N_x - 1, \\
g_y &= 0, ..., N_y - 1, \\
g_z &= 0, ..., N_z - 1,
\end{aligned}$$

At last, one FFT is to transform along x-axis for all g_y, g_z to obtain $\bar{f}(g_x, g_y, g_z)$, i.e.

$$\begin{aligned} \bar{f}(g_x, g_y, g_z) &= \sum_{x=0}^{N_x-1} \check{f}(x, g_y, g_z) W_{N_x}^{x g_x}, \\ g_x &= 0, ..., N_x - 1, \\ g_y &= 0, ..., N_y - 1, \\ g_z &= 0, ..., N_z - 1, \end{aligned}$$

Note that according to the symmetry of the transform equation, it doesn't matter in which order three dimensions of 1D FFTs are calculated. We can implement it in any order.

4.2 Parallel implementation on QCDOC

The topology of QCDOC network is a truncated 6D torus that can be expressed as $p_x \times 2 \times p_y \times 2 \times p_z \times 2$, where $p_i = 2^{c_i}$ represents the number of nodes in each of the three regular dimensions and c_i can be any positive integer. The other three dimensions are truncated, with two nodes in each of them. Along each regular dimension, there is a bidirectional link between each pair of nearest neighbors; while along each truncated dimension, there are two bidirectional links connecting the two nearest nodes, which can do the data communication independently and simultaneously. Technically, we manipulate QCDOC as if every processor were directly connected to 12 nearest neighboring processors with bidirectional communication channels. Let $P_x = p_x \times 2$, $P_y = p_y \times 2$ and $P_z = p_z \times 2$, i.e. P_i is the number of nodes in two-dimensional mesh consisting of one regular dimension and one truncated dimension. In the parallel framework implementations, we adopted a volume decomposition method for the

FFT lattice. Without loss of generality, we assume the original function is of size $N = N_x \times N_y \times N_z$.

Initially, a node with coordinate (a, b, c, d, e, f), where

$$\begin{aligned} a &= 0, ..., p_x - 1, \\ c &= 0, ..., p_y - 1, \\ e &= 0, ..., p_z - 1, \\ b, d, f &= 0, 1 \qquad\quad , \end{aligned}$$

stores a sub-block of data with the size of $n_x \times n_y \times n_z$, which is defined as

$$\begin{aligned} f_{i,j,k} &\triangleq f\left((i-1)n_x : in_x,\ (j-1)n_y : jn_y,\ (k-1)n_z : kn_z\right) \qquad (15) \\ i &= b \cdot p_x + a, \\ j &= d \cdot p_y + c, \\ k &= f \cdot p_z + e, \end{aligned}$$

where $n_\alpha = \lceil N_\alpha / P_\alpha \rceil$ with $\alpha = x, y, z$. In the implementation, there are some constraints on the sizes of the data block:

$$\begin{aligned} n_x \cdot n_y &= \alpha \times P_z, \\ n_x \cdot n_z &= \beta \times P_y, \\ n_y \cdot n_z &= \gamma \times P_x, \end{aligned}$$

where α, β and γ are integers.

As mentioned in the previous subsection, the 3D FFT is computed dimension by

dimension. First of all, along the z axis, we compute $N_x \times N_y$ independent 1D FFTs of size N_z each. Before the actual 1D transform, we rearrange the scattered data along the z axis so that each of the $N_x \times N_y$ 1D FFTs is stored locally. To do this, an all-to-all personalized broadcast along the z axis must be performed. Serial FFT computations are carried out using any sequential FFT package, e.g. FFTW [4] in each node independently, followed by the same communication as above to transfer the data back to the original distribution described by Equation 15. Thus, we have completed the FFT computation along one dimension and obtained the intermediate function $\hat{f}$. Next, along the y axis, $N_z \times N_x$ independent 1D FFTs with the size of N_y each are performed to obtain $\check{f}$. Finally, the similar operations can be performed along x axis to obtain $\bar{f}$. All of these consist of the same three steps—forward transpose (FT), sequential computation (SC), and backward transpose (BT), among which communication is needed for the transposes.

In the implementation, the most straightforward scheme is to link the three steps as shown in Part A of Figure 5, resulting in utilization of communication channels in only two out of six dimensions concurrently. There are at least two improvements that can be made to increase the efficiency.

Since the forward transpose along one axis and the backward transpose along the next axis are applied within different 2D torus plane, we can combine them to reduce the communication as shown in Part B of Figure 5. That is, the backward transpose along the z logical axis can be done on the ab torus plane simultaneously with, and independently upon, its succeeding forward transpose along the y logical axis done on the cd torus plane, exploiting the independent bidirectional communication channels in four dimensions concurrently. The same manipulation is applied to the backward transpose along the y axis and its succeeding forward transpose along the x axis. In

this way, instead of six, we actually count only four times of communication, which has already been taken into consideration in the model subscribed in the next subsection.

Furthermore, within each one of the three dimensional FFT computations, instead of doing all local FFTs sequentially with the communication channels idle, followed by an intensive communication with all CPUs idle, we chop the local data into pieces by means of giving another input parameter, indicating how big the chop is. By doing this, the sequential 1D FFT computation on one piece is performed, while the next piece is being transferred simultaneously. This communication-computation overlapping, shown in Part C of Figure 5, takes the advantage of the unblocking communication scheme of QMP[3]. Some remarks here are necessary. First, the parameter we choose to indicate the size of the chopped data is dependent upon the hardware, including the speed of the CPU, the communication latency and the channel bandwidth. In this dissertation, all the efficiencies of both 3D complex-to-complex FFT and spherical-cutoff real-to-complex FFT shown in the following sections, are obtained on QCDOC by choosing optimal parameters empirically. Second, in Part C of Figure 5, we only depict the communication part, which is the bottleneck in parallel 3D FFT calculation, with shaded areas representing the time mostly spent on unavoidable memory access and a little on sequential 1D FFT computation which has not been overlapped. Most of the sequential computation time (shaded area) appearing in Part A and B has been overlapped with communication.

4.3 Parallel Performance Analysis

In this subsection, both the theoretical model and the actual parallel performance are presented, in conjunction with the analysis of their deviation. At last, the simple

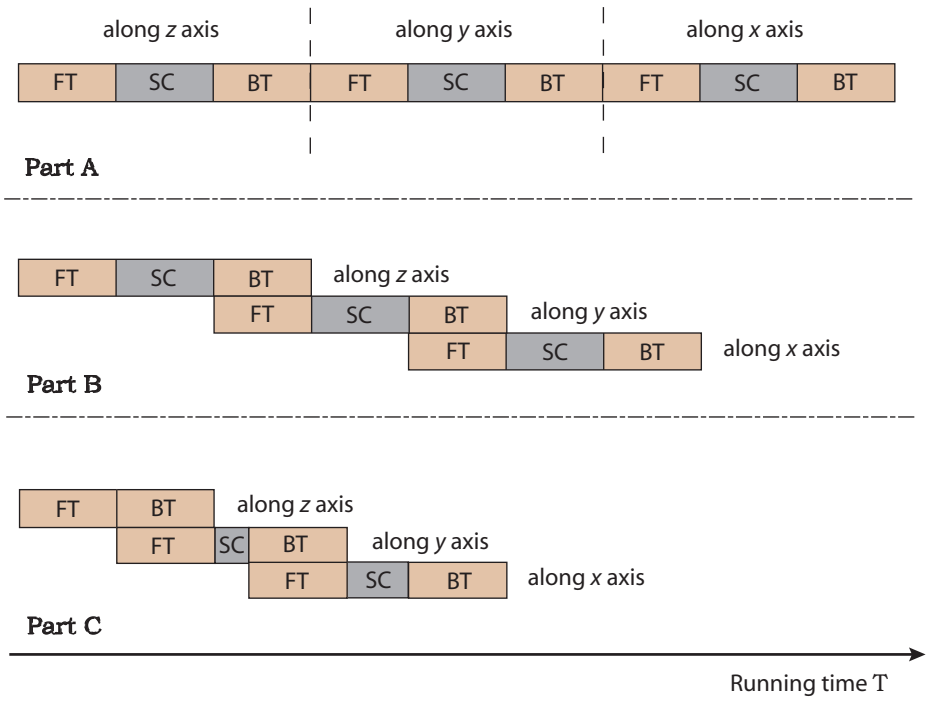


Figure 5: 3D FFT communication and computation scheme

This image shows different steps in 3D FFT calculation. Both of the communicational steps, forward transpose (FT) and backward transpose (BT), are shown in yellow; the sequential computation part(SC) is shaded.

comparison of the scalability with Volumetric 3D FFT on BlueGene/L will be shown.

4.3.1 The Model

QCDOC off-node network's store-and-forward routing determines our communicational model:

$$t_{\text{comm}} = t_{\text{s}} + (mt_{\text{w}} + t_{\text{h}})l,$$

where $t_{\rm s}$ is the start-up time, $t_{\rm w}$ the per-word transferring time, $t_{\rm h}$ the per-hop time, m the number of words transferred and l the number of hops [50]. The fact that QMP supports non-blocking communication allows us the convenience of building communication handlers before actual data transfer, justifying negligence of start-up time. Moreover, because the off-node memory-to-memory transfer time (latency) between nearest neighbors is around $0.6\mu s$, the total per-hop time in the FFT implementation is so small that it also can be ignored. Additionally, we may assume the actual link transferring speed is approximately 90% of its theoretical peak value. With these assumptions, we simplify the communication model as $t_{\rm comm} = mt_{\rm w} \times l/0.9$.

Because parallel 3D FFT is communication-intensive, we only take into consideration the communicational time in the model. However, for the computational part, we may adopt the idealized bound given by IBM [27]: although the theoretical computational order of 1D FFT on a problem of size N is $5N \log_2 N$, data dependencies force a fused multiply-add (FMA) machine to use eight cycles when a fused multiply-add is issued every cycle, resulting in a more accurate bound of $8N \log_2 N$ clock cycles.

4.3.2 Evaluation of Efficiency and Performance Analysis

We have benchmarked our parallel FFT algorithm on three QCDOC prototypes. The first, with fewer than 16 nodes running at 360MHz, resides at Columbia University; the second, with up to 64 nodes running at 420MHz, resides at Brookhaven National Laboratory (BNL); the third, up to 4096 nodes running at 400MHz, also resides at BNL. Among all three prototypes, the 4-node and 8-node systems are configured as two-dimensional tori, while the 16-node and 32-node systems are configured as four-dimensional tori. All others are configured as truncated six-dimensional tori.

Our implementation is based on QMP message-passing interface and FFTW 3.1 library as sequential 1D FFT solver, embedded into the parallel framework. The efficiencies are measured in terms of CPU time, collected from the experiments of varying node numbers, with fixed problem sizes—strong scaling experiments.

Figure 6 shows the speedup of three complex-to-complex 3D FFTs for different problem sizes: 32^3, 64^3 and 128^3.

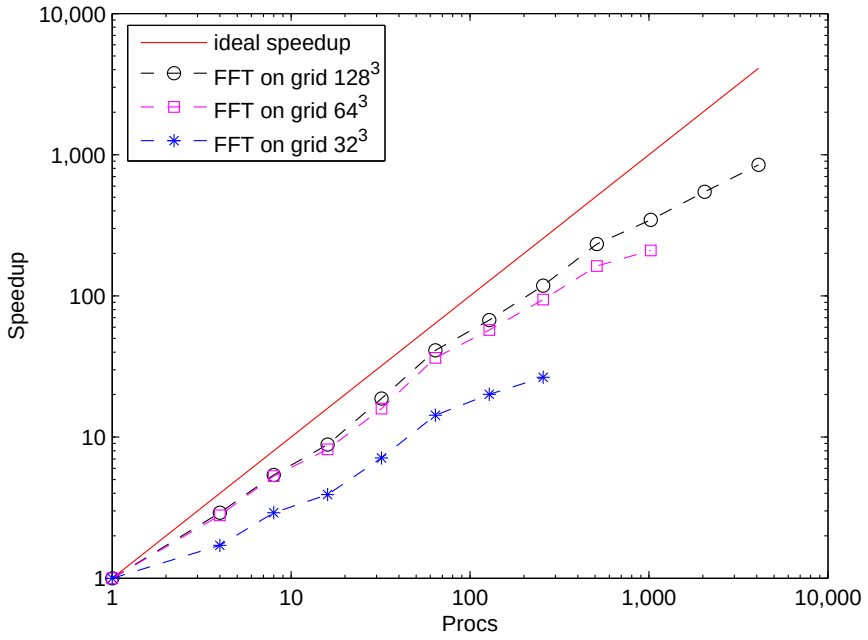


Figure 6: Speedup of complex-to-complex 3D FFT on QCDOC

This image represents speedup of three complex-to-complex 3D FFTs run on 4096-node QCDOC machine. The FFT grid sizes are set to be 32^3, 64^3 and 128^3, respectively. On 4096 node machine, the speedup can obtain 895x for 128^3 3D FFT.

Figure 7 shows the actual CPU time compared with the model. In the 32^3 case, all original data can fit into the fast EDRAM, the sequential performance on one node is much better than 64^3 and 128^3 cases, for which the data have to be stored in slower DDR memory. So we notice that the deviation of the experimental performance from

the model, for the 32^3 case, is smaller than the other two cases. The performance for larger systems deviates from the model due to memory hierarchy effect, i.e., data prefetching from the two types of memory at different speeds. Also, the store-and-forward routing makes intra-memory data transferring inevitable. The existence of the non-overlapped parts, e.g. memory access and sequential computation, as shown in Part C of Figure 5, causes additional deviation from the model. Table 1 lists the time attributed to each of these two parts and the modeled communicational time for a complex-to-complex 3D FFT problem with the size of 128^3, on 64 and 512-node QCDOC platforms.

Table 1: The deviation of the 3D FFT Parallel Performance from the Model

The modeled communicational time, the measured non-overlapped sequential FFT computational time and memory hierarchy time are presented, for 3D FFT with the size of 128^3 run on 64 and 512-node QCDOC platforms, in addition to their actual CPU times. Δ, is the ratio of the hardware and software overheads (the difference between the actual CPU time and the accumulation of the other three terms) to the actual CPU time. All the numbers are in the unit of 10^{-3} seconds.

	T_{model}	$T_{non-overlapped\,FFT}$	$T_{memory\,hierachy}$	$T_{actual\,performance}$	Δ
64 nodes	41.6	7.2	15.9	69.0	4.3/69.0
512 nodes	6.4	0.6	2.2	12.2	3.0/12.2

For asymptotic analysis of scalability, the bandwidth-only model is inadequate because of the increasing hardware and software overheads for larger systems. For example, with more than 512 nodes, the software and hardware overheads in communication become significant and prevent the running-performance from increasing, as evident in Table 1. On the other hand, for 64-node system, the CPU time for the non-overlapped memory access and sequential FFT computation is more significant in the actual performance than it is on the 512-node system. The software and hardware overheads for the 512-node system can no longer be neglected. Even with the

increased overhead, our algorithm can scale well up to 4096 nodes for FFT grid size 128^3, as shown in Figure 7.

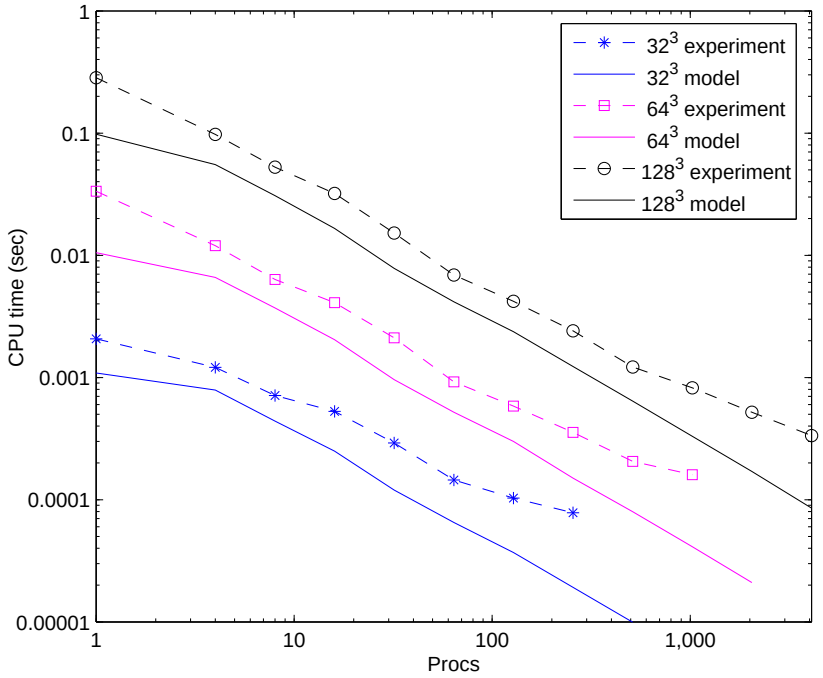


Figure 7: Parallel Performance of complex-to-complex 3D FFT on QCDOC

The actual performance vs. theoretical model, measured in terms of the actual CPU time, is shown here. Three sets of 3D-FFT problems are tested: 32^3, 64^3 and 128^3.

4.3.3 Comparison with Volumetric 3D FFT on BlueGene/L

Eleftheriou et al. [27] presented the volumetric complex-to-complex 3D FFT method and its performance on BlueGene/L [6]. Here, we compare the performance obtained on QCDOC with that on BlueGene/L. The performance on QCDOC is obtained from BNL's 512-node system running at 400MHz, while the performance on BlueGene/L is quoted from [27].

Table 2 lists hardware differences between QCDOC and BlueGene/L, which are essentially relative to the performance of parallel 3D-FFT calculation. Figure 8 represents the comparison of the actual performances of our scheme on QCDOC, and Eleftheriou et al.'s on BlueGene/L, in solving 3D FFTs of size 32^3, 64^3 and 128^3. The efficiency deviation is within the factor of hardware difference, according to Table 2. For bigger machines of more than 1,000 nodes, the better scalability of our scheme on QCDOC is apparent.

Table 2: Hardware differences between QCDOC and BlueGene/L

This table lists some hardware parameters that differ in QCDOC and in BlueGene/L. Unless the differences are taken into consideration, the performance comparison of the parallel scheme does not make any sense at all.

	QCDOC	Blue Gene/L
Dimensionality	6-dimensional torus $p_x \times 2 \times p_y \times 2 \times p_z \times 2$	3-dimensional torus $b_x \times b_y \times b_z$
Processor Clock Speed	400MHz	700MHz
Bandwidth per Link	1 bit/clockcycle	2 bits/clockcycle
Message Passing Interface	QMP	MPI, Active packet
Communication routing	Store-and-Forward routing	Cut-Through routing

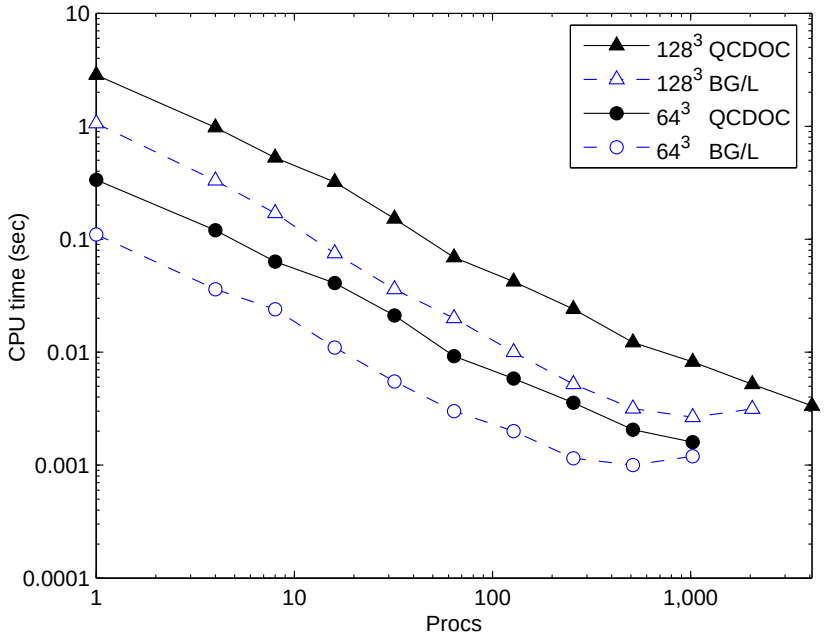


Figure 8: The comparison of Parallel 3D FFT Performance: QCDOC vs. BlueGene/L

The performance comparison between our scheme implemented on QCDOC and Eleftheriou et al.'s Volumetric 3D FFT on BlueGene/L, in solving complex-to-comple 3D FFTs with the sizes of 32^3, 64^3 and 128^3.

5 Parallel SPME

At the beginning of this section, let us review the general steps of SPME. According to Equation 14 given in Section 2, SPME contains five steps:

(1°) Charge assignment. Calculate the mesh based charge density using Euler Exponential Spline function and obtain $n^{(Euler)}(\overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}, n, L_{\mathrm{FFT}})$

(2°) Calculate the finite Fourier transform $\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, n, L_{\mathrm{FFT}})$ of the mesh based charge density.

(3°) Multiply $\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, n, L_{\mathrm{FFT}})$ with precomputed kernel function $W(\mathbf{g})$, then the potential energy could be obtained according to Equation 14.

(4°) Apply an inverse finite Fourier transform to this product to end up with the finite convolution of mesh based charge density function.

(5°) Force interpolation: The analytic differentiation of the potential energy is made to get long-range forces. Back interpolation is needed.

Here in SPME, a complex-to-complex 3D FFT package is neither suitable nor efficient, since all the interpolated charges are real numbers. Moreover, as we mentioned in Section. 2, summing up the higher order Fourier coefficients is useless and inefficient according to the interpolated error which scales as $[2\hat{\mathbf{g}}/Ł_{\mathrm{FFT}}]^n$. However, relying on the basic implementation ideas and parallelism techniques we introduced in the previous section, we expanded the functionality of our parallel framework in two aspects; one is to support spherical cutoff real-to-complex 3D FFT calculation, the other is to loose the size constraint so that it can support any size of FFT grid,

retaining the load balancing. With this package expansion, we paved the way for the implementation of parallel SPME.

In this section, the way the spherical cutoff in Fourier transform will be presented, followed by the decomposition scheme in SPME and the details of the interpolation process.

5.1 Spherical Cutoff 3D FFT

In SPME, because of the non-linearly scaling interpolation error, only the non-zero Fourier-coefficient sphere, with the radius $\hat{g}_c$ in a cube of edge L_{FFT}, is of our interest, where $L_{\mathrm{FFT}} > 2\hat{g}_c$. This gives us an opportunity to reduce floating point operations in sequential platforms and the amount of data to broadcast in parallel platforms. As shown in Figure 9, the first set of 1D-FFTs, $f(x, y, z) \rightarrow \hat{f}(g_x, y, z)$ (same notation as Section. 4.1), requires $L^3_{\mathrm{FFT}} \log L_{\mathrm{FFT}}$ Flops, but the second set, $\hat{f}(g_x, y, z) \rightarrow \check{f}(g_x, g_y, z)$, requires only $2\hat{g}_c L^2_{\mathrm{FFT}} \log L_{\mathrm{FFT}}$ Flops; while the third set, $\check{f}(g_x, g_y, z) \rightarrow \bar{f}(g_x, g_y, g_z)$, is further reduced to $\pi \hat{g}_c^2 L_{\mathrm{FFT}} \log L_{\mathrm{FFT}}$. Figure 9 and the above description is for complex-to-complex 3D FFT. In the case of real-to-complex 3D FFT, a half more could be reduced, which will be shown shortly. Pseudocode is also provided here with an explanation of real-to-complex 3D-FFTs.

On the QCDOC parallel platform, we still exploited the volume decomposition scheme, presented by Figure 10. The blue arrows in Part A show the data blocks originally assigned to the processors. We can formulate the decomposition using the same expression as Equation 15, except that all the elements are real numbers (mesh-based charge density), instead of complex numbers. To calculate the first set along z axis, i.e. $f(x, y, z) \rightarrow \hat{f}(x, y, g_z)$, all-to-all personalized broadcasts must be done

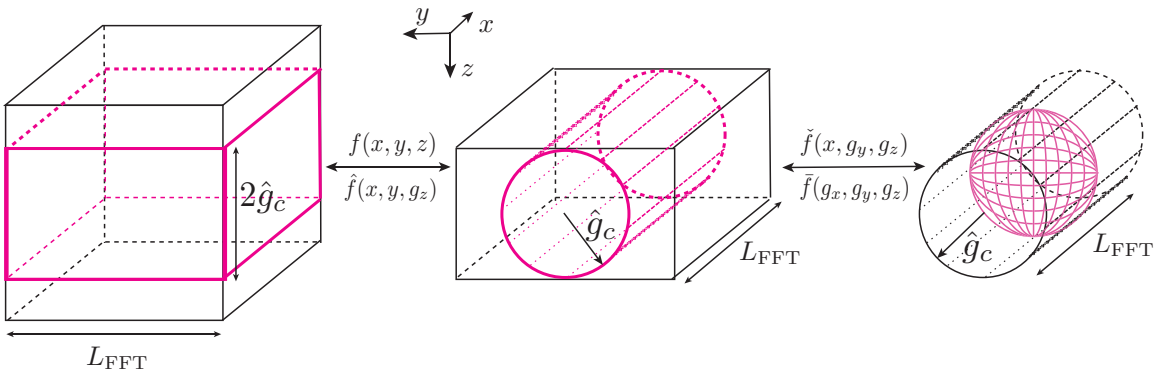


Figure 9: Spherical Cutoff complex-to-complex 3D FFT.

With spherical cutoff $\hat{g}_c$ and FFT grid size L^3_{FFT}, complex-to-complex 3D FFT calculation procedure has been shown in this figure. The cutoffs are presented in pink.

among the processors along the same vertical line on Figure 10, i.e. having the same coordinates on dimension c, d, e, f. The amount of data being transferred is L^3_{FFT} (real numbers). After the first set of 1D-FFTs, a cutoff can be made in the z dimension, which is shown in pink in the figure. The reason why the spherical cutoff is $\hat{g}_c$ rather than $2\hat{g}_c$ is that the conjugate-symmetric property of the real-to-complex FFT can be employed. In order to retain its load balancing after the cutoff, data within the pink cuboid, with the total data amount being equal to $\hat{g}_c L^2_{\mathrm{FFT}}$ (complex numbers), has to be re-distributed to each processor during back transpose, as shown by brown arrows in Part A. Herein, the re-distribution cannot be absolutely even, however, we made it as even as possible in such way that the maximum difference along z axis is no bigger than one. To calculate the second set along y axis, i.e. $\hat{f}(x,y,g_z) \rightarrow \check{f}(x,g_y,g_z)$, the broadcasts are needed to be done among the processors having the same coordinates on dimension a, b, e, f. After the second set of calculations, the cutoff could be made in the y dimension as shown in pink in Part B Figure 10. Right now the rest is half a cylinder. We re-distributed again along y axis as shown by brown arrows in Part B,

Pseudocode 1 Real-to-complex 3D FFT with a spherical cutoff.

We assume for simplicity that the cell is a cubic. Note, $g_x \geq 0$ but g_y and g_z take on positive and negative values.

for all y **do**
  for all x **do**
    Real to complex 1D-FFT $f(x, y, z) \longrightarrow \hat{f}(x, y, g_z); g_z \geq 0$
    Unpack g_z in range $0 \leq g_z \leq g_c$
  end for
end for

for all x **do**
  for all g_z such that $0 \leq g_z \leq g_c$ **do**
    Complex-to-Complex 1D-FFT $\hat{f}(x, y, g_z) \longrightarrow \check{f}(x, g_y, g_z)$
    Unpack $\pm g_y$ in range $0 \leq [g_y^2 + g_z^2]^{1/2} \leq g_c$
  end for
end for

for all g_z such that $0 \leq g_z \leq g_c$ **do**
  for all $\pm g_y$ such that $0 \leq [g_y^2 + g_z^2]^{1/2} \leq g_c$ **do**
    Complex-to-Complex 1D-FFT $\check{f}(x, g_y, g_z) \longrightarrow \bar{f}(g_x, g_y, g_z)$
    Unpack $\pm g_x$ in range $0 \leq [g_x^2 + g_y^2 + g_z^2]^{1/2} \leq g_c$
  end for
end for

with the total data amount being equal to $\frac{1}{2}\pi \hat{g}_c^2 L_{\text{FFT}}$. To calculate the third set of 1D-FFTs, i.e. $\check{f}(x, g_y, g_z) \rightarrow \bar{f}(g_x, g_y, g_z)$, the broadcasts are needed to be done among the processors having the same coordinates on dimension a, b, c, d. After the third set of calculations, the final cutoff is made, which is half a ball in the half cylinder. There is no need to do re-distribution any more, since after the multiplication with $W(\mathbf{g})$, inverse 3D FFT are going to be processed. Hence, we save twice the amount of data being communicated. The inverse 3D FFT has similar procedure as the forward 3D FFT. The only difference is that instead of cutting the edges off, inverse 3D FFT needs to concatenate some zero-value edges to recover the original cube.

Similar implementation techniques, including full utilization of different channels

and computation-communication overlap, which we mentioned in Section. 4.2, have been used in SPME. These reduce the total data amount (in the unit of complex number) being transferred to be equal to

$$2\left(\frac{1}{2}L_{\mathrm{FFT}}^3+\hat{g}_c L_{\mathrm{FFT}}^2+\frac{1}{2}\pi\hat{g}_c^2 L_{\mathrm{FFT}}\right),$$

where the first constant factor 2 indicates the similar communication process for both forward 3D FFT and inverse 3D FFT. Compared to the original 3D FFT without a spherical cutoff, the corresponding amount of data is reduced by a factor of $3/(1+\bar{L}_{\mathrm{FFT}}+0.25\pi\bar{L}_{\mathrm{FFT}}^2)$ where $\bar{L}_{\mathrm{FFT}}=2\hat{g}_c/L_{\mathrm{FFT}}<1$.

Note, although Figure 10 pictured QCDOC as a 3-dimensional torus, we should keep in mind that it is actually a 6-dimensional torus, which means all communications along one dimensional rings in Figure 10 are actually within two dimensional tori.

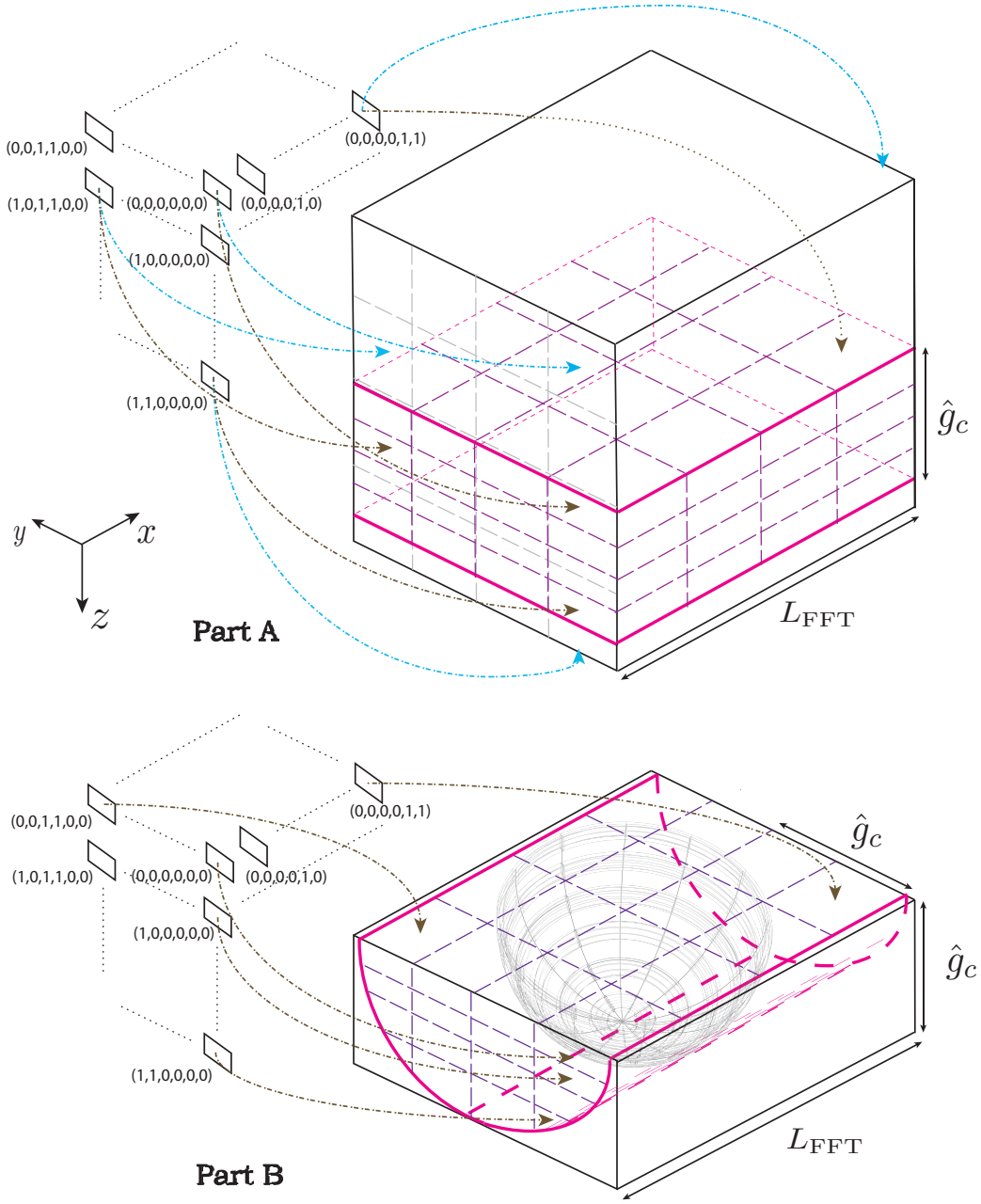


Figure 10: The parallel implementation of real-to-complex 3D FFT on QCDOC

This figure presents the procedure to solve spherical cutoff real-to-complex 3D FFT, with the cutoff radius $\hat{g}_c$ and the FFT grid size L_{FFT}. The parallel platform is a 64 node QCDOC machine, wired as a $2 \times 2 \times 2 \times 2 \times 2 \times 2$ 6D torus, although it has been pictured as a 3D torus. Some of the coordinates of the processors have been shown in the figure. The dotted-line arrows show the data decomposition and re-distribution after each time of cutoff.

5.2 Charge Assignment and Force Interpolation

In this subsection, the basic interpolation algorithm of charge assignment and force interpolation is presented, followed by its implementation in our parallel SPME package on QCDOC.

5.2.1 Basic Algorithm

To discretize the continuous charge density function and hence to approximate reciprocal potential energies $\phi_{\mathrm{rec}}^{(\mathrm{Coul})}$, the interpolation phrases employ Euler exponential spline:

$$\begin{aligned} \exp\left(\frac{2\pi i\hat{g}p}{L}\right) &= d_n(\hat{g},L)\sum_{\hat{s}=-\infty}^{\infty} M_n(p-\hat{s})\exp\left(\frac{2\pi i\hat{g}\hat{s}}{L}\right)+\mathcal{O}\left(\frac{2|\hat{g}|}{L}\right)^n \qquad (16)\\ d_n(\hat{g},L) &= \frac{\exp\left(2\pi i(n-1)/L\right)}{\left[\sum_{j=0}^{n-2} M_n(j+1)\exp\left(2\pi i\hat{g}j/L\right)\right]} \end{aligned}$$

where $\hat{s}$ is an integer, p is a real number, n is the spline order assumed to be even and the $M_n(p)$ are the Cardinal B-splines, with the properties described below.

For any real number p, let $M_2(p)$ denote the linear hat function given by $M_2(p) = 1-|p-1|$ for $0 \leqslant p \leqslant 2$ and $M_2(p) = 0$ for $p < 0$ or $p > 2$. For n greater than 2, define $M_n(p)$ by the recursion

$$M_n(p) = \left[\frac{p}{n-1}\right] M_{n-1}(p) + \left[\frac{n-p}{n-1}\right] M_{n-1}(p-1) \qquad (17)$$

One of the many convenient properties of Cardinal B-splines is that they can be easily

differentiated analytically. For $n > 2$

$$\frac{dM_n(p)}{dp} = M_{n-1}(p) - M_{n-1}(p-1) \tag{18}$$

From Equation 17 one can see that $M_n(p)$ has finite support, being zero outside the interval $0 \leqslant p \leqslant n$, which means the interpolation can be calculated locally. From Equation 18 it is clear that $M_n(p)$ is $n-2$ times continuously differentiable, which turns possible to obtain reciprocal forces and stress tensors through the analytical differentiation of the approximated reciprocal energy. More details are given in Appendix C.

In our implementation, to avoid redundant computation, given the interpolation order n, the explicit form of Cardinal B-splines, $M_n(p)$, is pre-calculated, rather than using the recursion-form definition, Equation. 17. Since the Cardinal B-spline, $M_n(p)$, could be represented as the summation of a basis of a polynomial space π_n, i.e.

$$\rho_l^n(x) := \binom{n}{l}(1-x)^{n-l}x^l, \quad 0 \leqslant k \leqslant n,$$

the following algorithm can provide us an explicit function form of $M_n(p)$ [19].

Algorithm 2 Cardinal B-spline B-net algorithm

Let $n \geqslant 2$ be any integer and set

$$M_n|_{\,[k-1,k)}(p) = \sum_{l=0}^{n-1} a_l^{n-1}(k)\rho_l^{n-1}(p-k+1) \ ,$$

$k = 1, ..., n$. Also set

$$a_j^{n-1}(0) = 0 \text{ and } a_j^{n-1}(n+1) = 0, \quad j = 0, ..., n-1 \ , \tag{19}$$

and consider the initial conditions

$$a_0^1(1) = 0, \ a_1^1(1) = 1 = a_0^1(2), \ a_1^1(2) = 0 \ . \tag{20}$$

Compute $b_j^{n-1}(k)$ and $a_l^n(k)$ below by using Equation. 19 and 20 for $n = 2$, and then repeat the same process by using Equation. 19 and the previous result for $n = 3, 4, ...$; where

$$b_j^{n-1}(k) := a_j^{n-1}(k) - a_j^{n-1}(k-1),$$

for $j = 0, ..., n-1$ and $k = 1, ..., n+1$;

$$a_l^n(k) = a_n^n(k-1) + \frac{1}{n}\sum_{j=0}^{l-1} b_j^{n-1}(k),$$

for $l = 0, ..., n$, and $k = 1, ..., n+1$.

Figure. 11 illustrates the scheme of the assignment of the charges, sitting anywhere in real space, to the nodes of a uniform grid (under periodic boundary condition). The force interpolation is the same but inverse process of charge assignment. Note, unlike Lagrangian polynomial interpolation, this scheme is not symmetric, because of the uneven property of M_n. However this *shift* has no effect on the numerical results, because it is undone in the force interpolation (back-interpolation).

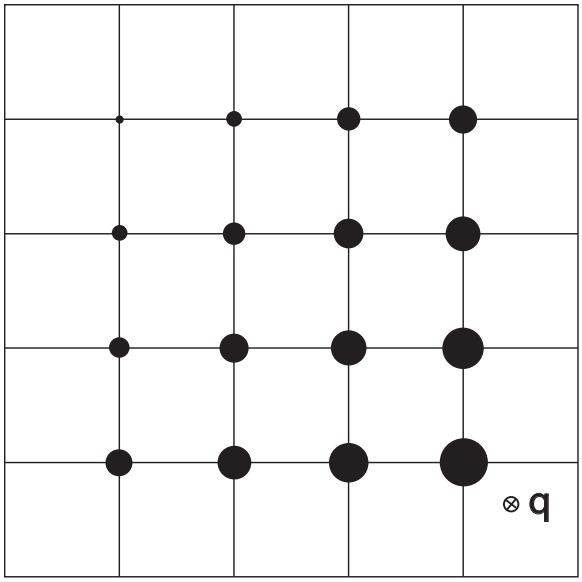


Figure 11: 2D demonstration of charge assignment/force interpolation.

This figure shows two dimensional charge assignment scheme, with Euler Exponential Spline function, Equation 16, being used. The force interpolation is similar but inverse process. Note, periodic boundary condition is employed here.

5.2.2 Parallel Implementation

Herein, the way how we implement charge assignment and force interpolation on QCDOC will be introduced. Briefly speaking, we adopt logical volume decomposition for charged particles. Figure 12 demonstrates this idea and how we meshed the charge density. We assume that a system of interest is solvated in a simulation box with the size of $40 \times 40\mathring{A}^2$ and a 16-processor QCDOC partition is employed, mapped as $2 \times 2 \times 2 \times 2 \times 1 \times 1$. L_{FFT} and n are set to be 18×18 and 4, respectively. For FFT grid, we volume-decompose them as evenly as possible, as we mentioned before; for instance, processor 6 is assigned 5×5 lattice points, which are the small

black dots in the light green area (labeled Volume 1). For the real charged particles, we actually keep a copy of every particle in each of the processors, in such a way that all particles residing in a "Bin" bounded in red, are stored in one hash data structure, which can provide quick search ability. In this figure, "Bin" size has been fixed at $5 \times 5\mathring{A}^2$. Every time step, the lists within each processor will be updated when integration is being done. (The optimized hardware-embedded global broadcast design on QCDOC [10, 3] makes it practical.) Under this decomposition scheme, since the lattice points are equally distant, 3D FFT calculation could be considered evenly. The CPU time comparison shown in the next section verifies it. However, load imbalance is unavoidable in the processes of charge assignment and force interpolation, which we will explain shortly.

In charge assignment and force interpolation processes, two volumes are predetermined, marginal volume (volume 2 in light yellow) and volume 3 in light pink, which are shown in Figure 12. Marginal volume is the volume that contains all the charged particles having partial/full influence, in respect of interpolation, on the FFT grid points stored in processor 6. Volume 3 is the union of "Bins" that contain all of these particles. So instead of checking the particles in the whole system, processor 6 only needs to check the coordinates of the particles within several "Bins" (covered in light pink); moreover, only the particles within volume 2 need to do interpolation calculation. Note depending on the position of the particles, not all of these interpolations need n^3 Flops calculation.

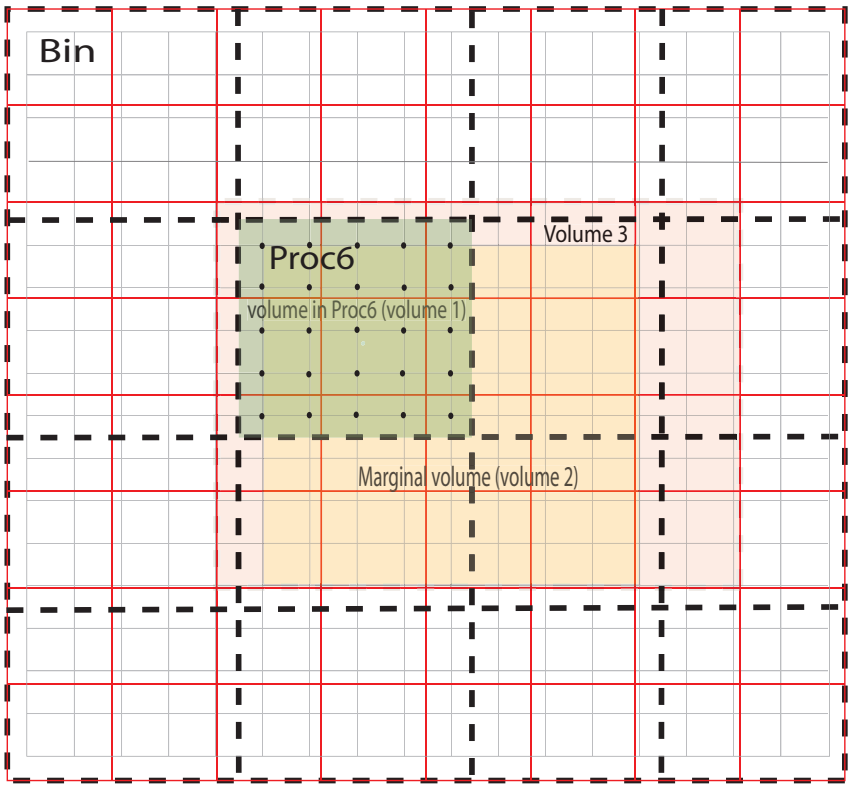


Figure 12: 2D demonstration of decomposition scheme in SPME on QCDOC.

Two dimensional simulation box $40 \times 40\mathring{A}^2$ was volume decomposed into 16 processors. Each processor stores FFT grid points belonging to its own assigned area, e.g. light green area (volume 1) for processor 6, and a set of copies of all the particles, ordered in unit of "Bins" (hashes physically). Marginal volume (volume 2 in light yellow) and volume 3 (in light pink) are also shown.

With this implementation, we instinctively expect two things to happen. First of all, the difference of the density in different domains will definitely cause load imbalance. How severe the load imbalance is depends on the system under investigation. From the two systems we present in this document, load imbalance usually becomes more severe when more processors being used. This makes sense–if a domain closes to the center of a dense area, the smaller this domain is, the more dense it would

be; on the other hand, if a domain is away from the center, the smaller this domain is, the less dense it would be. Secondly, it is easy to see that given a fixed B-spline order, the bigger the number of grid point is, the smaller the processor's marginal volume is, and the less the computation needs; on the other hand, given a fixed size of grid-point lattice, the bigger the B-spline order is, the larger the marginal volume is, and the more the computation needs. The experiments presented in the next section will verify these suppositions.

6 Evaluation of Accuracy and Efficiency of Ewald/SPME treatment of the Intermolecular Forces

In section 2, a set of methods designed to evaluate intermolecular forces have been presented. The reciprocal space part of the Ewald sum was assumed to be evaluated either directly (MDoC support of the direct Ewald summation [69]) or via Smooth Particle Mesh Ewald method on a spherically cutoff reciprocal space, whose implementation details have been presented in the previous sections. In this section, the accuracy (truncation error) and efficiency (cpu time) of these techniques is tested on two protein/water systems described below. The parallel platform is the QCDOC supercomputer.

6.1 Model Systems: Preparation

In order to test the accuracy and the efficiency of the parallel SPME methodology described in this dissertation, two systems were examined. One is the first β-hairpin in streptococcal protein G (PDB identification label,"1PGB"), 20 residues, solvated by 1112 waters, with the total number of atoms being 3579. The other is HIV-1 Protease (PDB identification label, 1G6L), 198 residues, solvated by 8793 waters, with the total number of atoms being 29508. Initial structures were first minimized using AMBER 7 followed by MD runs. The simulation boxes are $V = 40.2\times31.8\times42.2\mathring{A}^3$ for β-hairpin and $V = 69.5\times64.7\times83.3\mathring{A}^3$ for HIV-1 respectively.

In detail, for both of the systems, a real space cutoff of $r_c = 10\mathring{A}$ was employed and

$5 \times 5 \times 5\mathring{A}^3$ "Bin" size is fixed. ("Bin"s closer to the boundaries maybe smaller than $5 \times 5 \times 5\mathring{A}^3$.) The Ewald convergence parameter was taken to be $\alpha_{ewd} = 0.346\mathring{A}^{-1}$ which yields $\text{erfc}(\alpha_{ewd} r_c) \approx 10^{-6}$. In reciprocal space the cutoff, $\hat{g}_c = 1.18\alpha_{ewd} L_{\max} = 18$ (in Equation 12, let $\exp(-\mathbf{g}^2/4\alpha_{ewd}^2) \approx 10^{-6}$, with $\mathbf{g} = 2\pi \hat{\mathbf{g}} \overleftrightarrow{\mathbf{h}}^{-1}$), was employed in the β-hairpin system while the cutoff, $\hat{g}_c = 32$, was employed in the HIV-1 system.

6.2 Accuracy Analysis

The accuracy of the reciprocal space part of the standard Ewald energy evaluated by direct summation is given as a function of reciprocal space cutoff in Table 3 and Table 4 for the case, $\alpha_{ewd} r_c = 3.46$. Clearly, if the reciprocal space cutoff are in the range, $\hat{g}_c \geq \alpha_{ewd} L = 18$ for β-hairpin, $\hat{g}_c \geq \alpha_{ewd} L = 32$ for HIV-1, then both the energy and forces are determined to high accuracy. For instance, for β-hairpin, 10 cal/mol divided over 3,500 atoms is a rather small error in the energy; and so is 123 cal/mol divided over 30,000 atoms for HIV-1.

Table 3: Accuracy of Standard Ewald summation. **System β-hairpin**

Evaluation of the accuracy of standard Ewald summation as a function of maximum reciprocal lattice vector. A calculation using $\hat{g}_c = 30$ is used to define the "exact answer" (E=2.7269e+5 cal/mol). The quantities, $\Delta E = E(\hat{g}_c) - E(30)$ and $F_{rms} = [\sum_k |\mathbf{F}_k(\hat{g}_c) - \mathbf{F}_k(30)|^2 / F_{mag}^{(tot)}]^{1/2}$, are used to assess the error. Here, $F_{mag}^{(tot)} = \sum_k |\mathbf{F}_k^{(tot)}|^2$ where $\mathbf{F}_k^{(tot)}$ is the total intermolecular force on the k^{th} particle, not just the reciprocal space part.

$\hat{g}_c$	ΔE(cal/mol)	F_{rms}
12	-4439	2.50e-03
15	-328	1.27e-04
18	-10	2.75e-06
20	0	1.15e-07

Table 4: Accuracy of Standard Ewald summation. **System HIV-1**

Evaluation of the accuracy of standard Ewald summation as a function of maximum reciprocal lattice vector. A calculation using $\hat{g}_c = 48$ is used to define the "exact answer" (E=3.9639e+6 cal/mol). The other quantities mean the same as Table 3.

$\hat{g}_c$	ΔE(cal/mol)	F_{rms}
18	-226492	8.00e-03
24	-28513	6.81e-04
28	-4447	8.54e-05
30	-1421	2.43e-05
34	-123	1.76e-06
38	-6	7.84e-07
40	-2	1.30e-08

The accuracy of SPME is given as a function of grid points, L_{FFT} and Euler exponential spline interpolation order, n in Table 5 and Table 6 for β-hairpin system, with $\hat{g}_c$ being fixed at 18 and 20 respectively; Table 7, Table 8 and Table 9 for HIV-1 system, with $\hat{g}_c$ being fixed at 30, 34 and 38 respectively. (The error ϵ is set to be greater than 10^{-6}, when $\hat{g}_c = 30$.) Acceptable levels of accuracy (within twice the energy difference between the approximated Ewald summation and the exact Ewald summation)are indicated with a star (*). SPME accuracy is ultimately limited by the reciprocal space cutoff, $\hat{g}_c$, given a sufficiently high interpolation level and/or a large number of grid points (small grid spacing). SPME should be optimized to produce errors of the same order as the Ewald summation at the same finite $\hat{g}_c$. In addition, $\hat{g}_c$ should **not** be used to determine the grid spacing via $L_{\mathrm{FFT}} \equiv 2\hat{g}_c$ as discussed above and in Appendix C. It is simply a parameter to optimize at fixed α_{ewd} and $\hat{g}_c$.

Theoretically, given $\hat{g}_c$, there are two ways to decrease the approximation error, either by increasing L_{FFT} with fixed n, or by increasing n with fixed L_{FFT}. For instance, in β-hairpin simulation with $\hat{g}_c = 18$, the parameter sets $\{L_{\mathrm{FFT}} = 56, n = 6\}$ and $\{L_{\mathrm{FFT}} = 128, n = 4\}$ produce similar errors; in the HIV-1 simulation with $\hat{g}_c = 30$, the parameter sets, $\{L_{\mathrm{FFT}} = 64, n = 8\}$, $\{L_{\mathrm{FFT}} = 72, n = 6\}$ and $\{L_{\mathrm{FFT}} = 128, n = 4\}$, all generate similar errors. However, the use of $n = 4$ is rarely a reasonable choice in practice because it yields low computational efficiency. (We will come back to this point in the next subsection.) Moreover, the use of such a small interpolation order becomes more problematic as the reciprocal cutoff is increased. For instance, setting $\hat{g}_c = 38$ in the HIV-1 system, the desired error tolerance cannot be achieved even with very large FFT size, $L_{\mathrm{FFT}} = 256$, and furthermore the sets' low computational efficiency would obviates its use in a parallel simulation (see Table 13). $n = 6$ or 8 is suggested to be used in practical simulation in parallel simulation.

Table 5: Accuracy of SPME. **System β-hairpin, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{18}$**

Evaluation of the accuracy of SPME as a function of interpolation order and grid spacing. The reciprocal space cutoff is fixed at $\hat{g}_c = 18$. Energies and forces are reported relative to the Ewald summation at the same cutoff and relative to the "exact results" (Ewald summation at $\hat{g}_c = 30$).

L_{FFT}	n	ΔE^{ex}(c/m)	ΔE^{rel}(c/m)	F^{ex}_{rms}	F^{rel}_{rms}
40	6	-105.9	-96.0	1.45e-04	1.45e-04
40*	8	-25.8	-15.9	2.14e-05	2.09e-05
56*	6	-16.9	-7.2	1.81e-05	1.77e-05
56*	8	-10.1	-0.4	1.24e-05	1.19e-05
64*	6	-12.7	-3.0	1.35e-05	1.30e-05
64*	8	-9.9	-0.2	1.24e-05	1.19e-05
96	4	-31.2	-21.3	7.02e-05	7.00e-05
128*	4	-17.1	-7.4	2.77e-05	2.75e-05

Table 6: Accuracy of SPME. **System β-hairpin, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{20}$**

Evaluation of the accuracy of SPME as a function of interpolation order and grid spacing. The reciprocal space cutoff is fixed at $\hat{g}_c = 20$. Energies and forces are reported relative to the Ewald summation at the same cutoff and relative to the "exact results" (Ewald summation at $\hat{g}_c = 30$).

L_{FFT}	n	ΔE^{ex}(c/m)	ΔE^{rel}(c/m)	F^{ex}_{rms}	F^{rel}_{rms}
56	6	-7.8	-7.2	1.77e-05	1.26e-05
56*	8	-0.8	-0.4	1.19e-05	5.69e-07
56*	10	-0.4	-0.1	1.18e-05	3.86e-08
64	6	-3.4	-3.0	1.30e-05	5.20e-06
64*	8	-0.6	-0.2	1.19e-05	1.63e-07
72*	6	-1.8	-1.2	1.22e-05	2.65e-06
72*	8	-0.4	-0.1	1.19e-05	6.00e-08
128	4	-7.8	-7.4	2.95e-05	2.75e-05

Table 7: Accuracy of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{30}$

Evaluation of the accuracy of SPME as a function of interpolation order and grid spacing. The reciprocal space cutoff is fixed at $\hat{g}_c = 30$. Energies and forces are reported relative to the Ewald summation at the same cutoff and relative to the "exact results" (Ewald summation at $\hat{g}_c = 48$).

L_{FFT}	n	ΔE^{ex}(c/m)	ΔE^{rel}(c/m)	F^{ex}_{rms}	F^{rel}_{rms}
64	4	-22756	-21337	2.00E-03	2.00E-03
64	6	-4807	-3386	2.35E-04	2.33E-04
64*	8	-2361	-942	5.96E-05	5.32E-05
72	4	-9223	-7802	1.20E-03	1.20E-03
72*	6	-2836	-1415	9.95E-05	9.49E-05
72*	8	-1679	-260	3.01E-05	1.55E-05
80	4	-9223	-7802	7.43E-04	7.42E-04
80*	6	-1985	-566	5.15E-05	4.40E-05
80*	8	-1492	-72	2.67E-05	8.55E-06
96*	6	-1574	-153	2.92E-05	1.39E-05
128*	4	-2482	-1063	1.04E-04	9.98E-05

Table 8: Accuracy of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{34}$

Evaluation of the accuracy of SPME as a function of interpolation order and grid spacing. The reciprocal space cutoff is fixed at $\hat{g}_c = 34$. Energies and forces are reported relative to the Ewald summation at the same cutoff and relative to the "exact results" (Ewald summation at $\hat{g}_c = 48$).

L_{FFT}	n	ΔE^{ex}(c/m)	ΔE^{rel}(c/m)	F^{ex}_{rms}	F^{rel}_{rms}
72	4	-14481	-14358	1.20E-03	1.20E-03
72	6	-1659	-1536	9.59E-05	9.57E-05
72	8	-447	-326	1.64E-05	1.61E-05
80	4	-8096	-7973	7.43E-04	7.43E-04
80	6	-743	-620	4.46E-05	4.44E-05
80*	8	-215	-91	9.04E-06	8.66E-06
96	4	-3996	-3875	3.33E-04	3.33E-04
96*	6	-288	-165	1.43E-05	1.40E-05
96*	8	-135	-12	7.50E-06	7.12E-06
160	4	-556	-435	4.79E-05	4.78E-05
256*	4	-179	-56	1.26E-05	1.23E-05

Table 9: Accuracy of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{38}$

Evaluation of the accuracy of SPME as a function of interpolation order and grid spacing. The reciprocal space cutoff is fixed at $\hat{g}_c = 38$. Energies and forces are reported relative to the Ewald summation at the same cutoff and relative to the "exact results" (Ewald summation at $\hat{g}_c = 48$).

L_{FFT}	n	ΔE^{ex}(c/m)	ΔE^{rel}(c/m)	F^{ex}_{rms}	F^{rel}_{rms}
80	6	-638	-632	4.46E-05	4.45E-05
80	8	-103	-97	9.04E-06	9.03E-06
80	10	-30	-24	7.52E-06	7.50E-06
90	6	-286	-280	2.02E-05	2.02E-05
90	8	-34	-26	7.61E-06	7.60E-06
96	6	-173	-167	1.43E-05	1.43E-05
96*	8	-20	-14	7.50E-06	7.49E-06
128*	6	-28	-22	7.69E-06	7.68E-06
128*	8	-6	0	7.43E-06	7.42E-06
160	4	-441	-435	4.78E-05	4.78E-05
256	4	-62	-56	1.23E-05	1.23E-05

6.3 Efficiency Analysis

The efficiency of SPME is not only dependent on grid points and interpolation order, but also dependent on the number of processors. Table 10 and Table 11 show the parallel efficiency(CPU time) of SPME to compute long-range forces in the β-hairpin system with $\hat{g}_c = 18$ and $\hat{g}_c = 20$, respectively; Table 12, Table 13 and Table 14 show the parallel efficiency(CPU time) of SPME to compute long-range forces in the HIV-1 system with $\hat{g}_c = 30$, $\hat{g}_c = 34$ and $\hat{g}_c = 38$, respectively. Briefly, on the massively parallel system, our spherical cutoff parallel 3D FFT framework can really give an efficient and scalable solution to solve the long-range potential energy, when attention is paid in choosing a proper parameter set.

Table 10: Parallel Efficiency of SPME. **System β-hairpin, $\hat{\mathbf{g}}_\mathbf{c} = 18$**

Parallel efficiency(CPU time) of SPME as a function of interpolation order and grid points. The reciprocal space cutoff is fixed at $\hat{g}_c = 18$. The efficiency is measured by total CPU time (in seconds) for 100 steps on different QCDOC partitions, ranged from 16 node to 512 nodes, running at 400MHz. The most efficient options are shown in bold font.

L_{FFT}	n	16 nodes	64 nodes	128 nodes	256 nodes	512 nodes
40	6	9.5	3.3	2.2	1.4	1.0
40*	8	19.0	6.3	4.5	2.7	2.0
56*	6	9.0	2.9	2.0	1.3	0.9
56*	8	17.5	5.9	3.7	2.2	1.5
64*	6	**8.6**	**2.7**	**1.7**	**1.2**	**0.8**
64*	8	15.8	5.4	3.5	2.1	1.4
96	4	10.1	2.9	1.8	1.2	0.8
128*	4	15.4	4.2	2.6	1.6	1.0

Table 11: Parallel Efficiency of SPME. **System β-hairpin, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{20}$**

Parallel efficiency(CPU time) of SPME as a function of interpolation order and grid points. The reciprocal space cutoff is fixed at $\hat{g}_c = 20$. The efficiency is measured by total CPU time (in seconds) for 100 steps on different QCDOC partitions, ranged from 16 node to 512 nodes, running at 400MHz. The most efficient options are shown in bold font.

L_{FFT}	n	16 nodes	64 nodes	128 nodes	256 nodes	512 nodes
56	6	9.5	3.1	2.0	1.4	1.0
56*	8	17.3	5.8	3.7	2.4	1.6
56*	10	31.8	11.4	7.0	4.1	2.8
64	6	9.1	2.8	1.9	1.3	0.9
64*	8	16.3	5.6	3.8	2.4	1.6
72*	6	**10.5**	**3.4**	**2.1**	**1.4**	**0.9**
72*	8	17.4	5.8	3.6	2.3	1.5
128	4	16.0	4.5	2.8	1.7	1.1

Table 12: Parallel Efficiency of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{30}$

Parallel efficiency(CPU time) of SPME as a function of interpolation order and grid points. The reciprocal space cutoff is fixed at $\hat{g}_c = 30$. The efficiency is measured by total CPU time (in seconds) for 100 steps on different QCDOC partitions, ranged from 16 node to 512 nodes, running at 400MHz. The most efficient options are shown in bold font.

L_{FFT}	n	16 nodes	64 nodes	128 nodes	256 nodes	512 nodes
64	4	23.0	6.9	4.3	2.6	1.6
64	6	47.2	14.1	8.9	5.4	3.4
64*	8	103.7	31.1	18.9	11.4	7.0
72	4	25.6	7.2	4.3	2.6	1.7
72*	6	48.5	14.6	8.9	5.4	3.4
72*	8	98.4	29.9	19.6	11.4	7.0
80	4	26.9	7.6	4.6	2.7	1.7
80*	6	51.0	14.6	8.9	5.4	3.2
80*	8	98.1	29.9	18.4	10.8	6.4
96*	6	48.7	14.0	8.9	5.2	3.0
128*	4	**33.6**	**9.2**	**5.6**	**3.3**	**2.0**

Table 13: Parallel Efficiency of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{34}$

Parallel efficiency(CPU time) of SPME as a function of interpolation order and grid points. The reciprocal space cutoff is fixed at $\hat{g}_c = 34$. The efficiency is measured by total CPU time (in seconds) for 100 steps on different QCDOC partitions, ranged from 16 node to 1024 nodes, running at 400MHz. The most efficient options are shown in bold font.

L_{FFT}	n	16 nodes	64 nodes	128 nodes	256 nodes	512 nodes	1024 nodes
72	4	25.5	7.5	4.6	2.7	1.6	1.0
72	6	49.1	15.1	9.8	5.6	3.3	2.2
72	8	101.4	31.0	19.7	11.0	6.9	4.7
80	4	26.8	7.9	4.7	2.8	1.7	1.1
80	6	51.7	14.8	9.0	5.3	3.3	2.2
80*	8	99.4	28.9	18.8	11.1	6.4	4.3
96	4	28.6	8.2	4.9	2.9	1.8	1.2
96*	6	**51.8**	**14.9**	**9.0**	**5.3**	**3.2**	**2.2**
96*	8	94.9	28.3	17.7	10.5	6.1	4.0
160	4	54.3	14.8	8.3	4.5	2.5	1.5
256*	4	—	38.8	20.0	9.9	5.1	3.0

Table 14: Parallel Efficiency of SPME. **System HIV-1**, $\hat{\mathbf{g}}_\mathbf{c} = \mathbf{38}$

Parallel efficiency(CPU time) of SPME as a function of interpolation order and grid points. The reciprocal space cutoff is fixed at $\hat{g}_c = 38$. The efficiency is measured by total CPU time (in seconds) for 100 steps on different QCDOC partitions, ranged from 16 node to 1024 nodes, running at 400MHz. The most efficient options are shown in bold font.

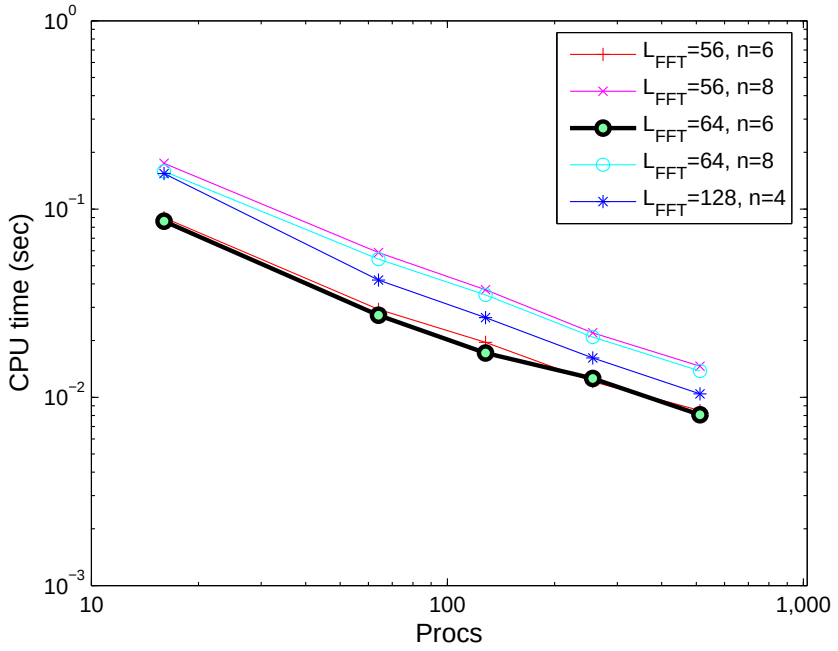
L_{FFT}	n	16 nodes	64 nodes	128 nodes	256 nodes	512 nodes	1024 nodes
80	6	51.5	15.2	9.3	5.6	3.3	2.3
80	8	102.1	29.8	19.2	11.0	6.5	4.5
80	10	180.5	55.5	34.4	21.2	11.8	8.2
90	6	52.3	16.4	9.7	5.5	3.2	2.0
90	8	100.3	31.2	17.8	10.7	6.4	4.0
96	6	51.0	15.0	9.5	5.5	3.2	2.2
96*	8	97.9	28.9	18.6	10.9	6.4	3.9
128*	6	**56.2**	**16.4**	**9.2**	**5.3**	**3.2**	**2.1**
128*	8	95.5	28.6	17.0	9.8	5.8	3.5
160	4	56.4	15.3	8.2	4.6	2.5	1.7
256	4	—	39.9	20.3	10.3	5.3	3.1

Figures 13 and 14 show the parallel efficiency (CPU time) of SPME computations for the two water/protein systems computed using a series of reciprocal space cutoffs and SPME parameters sets within the error tolerence. From the two figures, it is easy to see that decreasing the interpolation order n and increasing the FFT grid size L_{FFT} leads to higher efficiency. However, this trend is only followed if L_{FFT} is of at least moderate size (from 64 to 128). As mentioned in the previous subsection, there are two ways to reach the desired approximation error–either by increasing L_{FFT} or by increasing n. Thus, if one would like to decrease n to attain higher efficiency, than a larger L_{FFT} must be selected to compensate. This strategy will generally work well but, of course, will fail if pushed too far. For instance, as shown in Figure. 14(b), although it is possible to compensate for the accuracy lost by taking the interpolation order as small as $n = 4$ by increasing the number of 3D-FFT lattice points to $L_{\mathrm{FFT}} = 256$, the 3D-FFT calculation becomes the dominant phase yielding less attractive scaling/efficiency than using a more moderate number of lattice points with a larger interpolation order, e.g. $\{L_{\mathrm{FFT}} = 96, n = 6\}$. Again, the optimal set of SPME parameters for a given processor number will depend on system size and composition as well as the machine architecture and the parallel implementation.

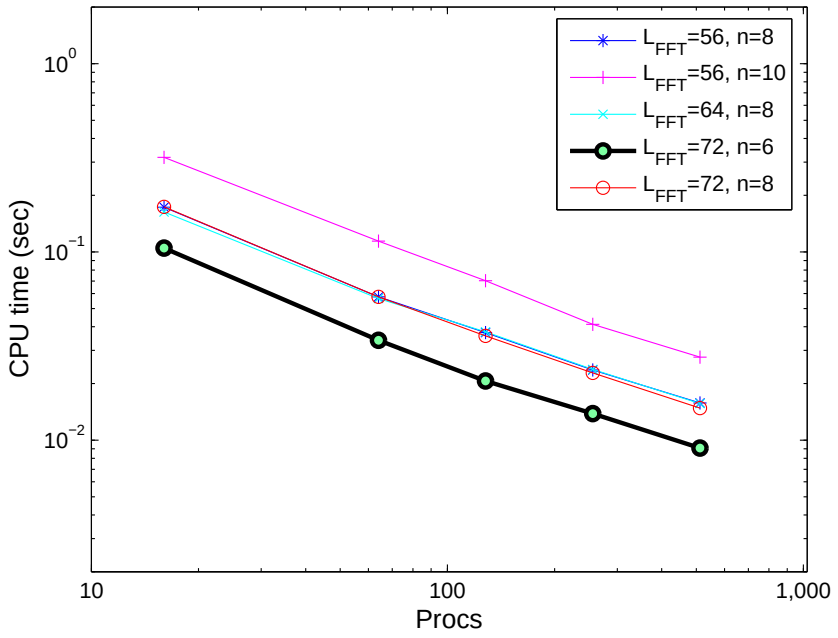
An intriguing observation that can be gleaned from Figure 13 and 14 is that at fixed interpolation order, the computational efficiency has no significant decrease as L_{FFT} increases (in a moderate-size range). This would certainly not be the case in scalar platforms as more scalar work is required. On parallel platforms, however, the larger L_{FFT} is taken, the smaller the processor's marginal volume and the better the load balance will be, as discussed in Section. 5. Thus, by increasing the grid size (i.e. decreasing the grid space), the amount of the computational work saved from charge assignment and force interpolation processes is nearly the same as the increased amount of the communicational cost required by the spherically truncated

3D-FFT.

It is, of course, possible to alleviate the interpolation phase bottleneck by improving the decomposition. Instead of using one particle "Bin" size optimized for the real space force evaluation, a second finer volume decomposition, "FineBin", can be introduced. In this way, the high density regions could be spread out over the more processors and low density regions could be "clumped together" over fewer processors. However, given the difficulty of implementing complex communication patterns using QMP [10], the only message passing interface currently supported by QCDOC, this strategy will be investigated as part of future work.



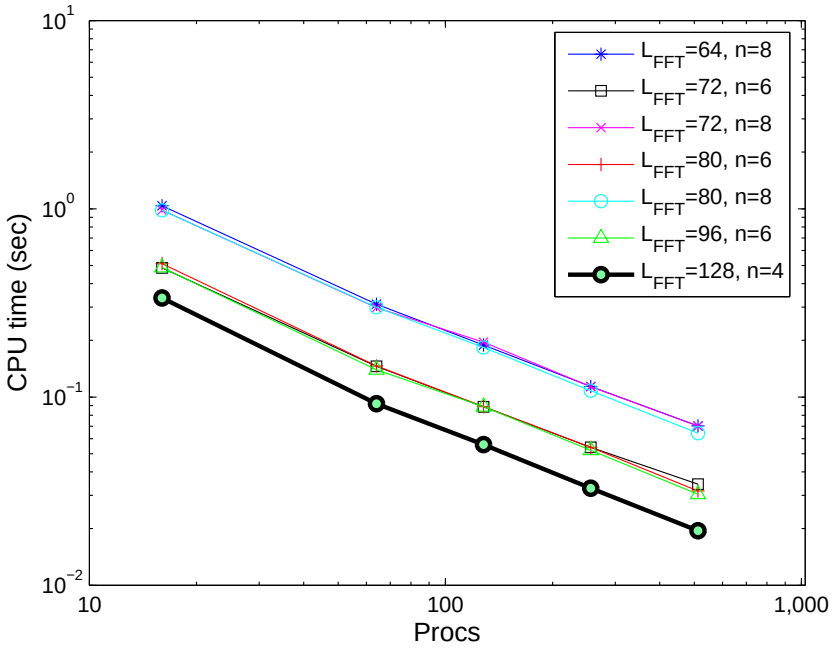
(a) Efficiency of SPME. $\hat{g}_c = 18$



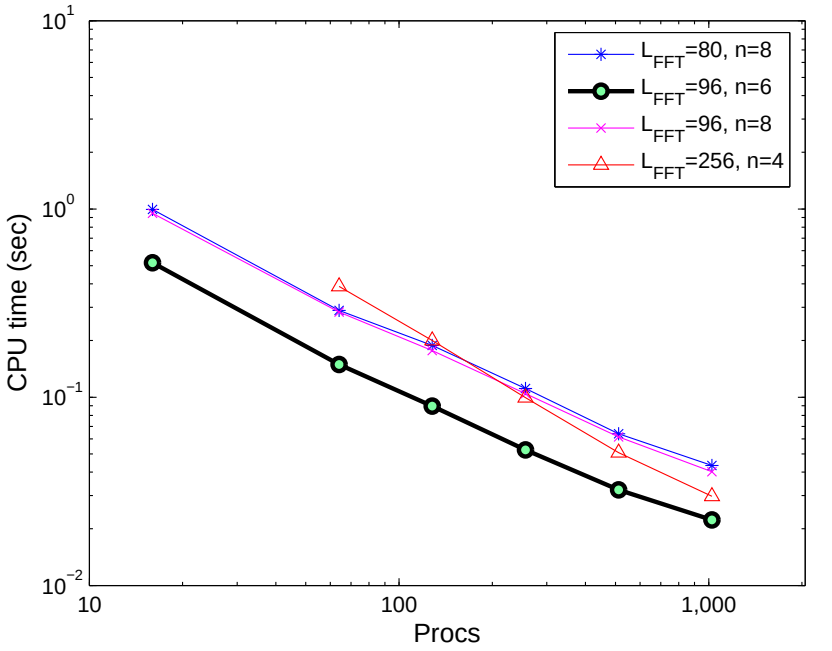
(b) Efficiency of SPME. $\hat{g}_c = 20$

Figure 13: Parallel Efficiency (CPU time) of SPME. **System β-hairpin.**

The first β-hairpin of Strptococcal Protein G protein/water system (3579 atoms in total) was tested. (a) and (b) are for different spherical cutoff $\hat{g}_c$, 18 and 20, respectively. The figures only plot the options which produce desired approximation errors. The options which produce the best efficiencies are high lighted.

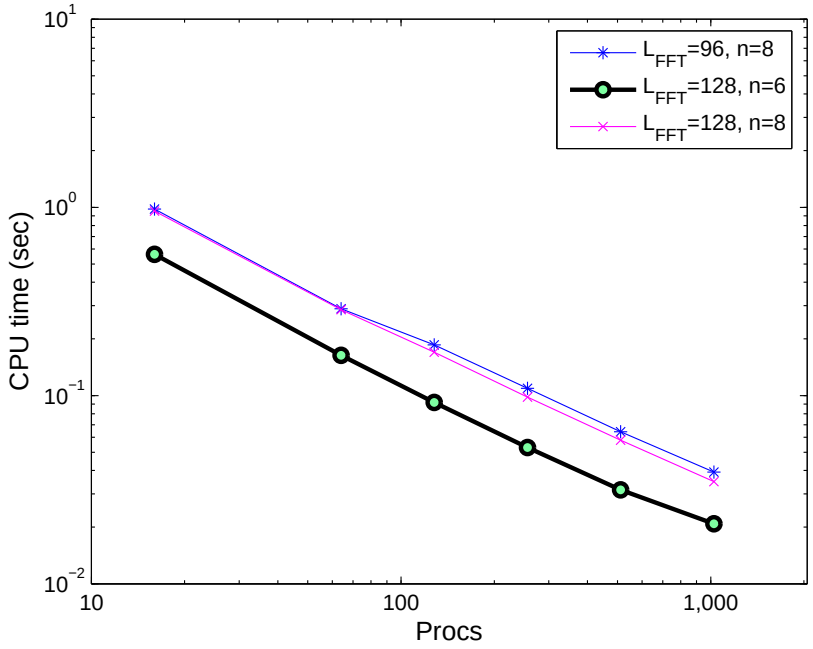


(a) Efficiency of SPME. $\hat{g}_c = 30$



(b) Efficiency of SPME. $\hat{g}_c = 34$

Figure 14: Parallel Efficiency (CPU time) of SPME. **System HIV-1.** (cont.)

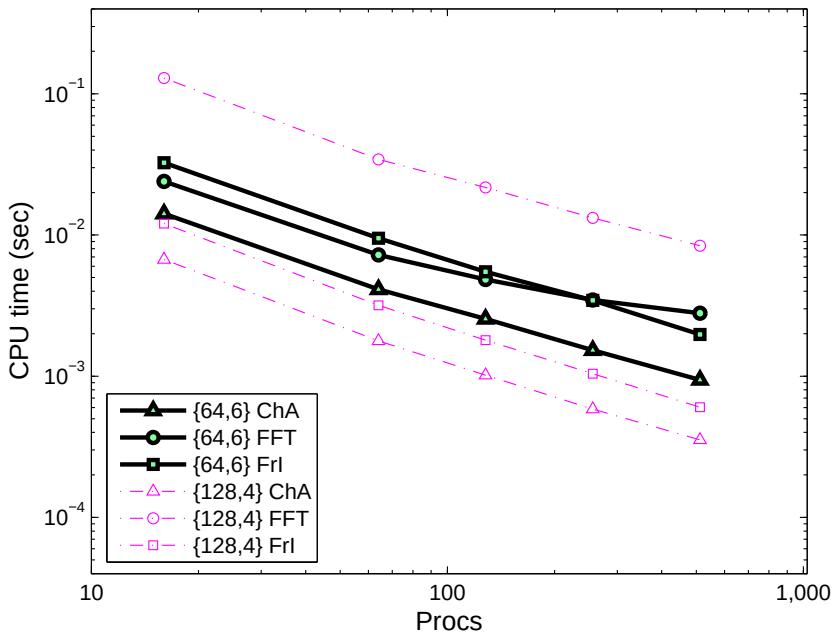


(c) Efficiency of SPME.$\hat{g}_c = 38$

Figure 14: Parallel Efficiency (CPU time) of SPME. **System HIV-1.**

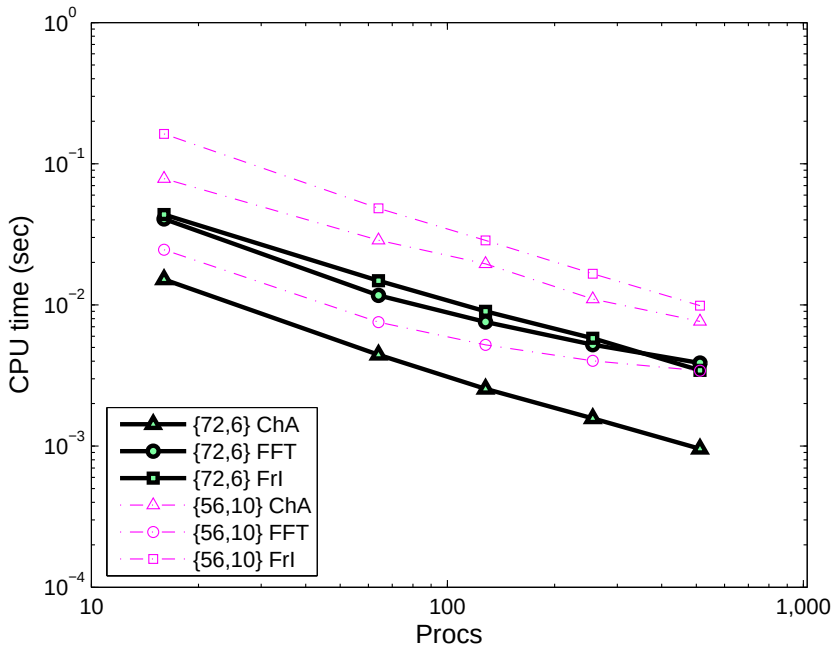
HIV-1 Protease protein/water system (29508 atoms in total) was tested. (a), (b) and (c) are for different spherical cutoff $\hat{g}_c$, 30, 34 and 38, respectively. The figures only plot the options which produce desired approximation errors. The options which produce the best efficiencies are high lighted.

In order to summarize the discussion, the computational cost of each of the SPME phases is shown in Figure. 15 and Figure. 16 for each of the computer experiments performed. It is clear that the spherically cutoff 3D-FFT is not the bottleneck in electrostatic computation on large parallel machines. It can be seen that the 3D-FFT does not scale as well as the interpolation phase, which may lead to the 3D-FFT computation becoming the bottleneck on yet larger supercomputers. Finally, Figure. 17 presents the parallel efficiency of the best results found in the study. For the HIV-1 water system with 30K atoms, a parallel speedup of 440x was generated on 1024 processors using the new framework, a result limited by the parallel efficiency of the charge interpolation phase.



(a) Three Components of SPME. $\hat{g}_c = 18$

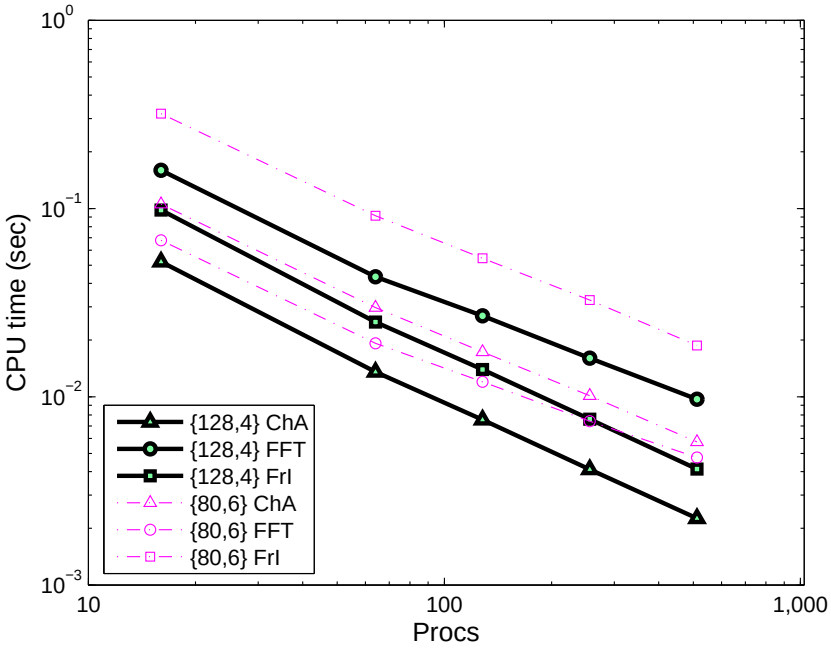
Figure 15: Parallel Efficiency (CPU time) of three components in SPME.**System β-hairpin.** (cont.)



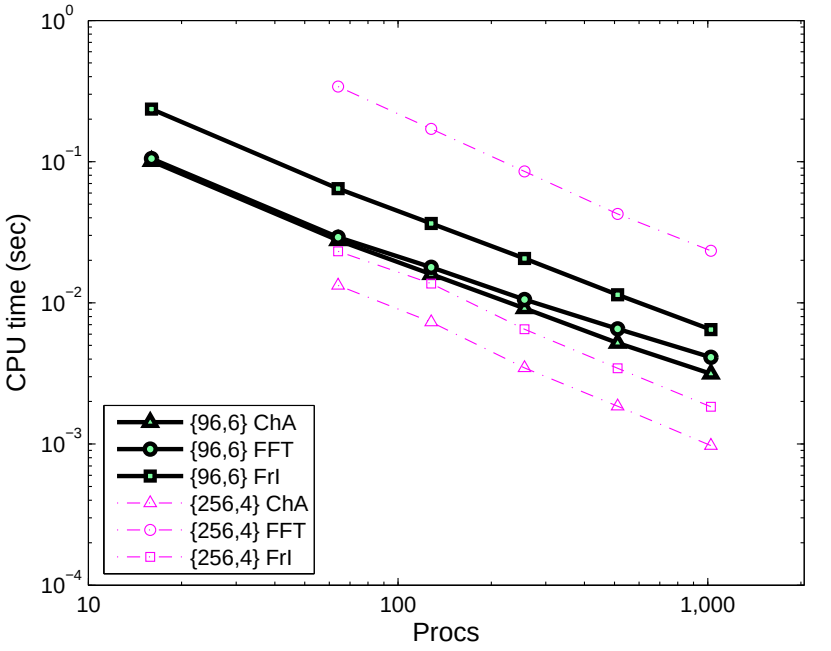
(b) Three Components of SPME. $\hat{g}_c = 20$

Figure 15: Parallel Efficiency (CPU time) of three components in SPME. **System β-hairpin.**

The first β-hairpin of Strptococcal Protein G protein/water system (3579 atoms in total) was tested. (a) and (b) are for different spherical cutoff $\hat{g}_c$, 18 and 20, respectively. Three components–charge assignment(ChA), 3D FFT(FFT) and force interpolation(FrI)–are plotted. Two options are shown for each subplot, including the most efficient set (high lighted) and another acceptable set. The median number across different processors for both charge assignment and force interpolation are plotted, and the load imbalance for 3D FFT could be ignored.

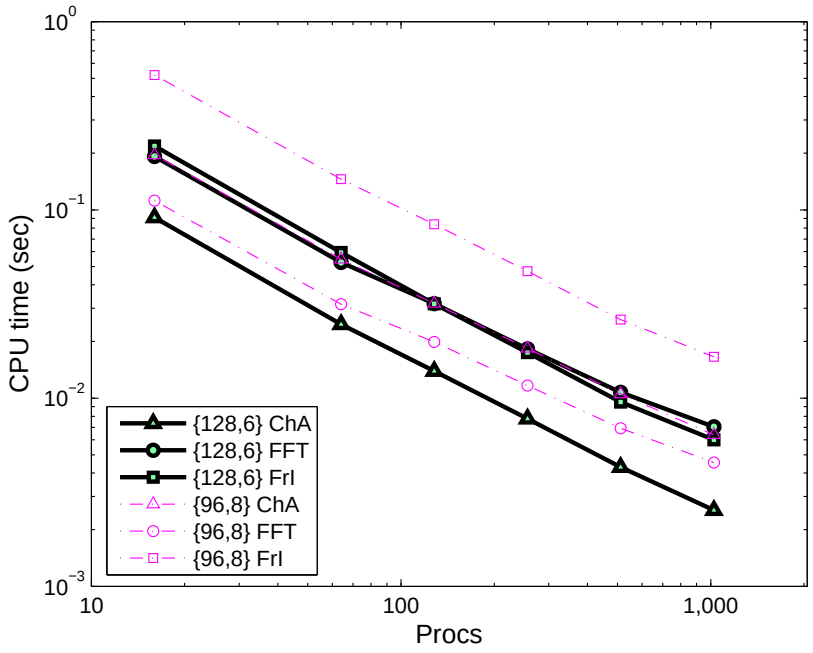


(a) Three Components of SPME. $\hat{g}_c = 30$



(b) Three Components of SPME. $\hat{g}_c = 34$

Figure 16: Parallel Efficiency (CPU time) of three components in SPME. **System HIV-1.** (cont.)



(c) Efficiency of SPME.$\hat{g}_c = 38$

Figure 16: Parallel Efficiency (CPU time) of three components in SPME. **System HIV-1.**

HIV-1 Protease protein/water system (29508 atoms in total) was tested. (a) and (b) are for different spherical cutoff $\hat{g}_c$, 30 and 34, respectively. Three components–charge assignment(ChA), 3D FFT(FFT) and force interpolation(FrI)–are plotted. Two options are shown for each subplot, including the most efficient set (high lighted) and another acceptable set. The median number across different processors for both charge assignment and force interpolation are plotted, and the load imbalance for 3D FFT could be ignored.

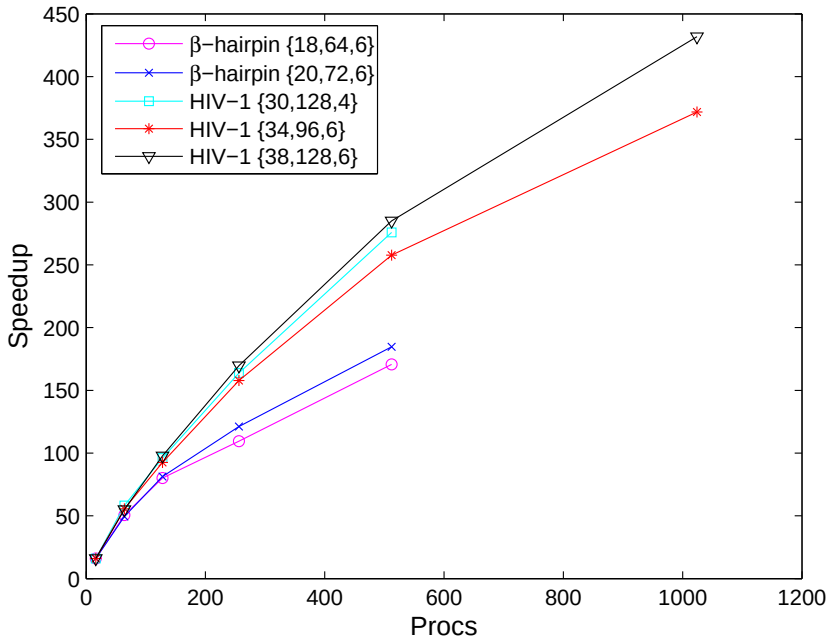


Figure 17: Parallel Efficiency (speedup) of SPME.

The speedup of five most efficient options in the β-hairpin and the HIV-1 system simulations are shown. The parameter triplet is in the form of $\{\hat{g}_c, L_{\mathrm{FFT}}, n\}$. The best speedup in 1024 node machine is 440x.

6.4 Relationship to previous work by others

NAMD [65, 64] is an MD application that exhibits high parallel performance on several computer architectures. NAMD benefits greatly from its novel message-driven execution model, charm++ [49]. The package combines force and spatial decompositions together and utilizes processor virtualization to solve load imbalance problems. NAMD employs the PME method to evaluate the long-range part of the Ewald sum. Since NAMD makes use of the Multiple Time Step integration methods, the reciprocal space part of the Ewald sum only has to be computed every few small time steps. Thus, NAMD scales to thousands of processors although the PME implementation is only parallelized over a hundred processors. The greatest speedup attained by NAMD on 1024-processor Lemieux cluster at PSC was 540x in benchmark simulation of ApoA1 (92K atoms), when PME was calculated every time step [65]. Given the data presented here, it is clear that our MD parallel framework, MDoC with embedded parallel SPME, can also reach this level of parallel efficiency for a whole MD time step.

Blue Matter [41] is an MD application designed for use on the BlueGene/L supercomputer with its 3D torus communication network [39]. BlueMatter takes advantage of BlueGene/L's novel architectural features, including the hardware-embedded global broadcast routing capability, and the fast nearest neighbour communication. The package uses a spatial decomposition to compute the long-range part of the Ewald summation via P^3ME. The scalability of the major components of an MD time-step for 43K atom system under BlueMatter have been presented in the literature[41]. Although CPU times were not given explicitly, BlueMatter seems to spend approximately 0.006-0.007s/step in the P^3ME computation per interation on a 1024 node BlueGene/L partition. The results presented here for the HIV-1 system with 30K

atoms, independent of reciprocal cutoff, are comparable when the slower clock speed of QCDOC is taken into account.

In general, the inefficiency and inflexibility of the decomposition employed herein to treat the charge interpolation phase of SPME method is the bottleneck to achieving better scaling than existing frameworks. It is expected that the improvements to our framework described above and others will alleviate this difficulty.

7 Summary and Future work

In order for the potential of atomistic molecular dynamics to impact biophysics to be unlocked, the time and length scales which the method probes must be increased. An important way to increase the speed, accuracy and computational efficiency of atomistic molecular dynamics is to perform coupled research into new algorithms, software development and computer architectures. In this way, methods, software and architecture can be adjusted to yield the best possible results.

7.1 Summary and Contributions

In this dissertation, two contributions have been accomplished to improve the accuracy and computational efficiency of atomistic molecular dynamics. First of all, a fine grained parallel 3D-FFT framework has been designed specifically for QCDOC, a 6D-torus network architecture. The highlight of this framework are its competitive scalability and flexibility. With the full exploit of 24 off-node links provided by QCDOC's novel design, the QMP-based implementation shows a speedup of 895x on 4096 nodes for complex-to-complex 3D FFTs of size $128 \times 128 \times 128$. Moreover, any existing sequential 1D-FFT package could be easily embedded into the framework. This provides it more flexibility. With these two properties, the parallel 3D-FFT framework could be regarded as a toolbox for daily use, which can provide high scalability and flexibility. This has already been beyond our initial purpose, which is to design a scalable real-to-complex 3D-FFT algorithm specifically for SPME in MD simulation.

Secondly, the accuracy and efficiency of the long range force evaluation in large

heterogeneous biophysical systems, a known bottleneck to high scaling on modern massively parallel architectures, has been studied. On both scalar and parallel machines with less than 1000 processors, the SPME method is more efficient than the Ewald technique[29, 67, 46]. This is due to the fact that the SPME method has a lower scaling with particle number. Nonetheless, the use of a spherical cutoff in reciprocal space, paying careful attention to accuracy in tuning the interpolation order and the FFT grid spacing, and on the base of our highly scalable 3D-FFT framework, introducing a parallel implementation that respects/exploits the spherical truncation is found to yield large gains in efficiency and accuracy both in scalar and in parallel. Accuracy is improved by eliminating inaccurate high Fourier components generated by the Euler exponential spline interpolation. Efficiency is improved by implementing a spherically truncated 3D-FFT which reduces the computational overhead required. On parallel machines, the spherical truncation reduces the data volume required to parallelize the 3D-FFTs such that the 3D-FFT is no longer the dominant part of SPME computation on less than 1000 processors. These conclusions were validated by studies of two realistic protein/water systems consisting of 3579 and 29508 atoms, respectively, on up to 1024 processors of the massively parallel QCDOC with its novel 6-D torus network architecture.

7.2 Future work

Future work will involve a number of important and exciting projects.

(1°) We can expand the functionality of our parallel spherical cutoff 3D FFT to let it support different cutoff radius $\hat{g}_c$ along different dimensions, if the simulation box is not a regular cube. Through this way, the efficiency of SPME would be

increased, with the accuracy retained.

(2°) The efficiency analyses show that for SPME alone, there is still much time occupied by off-node data communication, while the processors are keeping idle. It definitely does make sense to overlap the short-range forces computation with SPME communication. This is essential for the scalability of the whole MDoC package.

(3°) As we discussed in Section 6, more finer volume decomposition can be introduced to eliminate the load imbalance in charge interpolation and force interpolation.

(4°) Like other mature MD simulation packages [41, 64], it will be worthy to try to combine SPME with multi-timing step integration method together. Multi-timing Step for Ewald summation has been studied by G.W.Han [43].

(5°) Refine the parallel 3D-FFT framework, scaling to 10,000 processors and examine larger protein/water systems.

Appendices

Appendix A

Consider the real space part of the total potential energy function

$$\phi^{(real)}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = \frac{1}{2} \sum_{\hat{\mathbf{s}}} {\sum_{ij}}' \varphi_{ij}(|\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|)\chi_\gamma(|\mathbf{r}_{ij} + \overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}|, r_c) = \sum_{i>j} \varphi_{ij}(r_{ij})\chi_\gamma(r_{ij}, r_c).$$

expressed as a pairwise additive sum over particles, (ij) and images $(\mathbf{S})$. A switching function has been introduced explicitly in order to reduce the computation of the sum to order N operations. The sum is thus restricted to nearest or first image particle separations (see above). Two commonly employed switching functions are a smooth switch, $\chi_\lambda(r, r_c)$, and a hard/sharp switch, $\chi_h(r, r_c)$. The force using a smooth switching function is taken to be

$$-\nabla_k \phi^{(real)}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = -\sum_{i \neq k} \frac{\mathbf{r}_{ki}}{r_{ki}} \left[\frac{d\varphi_{ki}(r_{ki})}{dr_{ki}} \chi_\lambda(r_{ki}, r_c) + \varphi_{ki}(r_{ki}) \frac{d\chi_\lambda(r_{ki}, r_c)}{dr_{ki}} \right]$$

while under a hard/sharp cutoff, the force is approximated by

$$-\nabla_k \phi^{(real)}(\mathbf{R}, \overleftrightarrow{\mathbf{h}}) = -\sum_{i \neq k} \frac{\mathbf{r}_{ki}}{r_{ki}} \left[\frac{d\varphi_{ki}(r_{ki})}{dr_{ki}} \chi_h(r_{ki}, r_c) \right].$$

The error induced by use of either type switching function can be handled effectively using first order thermodynamic perturbation theory provided the pair potential goes to zero more quickly than $1/r^3$. Coulombic interactions, for example, should be treated using Ewald methods in periodic systems. The Ewald method introduces a long range reciprocal space piece to the potential energy and the effective real space potential falls off as $\mathrm{erfc}(\alpha r)/r$. Dispersion is typically treated using a cutoff. Reciprocal sums could be employed to compute the dispersion interaction but unlike the Coulomb interaction, the coefficients are not guaranteed to obey separable combining

rules, $C_{ij} = C_i C_j$.)

The corrections to the potential energy and the pressure using the smooth switching function will be considered first. Assuming for simplicity one particle type and N particles, in 3 spatial dimensions, the correction to the total potential energy becomes

$$\begin{aligned}\Delta\phi^{(real)}(V, r_c, \lambda) &= \left[\frac{\rho N}{2}\right]\left[\int_{r_c-\lambda}^{r_c} dr \int d\Omega r^2 \varphi(r)\left[1-\chi_\lambda(r, r_c)\right] g(r)\right. \\ &\qquad \left. + \int_{r_c}^{\infty} dr \int d\Omega r^2 \varphi(r) g(r)\right]\end{aligned}$$

where ρ is the density and $g(r)$ is the radial distribution function. The correction can be applied exactly on a configuration by configuration basis, provided the factor of N is replaced by a sum over all atoms and the instantaneous particle dependent radial distribution function is employed, $g(r) = g_k^{(inst)}(r, \Omega, \mathbf{R})$. Approximating the radial distribution function by unity beyond $r_c - \lambda$, $g(r) = g_k^{(inst)}(r, \Omega, \mathbf{R}) = 1$, reduces the correction to a constant, in general,

$$\Delta\phi^{(real)}(V, r_c, \lambda) = 2\pi\rho N\left[\int_{r_c-\lambda}^{r_c} dr r^2 \varphi(r)\left[1-\chi_\lambda(r, r_c)\right] + \int_{r_c}^{\infty} dr r^2 \varphi(r)\right]$$

Within this approximation, the long range correction to the pressure can be shown to be

$$\begin{aligned}V\Delta P^{(real)} &= \left[\frac{2\pi\rho N}{3}\right]\left[\int_{r_c-\lambda}^{r_c} dr r^3 \varphi(r)\frac{d\chi_\lambda(r, r_c)}{dr} - \int_{r_c-\lambda}^{r_c} dr r^3 \frac{d\varphi(r)}{dr}\left(1-\chi_\lambda(r, r_c)\right)\right. \\ &\qquad \left. - \int_{r_c}^{\infty} dr r^3 \frac{d\varphi(r)}{dr}\right] \\ &= \Delta\phi^{(real)}(V, r_c, \lambda).\end{aligned}$$

where an integration by parts has been performed and it has been assumed that the switching function has a continuous first derivative. The long range correction to the

pressure and the potential are thermodynamically consistent,

$$V\Delta P^{(real)} = -V\frac{\partial \Delta\phi^{(real)}(V, r_c, \lambda)}{\partial V},$$

if the cutoff, r_c and the healing length, λ, are taken independent of the volume and ρ is expressed as N/V. Finally, if we take

$$\begin{aligned} \chi_\lambda(r, r_c) &= 1 + \tilde{r}^2(2\tilde{r} - 3) \\ \tilde{r} &= \frac{(r - r_c + \lambda)}{\lambda} \end{aligned}$$

on the range $r_c - \lambda < r < r_c$ and $\varphi(r) = -C/r^6$ then

$$\Delta\phi^{(real)}(V, r_c, \lambda) = -\left[\frac{2C\pi\rho N}{\lambda^3}\right]\left[\frac{1}{s_c} + \frac{1}{s_c - 1} + 2\ln\left(\frac{s_c - 1}{s_c}\right)\right]$$

where $s_c = r_c/\lambda$.

The corrections to the average potential energy and the pressure under the sharp switching function are different,

$$\Delta\phi^{(real)}(V, r_c) = 2\pi\rho N\left[\int_{r_c}^{\infty} dr r^2\varphi(r)\right]$$

and

$$V\Delta P^{(real)} = 2\pi\rho N\left[\frac{1}{3}\varphi(r_c)r_c^3 + \int_{r_c}^{\infty} dr r^2\varphi(r)\right] \neq \Delta\phi^{(real)}(V, r_c).$$

The two corrections are thermodynamically consistent,

$$V\Delta P^{(real)} = -V\frac{\partial \Delta\phi^{(real)}(V, r_c)}{\partial V},$$

if the cutoff, r_c, is taken to be volume dependent, $r_c = V^{1/3} s_c$ and ρ is expressed as N/V. Therefore, if a sharp switching function is used in a constant pressure simulation, then the cutoff must be allowed to fluctuate in time, $r_c(t) = r_c(0)[V(t)/V(0)]^{1/3}$.

Appendix B

In this appendix, the reciprocal space form of the average potential energy for an infinitely replicated periodic system is derived. The starting point is the real space expression for the average potential energy

$$\langle\phi\rangle = \frac{1}{2}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}')\sum_{\hat{\mathbf{S}}}\phi(\mathbf{r}-\mathbf{r}'+\overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}})$$

where the sum over the periodic replicas of the primary domain has been introduced (i.e. $\hat{\mathbf{S}}$). (If the charge density is comprised of point particles, $\mathrm{n}(r)=\sum_i q_i\delta(\mathbf{r}-\mathbf{r}_i)$, then the form given in the text is recovered.) The Poisson Summation Formula states

$$\begin{aligned}\sum_{\hat{\mathbf{S}}} F(\overleftrightarrow{\mathbf{h}}\hat{\mathbf{S}}) &= \left[\frac{1}{\det\overleftrightarrow{\mathbf{h}}}\right]\sum_{\hat{\mathbf{g}}} G(\hat{\mathbf{g}}\overleftrightarrow{\mathbf{h}}^{-1}) \\ G(\hat{\mathbf{g}}\overleftrightarrow{\mathbf{h}}^{-1}) &= \int_{allspace} d\mathbf{r}F(\mathbf{r})\exp(-i2\pi\hat{\mathbf{g}}\overleftrightarrow{\mathbf{h}}^{-1}\mathbf{r})\end{aligned}$$

where the integral is over all space. Poisson Summation can be directly applied to yield

$$\begin{aligned}\langle\phi\rangle &= \frac{1}{2V}\sum_{\hat{\mathbf{g}}''}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}')\tilde{\phi}(\mathbf{g}'')\exp(i\mathbf{g}''\cdot(\mathbf{r}-\mathbf{r}')) \\ \langle\phi\rangle &= \frac{1}{2V^3}\sum_{\hat{\mathbf{g}},\hat{\mathbf{g}}',\hat{\mathbf{g}}''}\bar{\mathrm{n}}(\mathbf{g})\bar{\mathrm{n}}(\mathbf{g}')\tilde{\phi}(\mathbf{g}'')\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\exp(i\mathbf{g}\cdot\mathbf{r})\exp(i\mathbf{g}'\cdot\mathbf{r}')\exp(i\mathbf{g}''\cdot(\mathbf{r}-\mathbf{r}')) \\ \langle\phi\rangle &= \frac{1}{2V}\sum_{\hat{\mathbf{g}}}\bar{\mathrm{n}}(\mathbf{g})\bar{\mathrm{n}}(-\mathbf{g})\tilde{\phi}(-\mathbf{g}) = \frac{1}{2V}\sum_{\hat{\mathbf{g}}}|\bar{\mathrm{n}}(\mathbf{g})|^2\tilde{\phi}(-\mathbf{g})\end{aligned}$$

where $\tilde{\phi}(\mathbf{g})$ is the Fourier Transform of $\phi(\mathbf{r})$ evaluated at the reciprocal lattice vector $\mathbf{g}=2\pi\hat{\mathbf{g}}\overleftrightarrow{\mathbf{h}}^{-1}$. Note, that the $\bar{\mathrm{n}}(\mathbf{g})$ are the Fourier **components** of the charge density; they are determined by integration over the finite volume defined by the simulation

cell, $\overleftrightarrow{\mathbf{h}}$.

Care must be taken at the point $\mathbf{g} = 0$ if the potential function is long range (decays as $1/r^n, n \leq 3$)

$$\tilde{\phi}^{(0)} = \frac{1}{2V}|\bar{\mathrm{n}}(0)|^2\tilde{\phi}^{(\mathrm{short})}(0) + \frac{1}{2V}|\bar{\mathrm{n}}(0)|^2 \lim_{\mathbf{g}\to 0}\tilde{\phi}^{(\mathrm{long})}(\mathbf{g}).$$

Here the potential has been divided into a **purely** long and **purely** short range part. The long range part of the $\mathbf{g} = 0$ term can be written in the prelimit form

$$\tilde{\phi}^{(0,\mathrm{long})} = \frac{1}{2V}\lim_{\xi\to\infty}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}')\int_{D(\overleftrightarrow{\tilde{\mathbf{h}}}\ (\xi))} d\mathbf{r}''\phi^{(\mathrm{long})}(\mathbf{r}''+\mathbf{r}-\mathbf{r}')$$

which follows from the Poisson Summation formula. The domain, $D(\overleftrightarrow{\tilde{\mathbf{h}}}(\xi))$, is assumed to scaled by the parameter ξ which will be taken to infinity, later. Changing to scaled coordinates yields

$$\tilde{\phi}^{(0,\mathrm{long})} = \frac{1}{2V}\lim_{\xi\to\infty}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}')\xi^3\int_{D(\overleftrightarrow{\tilde{\mathbf{h}}})} d\tilde{\mathbf{r}}\ \phi^{(\mathrm{long})}\left(\xi\left[\tilde{\mathbf{r}}+\frac{(\mathbf{r}-\mathbf{r}')}{\xi}\right]\right)$$

This form can be Taylor expanded and the limiting properties examined.

If $\phi^{(\mathrm{long})}(\mathbf{r}) = 1/|\mathbf{r}|$ then

$$\begin{aligned}
\tilde{\phi}^{(0,\mathrm{coul})} &= \frac{1}{2V}\lim_{\xi\to\infty}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}') \\
&\times \int_{D(\overleftrightarrow{\tilde{\mathbf{h}}})} d\tilde{\mathbf{r}}\left\{\frac{\xi^2}{|\tilde{\mathbf{r}}|} - \frac{\xi\tilde{\mathbf{r}}\cdot(\mathbf{r}-\mathbf{r}')}{|\tilde{\mathbf{r}}|^3} + \frac{1}{2}(\mathbf{r}-\mathbf{r}')\left[\nabla\nabla\left(\frac{1}{|\tilde{\mathbf{r}}|}\right)\right](\mathbf{r}-\mathbf{r}')\right\} \\
\tilde{\phi}^{(0,\mathrm{coul})} &= \frac{1}{2V}|\bar{\mathrm{n}}(0)|^2\gamma + \frac{1}{2}\cdot\frac{1}{2V}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}\int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r}'\mathrm{n}(\mathbf{r})\mathrm{n}(\mathbf{r}')[(\mathbf{r}-\mathbf{r}')\overleftrightarrow{\mathbf{T}}(\mathbf{r}-\mathbf{r}')]
\end{aligned}$$

where

$$\begin{aligned} \gamma &\propto \int_{D(\overleftrightarrow{\tilde{\mathbf{h}}})} d\tilde{\mathbf{r}} \, \frac{1}{|\tilde{\mathbf{r}}|} \\ \overleftrightarrow{\mathbf{T}} &= \int_{D(\overleftrightarrow{\tilde{\mathbf{h}}})} d\tilde{\mathbf{r}} \left[\nabla\nabla \left(\frac{1}{|\tilde{\mathbf{r}}|} \right) \right] \end{aligned}$$

The constant parameter, γ, is called the infinite background. It does not contribute for neutral charge densities ($\bar{\mathrm{n}}(0) = 0$) and cancels in energy differences between systems with the same total charge. In contrast, energy differences between systems with the different total charge are ill-defined, although, an appropriately chosen reference system can sometimes be constructed to remedy the situation. The depolarization tensor, $\overleftrightarrow{\mathbf{T}}$, depends on the boundary condition at infinity, $D(\overleftrightarrow{\tilde{\mathbf{h}}})$ and vanishes if the system is assumed to be surrounded by a conductor.

Appendix C

Consider a real function, $f(\mathbf{r}) = f^*(\mathbf{r})$ with Fourier components,

$$\begin{aligned}\bar{f}(\mathbf{g}) &= \int_{D(\overleftrightarrow{\mathbf{h}})} d\mathbf{r} \exp[-i\mathbf{g}\cdot\mathbf{r}] f(\mathbf{r}) \\ &= V \int_0^1 ds_a \int_0^1 ds_b \int_0^1 ds_c e^{-2\pi i \hat{g}_a s_a} e^{-2\pi i \hat{g}_b s_b} e^{-2\pi i \hat{g}_c s_c} f(\overleftrightarrow{\mathbf{h}}\mathbf{s}),\end{aligned}$$

where $\mathbf{r} = \overleftrightarrow{\mathbf{h}}\mathbf{s}$, $\mathbf{g} = 2\pi \overleftrightarrow{\mathbf{h}}^{-1} \hat{\mathbf{g}}$, $V = \det \overleftrightarrow{\mathbf{h}}$ and the labels a, b, c represent the three directions defining the parallelepiped bounding the system. In order to accurately approximate the $\bar{f}(\mathbf{g})$ on a finite $\mathbf{g}$ space, $-L_\beta/2 < \hat{g}_\beta < L_\beta/2$, it useful to employ the Euler exponential spline

$$\begin{aligned}\exp\left(\frac{2\pi i \hat{g} p}{L}\right) &= d_n(\hat{g}, L) \sum_{\hat{s}=-\infty}^{\infty} M_n(p - \hat{s}) \exp\left(\frac{2\pi i \hat{g} \hat{s}}{L}\right) + \mathcal{O}\left(\frac{2|\hat{g}|}{L}\right)^n \\ d_n(\hat{g}, L) &= \frac{\exp\left(2\pi i (n-1)/L\right)}{\left[\sum_{j=0}^{n-2} M_n(j+1) \exp\left(2\pi i \hat{g} j / L\right)\right]}\end{aligned}$$

where $\hat{s}$ is an integer, p is a real number, n is the spline order assumed to be even and the $M_n(p)$ are the Cardinal B splines

$$\begin{aligned}M_2(p) &= 1 - |p - 1| \\ M_n(p) &= \left[\frac{p}{n-1}\right] M_{n-1}(p) + \left[\frac{n-p}{n-1}\right] M_{n-1}(p-1) \\ M_n(p) &\neq 0 \qquad\qquad 0 \le p \le n \\ \frac{dM_n(p)}{dp} &= M_{n-1}(p) - M_{n-1}(p-1)\end{aligned}$$

Inserting the Euler exponential spline into the first equation yields a well defined approximation to $\bar{f}(\mathbf{g})$,

$$\begin{aligned}
\bar{f}^{(\mathrm{Euler})}(\mathbf{g},\mathbf{L},n) &= [Vd_n^*(\hat{g}_a,L_a)d_n^*(\hat{g}_b,L_b)d_n^*(\hat{g}_c,L_c)] \\
&\quad\times\sum_{\hat{s}_a=0}^{L_a-1}\sum_{\hat{s}_b=0}^{L_b-1}\sum_{\hat{s}_c=0}^{L_c-1} e^{-2\pi i\hat{g}_a\hat{s}_a/L_a}e^{-2\pi i\hat{g}_b\hat{s}_b/L_b}e^{-2\pi i\hat{g}_c\hat{s}_c/L_c}f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}}) \\
&= [Vd_n^*(\hat{g}_a,L_a)d_n^*(\hat{g}_b,L_b)d_n^*(\hat{g}_c,L_c)]\times\mathrm{FFT}[f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}});\mathbf{L}]
\end{aligned}$$

where

$$\begin{aligned}
f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}}) &= \int_0^1 ds'_a\int_0^1 ds'_b\int_0^1 ds'_c\sum_{k_a}\sum_{k_b}\sum_{k_c} f(\overleftrightarrow{\mathbf{hs}'}) \\
&\quad\times M_n([s'_a-k_a]L_a-\hat{s}_a)M_n([s'_b-k_b]L_b-\hat{s}_b)M_n([s'_c-k_c]L_c-\hat{s}_c).
\end{aligned}$$

and $s_\beta=\hat{s}_\beta/L_\beta$.

Thus, accurate approximations to the Fourier components of $f(\mathbf{r})$, $\bar{f}^{(\mathrm{Euler})}(\mathbf{g},\mathbf{L},n)$, on the finite range, $-L_\beta/2<\hat{g}_\beta<L_\beta/2$ can be evaluated using a real-to-complex 3D-FFT of size $L_a\times L_b\times L_c$ in order $L^3\log L$ provided the function, $f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}})$, defined on the discrete real space, can be constructed in a computationally efficient manner. If $L_\beta>m+1$ then each point in the continuous space, $\{s'_a,s'_b,s'_c\}$ with $\mathbf{r}'=\overleftrightarrow{\mathbf{hs}}'$, is mapped to n^3 unique points on the discrete real space grid, $\{\hat{s}_a,\hat{s}_b,\hat{s}_c\}$ with $\hat{\mathbf{r}}=\overleftrightarrow{\mathbf{h}}\hat{\mathbf{s}}$, due to the finite support of the $M_n(p)$ indicating that a computationally efficient methods to generate $f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}})$ can be designed for many problems of interest. Finally, the error in the Euler spline interpolation does not vary uniformly with $\hat{\mathbf{g}}$ but rather **increases** as $|\hat{\mathbf{g}}|^n$. It is, therefore, important to choose $L_\beta>2\hat{g}_\beta^{(max)}$ where $\hat{g}_\beta^{(max)}$ is the largest desired lattice vector in the β direction (e.g. otherwise, Fourier coefficients with very large error will be used in the desired application).

In the text, the Fourier components of a charge density consisting of N point particles is of interest, $\mathrm{n}(\mathbf{r}) = \sum_{j=1}^{N} q_j \delta(\mathbf{r} - \mathbf{r}_j)$. Substitution into the expression for convolution function yields

$$\begin{aligned} n^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{h}}\mathbf{s}) &= V^{-1} \sum_{j=1}^{N} \sum_{k_a} \sum_{k_b} \sum_{k_c} q_j M_n([s_{a,j} - k_a]L_a - \hat{s}_a) \\ &\quad \times M_n([s_{b,j} - k_b]L_b - \hat{s}_b) M_n([s_{c,j} - k_c]L_c - \hat{s}_c). \end{aligned}$$

where $\mathbf{s}_j = \overleftrightarrow{\mathbf{h}}^{-1} \mathbf{r}_j$ is the scaled position of point particle, j. The computational cost to the evaluate the convoluted density function is $n^3 N$ because each particle position is mapped to n^3 discrete grid points. Thus, the Fourier components of the charge density on the desired finite range can be generated in order $N \log N$, limited by the scaling of 3D-FFTs. Again, the number of particles is assumed to increase at constant density $N = \rho V$ and in order to keep the grid spacing in the FFTs constant, the size of the FFTs must scale as particle number to the one-third power, $L_\beta \propto N^{1/3}$, such that $L^3 \log L \sim N \log N$. Methods of reducing the cost of the 3D-FFTs are discussed in the text and Appendix D. Last, since the Cardinal B splines possess $n - 1$ continuous derivatives, forces derived from potential energy functions that depend on Euler spline based Fourier components, $\bar{\mathrm{n}}^{(\mathrm{Euler})}(\mathbf{g}, \mathbf{L}, n)$, will, also, possess $n - 1$ continuous derivatives (i.e. with respect to particle position).

Appendix D

PME

To accurately approximate the same $\bar{f}(\mathbf{g})$ on a finite $\mathbf{g}$ space, PME method employs the Lagrangian polynomial interpolation, instead of Euler exponential spline

$$\exp\left(\frac{2\pi i\hat{g}p}{L}\right) = \sum_{\hat{s}=-\infty}^{\infty} W_{n,\hat{s}}(p)\exp\left(\frac{2\pi i\hat{g}\hat{s}}{L}\right) \equiv \mathrm{P}(p)$$

where $\hat{s}$ is an integer, p is a real number, n is the order of the Lagrangian polynomial. $W_{n,\hat{s}}(x), \hat{s} \in \Omega^1(p,n)$, are the Lagrangian polynomials uniquely defined by

$$\begin{aligned} &W_{n,\hat{s}}(x) = 1; \quad \hat{s} \in \Omega^1(p,n), \quad x = \hat{s}; \\ &W_{n,\hat{s}}(x) = 0; \quad \hat{s}, x \in \Omega^1(p,n), \quad x \neq \hat{s}; \\ &W_{n,\hat{s}}(x) \equiv 0; \quad \hat{s} \notin \Omega^1(p,n), \quad x \in \mathbb{R}. \end{aligned}$$

where $\Omega^1(p,n)$ be defined as the subset of the integers in one dimensional range $[0,L]^1$, containing the $n+1$ integer numbers which are closest to p (periodic boundary conditions are used).

Inserting the Lagrangian polynomial into the equation of $\bar{f}(\mathbf{g})$ yields a well defined approximation,

$$\begin{aligned} \bar{f}^{(\mathrm{Lagrange})}(\mathbf{g},\mathbf{L},n) &= V \times \sum_{\hat{s}_a=0}^{L_a-1}\sum_{\hat{s}_b=0}^{L_b-1}\sum_{\hat{s}_c=0}^{L_c-1} e^{-2\pi i\hat{g}_a\hat{s}_a/L_a} e^{-2\pi i\hat{g}_b\hat{s}_b/L_b} e^{-2\pi i\hat{g}_c\hat{s}_c/L_c} f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}}) \\ &= V \times \mathrm{FFT}[f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}}); \mathbf{L}] \end{aligned}$$

where

$$f^{(\mathrm{conv})}(\overleftrightarrow{\mathbf{hs}}) = \int_0^1 ds'_a \int_0^1 ds'_b \int_0^1 ds'_c \sum_{k_a} \sum_{k_b} \sum_{k_c} f(\overleftrightarrow{\mathbf{hs}}') \times W_{n,\hat{s}_a+k_aL_a}(s'_aL_a) W_{n,\hat{s}_b+k_bL_b}(s'_bL_b) W_{n,\hat{s}_c+k_cL_c}(s'_cL_c).$$

and $s_\beta = \hat{s}_\beta / L_\beta$.

Note, PME uses even function $W_{n,\hat{s}}(p)$ as the interpolation scheme, so the $n+1$ interpolation points are the closest ones to p. However $M_n(p-\hat{s})$ is not even function, so in SPME the original system is represented by a shifted mesh system, which is from a practical point of view irrelevant, because this shift is undone in the back-interpolation (if accomplished with the same assignment function).

P^{3}ME

Hockney and Eastwood [46] employ the same charge assignment scheme as PME. However, they use different Kernel function (influence function) $W^{(\mathrm{Kern})}(\mathbf{g})$ to make the final result of the mesh calculation to be as close as possible to the original continuum problem.

They first make the statement "as close as possible" more quantitative in such a way that making Q as small as possible, where Q represents the sum of the differences between the mesh-calculated forces and the true forces in terms of the 2nd norm.

$$Q \triangleq \frac{1}{V_c} \int_{V_c} \mathrm{d}\mathbf{r_1} \int_V \mathrm{d}\mathbf{r} \left[\mathbf{F}(\mathbf{r};\mathbf{r_1}) - \mathbf{R}(\mathbf{r})\right]^2 .$$

where V_c is the volume of one mesh cell, V is the volume of the simulation box.

Since the discretization error Q can be regarded as a functional of W, the optimal kernel function W_{opt} can be obtained by setting the functional derivative of Q with respect to W to zero, i.e.,

$$\frac{\delta Q}{\delta W}|_{W=W_{\text{opt}}} = 0.$$

From this idea, Hockney and Eastwood were able to derive the following expression for W_{opt}:

$$W_{\text{opt}}(\mathbf{g}) = \frac{\tilde{\mathbf{D}}(\mathbf{g}) \cdot \sum_{m \in \mathbb{Z}^3} \tilde{U}^2 \left(\mathbf{g} + \frac{2\pi}{h}\mathbf{m}\right) \tilde{\mathbf{R}} \left(\mathbf{g} + \frac{2\pi}{h}\mathbf{m}\right)}{|\tilde{\mathbf{D}}(\mathbf{g})|^2 \left[\sum_{m \in \mathbb{Z}^3} \tilde{U}^2 \left(\mathbf{g} + \frac{2\pi}{h}\mathbf{m}\right)\right]^2}$$

Here we assume each mesh cell is a cube with the length of each side equal to h. $\tilde{\mathbf{D}}(\mathbf{g})$ is the Fourier transform of the employed differentiation operator, $\tilde{U}(\mathbf{g})$ is the Fourier transform of the charge assignment function divided by the volume of one mesh cell and $\tilde{\mathbf{R}}(\mathbf{g})$ is the Fourier transform of the true reference force.

References

[1] *FFTW: An adaptive software architecture for the FFT.*, volume 3, 1998.

[2] *FFT compiler techniques*, volume 2985, 2004.

[3] *QMP: LQCD Message Passing API version 2.0*, 2004.

[4] *The design and implementation of FFTW3*, volume 93, 2005.

[5] D. J. Adams and G. S. Dubey. Taming the ewald sum in the computer-simulation of charged systems. *Journal of Computational Physics*, 72(1):156–176, 1987. J8327 Times Cited:116 Cited References Count:37.

[6] N. R. Adiga, M. A. Blumrich, D. Chen, P. Coteus, A. Gara, M. E. Giampapa, P. Heidelberger, S. Singh, B. D. Steinmacher-Burow, T. Takken, M. Tsao, and P. Vranas. Blue Gene/L torus interconnection network. *IBM. J. Res. Dev.*, 49(2-3):265–276, 2005. 925ZP Times Cited:6 Cited References Count:16.

[7] M. P. Allen. Introduction to molecular dynamics simulation. *Computational Soft Matter: From Synthetic Polymers to Proteins, Lecture Notes*, 23:1–28, 2004.

[8] M. P. Allen and D. J. Tildesley. *Computer Simulation of Liquids*. Oxford science publications. Clarendon Press; Oxford University Press, Oxford [England], 1989.

[9] A. Balaeff and J. Pitera. Simulating thermodynamics of protein folding with Blue Matter. *Biophys. J.*, 86(1):414A–415A, 2004. Part 2 Suppl. S 762KQ Times Cited:0 Cited References Count:0.

[10] P. Boyle, D. Chen, N. Christ, M. Clark, S. Cohen, C. Cristian, Z. Dong, A. Gara, B. Joo, C. Jung, C. Kim, L. Levkova, X. Liao, G. Liu, S. Li, H. Lin, R. Mawhinney, S. Ohta, K. Petrov, T. Wettig, and A. Yamaguchi. The QCDOC project. *Nucl. Phys. B-proc. sup.*, 140:169–175, 2005. 901DU Times Cited:0 Cited References Count:5.

[11] P. A. Boyle, D. Chen, N. H. Christ, M. Clark, S. D. Cohen, C. Cristian, Z. Dong, A. Gara, B. Joo, C. Jung, C. Kim, L. Levkova, X. Liao, S. Li, H. Lin, G. Liu, R. D. Mawhinney, S. Ohta, K. Petrov, T. Wettig, and A. Yamaguchi. The status of user software on QCDOC. *Nucl. Phys. B-proc. sup.*, 140:829–831, 2005. 901DU Times Cited:0 Cited References Count:2.

[12] P. A. Boyle, D. Chen, N. H. Christ, M. A. Clark, S. D. Cohen, C. Cristian, Z. Dong, A. Gara, B. Joo, C. Jung, C. Kim, L. A. Levkova, X. Liao, G. Liu, R. D. Mawhinney, S. Ohta, K. Petrov, T. Wettig, and A. Yamaguchi. Overview of the QCDSP and QCDOC computers. *IBM. J. Res. Dev.*, 49(2-3):351–365, 2005. 925ZP Times Cited:1 Cited References Count:11.

[13] R. N. Bracewell. *The Fourier transform and its applications*. McGraw-Hill series in electrical engineering. Circuits and systems. McGraw-Hill, New York, 2nd edition, 1986.

[14] E. O. Brigham. *The Fast Fourier transform*. Prentice-Hall, Englewood Cliffs, N.J.,, 1974.

[15] D. A. Case, T. E. Cheatham, T. Darden, H. Gohlke, R. Luo, K. M. Merz, A. Onufriev, C. Simmerling, B. Wang, and R. J. Woods. The Amber biomolecular simulation programs. *J. Comput. Chem.*, 26(16):1668–1688, Dec 2005.

[16] T. E. Cheatham, J. L. Miller, T. Fox, T. A. Darden, and P. A. Kollman. Molecular-Dynamics Simulations on Solvated Biomolecular Systems - the Particle Mesh Ewald Method Leads to Stable Trajectories of DNA, Rna, and Proteins. *J. Am. Chem. Soc.*, 117(14):4193–4194, 1995. Qt132 Times Cited:226 Cited References Count:23.

[17] D. Chen, N. H. Christ, C. Cristian, Z. Dong, A. Gara, K. Garg, B. Joo, C. Kim, L. Levkova, X. Liao, R. D. Mawhinney, S. Ohta, and T. Wettig. QCDOC: A 10-teraflops scale computer for lattice QCD. *Nucl. Phys. B-proc. sup.*, 94:825–832, 2001. 417CZ Times Cited:8 Cited References Count:5.

[18] X. Cheng, C. Kelso, V. Hornak, C. de los Santos, A. P. Grollman, and C. Simmerling. Dynamic behavior of dna base pairs containing 8-oxoguanine. *J Am Chem Soc*, 127(40):13906–13918, Oct 2005.

[19] C. K. Chui. *An introduction to wavelets*. Wavelet analysis and its applications; v. 1. Academic Press, Boston, 1992.

[20] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz, D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules (vol 117, pg 5179, 1995). *J. Am. Chem. Soc.*, 118(9):2309–2309, 1996. Ty698 Times Cited:68 Cited References Count:1.

[21] C. E. Cramer and J. A. Board. The development and integration of a distributed 3D FFT for a cluster of workstations. In *4th annual Linux Showcase and Conference*, page 121, 2000.

[22] T. Darden, D. York, and L. Pedersen. Particle Mesh Ewald - an N.Log(N) Method for Ewald Sums in Large Systems. *J. Chem. Phys.*, 98(12):10089–10092, 1993. Lg101 Times Cited:1354 Cited References Count:23.

[23] Y. Deng, J. Glimm, J. W. Davenport, X. Cai, and E. Santos. Performance models on QCDOC for molecular dynamics with Coulomb potentials. *Int. J. High. Perform. C.*, 18(2):183–195, 2004. 827CD Times Cited:0 Cited References Count:21.

[24] M. Deserno and C. Holm. How to mesh up Ewald sums. I. A theoretical and numerical comparison of various particle mesh routines. *J. Chem. Phys.*, 109(18):7678–7693, 1998. 132ZZ Times Cited:84 Cited References Count:19.

[25] M. Deserno and C. Holm. How to mesh up Ewald sums. II. An accurate error estimate for the particle-particle-particle-mesh algorithm. *J. Chem. Phys.*, 109(18):7694–7701, 1998. 132ZZ Times Cited:39 Cited References Count:10.

[26] A. Edelman, P. McCorquodale, and S. Toledo. The future fast Fourier transform? *SIAM. J. Sci. Comp.*, 20(3):1094–1114, 1999. 169UN Times Cited:3 Cited References Count:23.

[27] M. Eleftheriou, B. G. Fitch, A. Rayshubskiy, T. J. C. Ward, and R. S. Germain. Scalable framework for 3D FFTs on the Blue Gene/L supercomputer: Implementation and early performance measurements. *IBM. J. Res. Dev.*, 49(2-3):457–464, 2005. 925ZP Times Cited:2 Cited References Count:16.

[28] M. Eleftheriou, J. E. Moreira, B. G. Fitch, and R. S. Germain. A volumetric FFT for BlueGene/L. *High Performance Computing - Hipc 2003*, 2913:194–203, 2003. By20R Times Cited:0 Cited References Count:12 Lecture Notes in Computer Science.

[29] U. Essmann, L. Perera, M. L. Berkowitz, T. Darden, H. Lee, and L. G. Pedersen. A Smooth Particle Mesh Ewald Method. *J. Chem. Phys.*, 103(19):8577–8593, 1995. Te364 Times Cited:1166 Cited References Count:67.

[30] P. P. Ewald. Die berechnung optischer und elektrostatischer gitterpotentiale. *Ann. Phys.*, 64:253–287, 1921.

[31] B. Fang, Y. Deng, and G. Martyna. Performance of 3D FFT on 6D QCDOC Torus Parallel Supercomputer. Comput. Phys. Commun., accepted 12/2006.

[32] B. Fang, G. Martyna, and Y. Deng. A Fine Grained Parallel Smooth Particle Mesh Ewald Algorithm for Biophysical Simulation Studies : Application to the 6-D torus QCDOC supercomputer. Comput. Phys. Commun., accepted 01/2007.

[33] S. E. Feller, R. W. Pastor, A. Rojnuckarin, S. Bogusz, and B. R. Brooks. Effect of electrostatic force truncation on interfacial and transport properties of water. *Journal of Physical Chemistry*, 100(42):17011–17020, 1996. Vn802 Times Cited:133 Cited References Count:52.

[34] B. G. Fitch, R. S. Germain, M. Mendell, J. Pitera, M. Pitman, A. Rayshubskiy, Y. Sham, F. Suits, W. Swope, T. J. C. Ward, Y. Zhestkov, and R. Zhou. Blue Matter, an application framework for molecular simulation on Blue Gene. *J. Parallel. Distr. Com.*, 63(7-8):759–773, 2003. 728HQ Times Cited:7 Cited References Count:29.

[35] B. G. Fitch, A. Rayshubskiy, M. Eleftheriou, T. J. C. Ward, M. Giampapa, M. C. Pitman, and R. S. Germain. Blue matter: Approaching the limits of concurrency for classical molecular dynamics. Technical report, IBM Research Division, Thomas J. Watson Research Center, 2006.

[36] D. Frenkel and B. Smit. *Understanding molecular simulation: from algorithms to applications*. Academic Press, San Diego, Calif.; London, UK, [2nd edition, 2002.

[37] M. Frigo and S. G. Johnson. The Fastest Fourier transform in the west. Technical Report MIT-LCS-TR-728, MIT, 1997.

[38] G. Galli and M. Parrinello. *Computer Simulation in Materials Science: Interatomic Potentials, Simulation Techniques and Applications*, volume 3, chapter The Ab-initio MD method, pages 283–304. Springer, 1991.

[39] A. Gara, M. A. Blumrich, D. Chen, G. L. T. Chiu, P. Coteus, M. E. Giampapa, R. A. Haring, P. Heidelberger, D. Hoenicke, G. V. Kopcsay, T. A. Liebsch, M. Ohmacht, B. D. Steinmacher-Burow, T. Takken, and P. Vranas. Overview of the Blue Gene/L system architecture. *IBM. J. Res. Dev.*, 49(2-3):195–212, 2005. 925ZP Times Cited:3 Cited References Count:25.

[40] R. S. Germain, B. Fitch, A. Rayshubskiy, M. Eleftheriou, M. C. Pitman, F. Suits, M. Giampapa, and T. C. Ward. Blue matter on blue gene/l: massively parallel computation for biomolecular simulation. In *CODES+ISSS '05: Proceedings of the 3rd IEEE/ACM/IFIP international conference on Hardware/software codesign and system synthesis*, pages 207–212, 2005.

[41] R. S. Germain, Y. Zhestkov, M. Eleftheriou, A. Rayshubskiy, F. Suits, T. J. C. Ward, and B. G. Fitch. Early performance data on the Blue Matter molecular simulation framework. *IBM. J. Res. Dev.*, 49(2-3):447–455, 2005. 925ZP Times Cited:4 Cited References Count:18.

[42] H. Grubmüller, H. Heller, A. Windemuth, and K. Schulten. Generalized verlet algorithm for efficient molecular dynamics simulations with long-range interactions. *Mol. Sim.*, 6:121–142, 1991.

[43] G. Han. *Analysis of Multi-scale Molecular Dynamics Algorithms for Simulating Biomolecules*. PhD thesis, Stony Brook University, 2006.

[44] J. Hansen. *Molecular dynamics simulation of Coulomb systems in Molecular dynamics simulation of statistical mechanical systems.* North Holland Physics, 1986.

[45] P. D. Haynes and M. Cote. Parallel fast Fourier transforms for electronic structure calculations. *Comput. Phys. Commun.*, 130(1-2):130–136, 2000. 338UK Times Cited:2 Cited References Count:7.

[46] R. W. Hockney and J. W. Eastwood. *Computer Simulation Using Particles*. McGraw-Hill International Book Co., New York, 1981.

[47] V. Hornak, A. Okur, R. C. Rizzo, and C. Simmerling. Hiv-1 protease flaps spontaneously close to the correct structure in simulations following manual placement of an inhibitor into the open state. *J Am Chem Soc*, 128(9):2812–2813, Mar 2006.

[48] V. Hornak, A. Okur, R. C. Rizzo, and C. Simmerling. Hiv-1 protease flaps spontaneously open and reclose in molecular dynamics simulations. *Proc Natl Acad Sci U S A*, 103(4):915–920, Jan 2006.

[49] L. V. Kale and S. Krishnan. Charm++: a portable concurrent object oriented system based on c++. In *OOPSLA '93: Proceedings of the eighth annual conference on Object-oriented programming systems, languages, and applications*, pages 91–108, New York, NY, USA, 1993. ACM Press.

[50] V. Kumar. *Introduction to parallel computing: design and analysis of algorithms*. Benjamin/Cummings Pub. Co., Redwood City, Calif., 1994.

[51] P. Lancaster and M. Tismenetsky. *The Theory of Matrices: with applications*. Academic Press, Orlando, 2nd edition, 1985.

[52] E. S. Lander, L. M. Linton, B. Birren, C. Nusbaum, M. C. Zody, J. Baldwin, K. Devon, K. Dewar, M. Doyle, W. FitzHugh, R. Funke, D. Gage, K. Harris, A. Heaford, J. Howland, L. Kann, J. Lehoczky, R. LeVine, P. McEwan, K. McKernan, J. Meldrim, J. P. Mesirov, C. Miranda, W. Morris, J. Naylor, C. Raymond, M. Rosetti, R. Santos, A. Sheridan, C. Sougnez, N. Stange-Thomann, N. Stojanovic, A. Subramanian, D. Wyman, J. Rogers, J. Sulston, R. Ainscough, S. Beck, D. Bentley, J. Burton, C. Clee, N. Carter, A. Coulson, R. Deadman, P. Deloukas, A. Dunham, I. Dunham, R. Durbin, L. French, D. Grafham, S. Gregory, T. Hubbard, S. Humphray, A. Hunt, M. Jones, C. Lloyd, A. McMurray, L. Matthews, S. Mercer, S. Milne, J. C. Mullikin, A. Mungall, R. Plumb, M. Ross, R. Shownkeen, S. Sims, R. H. Waterston, R. K. Wilson, L. W. Hillier, J. D. McPherson, M. A. Marra, E. R. Mardis, L. A. Fulton, A. T. Chinwalla, K. H. Pepin, W. R. Gish, S. L. Chissoe, M. C. Wendl, K. D. Delehaunty, T. L. Miner, A. Delehaunty, J. B. Kramer, L. L. Cook, R. S. Fulton, D. L. Johnson, P. J. Minx, S. W. Clifton, T. Hawkins, E. Branscomb, P. Predki, P. Richardson, S. Wenning, T. Slezak, N. Doggett, J. F. Cheng, A. Olsen, S. Lucas, C. Elkin, E. Uberbacher, M. Frazier, et al. Initial sequencing and analysis of the human genome. *Nature*, 409(6822):860–921, 2001. 401QC Times Cited:5272 Cited References Count:450.

[53] T. Lippert. Recent development and perspectives of machines for lattice QCD. *Nucl. Phys. B-proc. sup.*, 129-30:88–101, 2004. 807BX Times Cited:2 Cited References Count:52.

[54] R. J. Loncharich and B. R. Brooks. The effects of truncating long-range forces on protein dynamics. *Proteins.*, 6(1):32–45, 1989. Av417 Times Cited:187 Cited References Count:20.

[55] B. A. Luty, M. E. Davis, I. G. Tironi, and W. F. Vangunsteren. A Comparison of Particle-Particle, Particle-Mesh and Ewald Methods for Calculating Electrostatic Interactions in Periodic Molecular-Systems. *Mol. Simul.*, 14(1):11–20, 1994. Qf464 Times Cited:48 Cited References Count:20.

[56] A. D. MacKerell, D. Bashford, M. Bellott, R. L. Dunbrack, J. D. Evanseck, M. J. Field, S. Fischer, J. Gao, H. Guo, S. Ha, D. Joseph-McCarthy, L. Kuchnir, K. Kuczera, F. T. K. Lau, C. Mattos, S. Michnick, T. Ngo, D. T. Nguyen, B. Prodhom, W. E. Reiher, B. Roux, M. Schlenkrich, J. C. Smith, R. Stote, J. Straub, M. Watanabe, J. Wiorkiewicz-Kuczera, D. Yin, and M. Karplus. All-atom empirical potential for molecular modeling and dynamics studies of proteins. *J. Phys. Chem. B*, 102(18):3586–3616, 1998. Zn306 Times Cited:1634 Cited References Count:128.

[57] G. J. Martyna, A. Hughes, and M. E. Tuckerman. Molecular dynamics algorithms for path integrals at constant pressure. *J. Chem. Phys.*, 110(7):3275–3290, 1999. 164DA Times Cited:26 Cited References Count:47.

[58] G. J. Martyna and M. E. Tuckerman. A reciprocal space based method for treating long range interactions in ab initio and force-field-based calculations in clusters. *J. Chem. Phys.*, 110(6):2810–2821, 1999. 163AJ Times Cited:91 Cited References Count:36.

[59] P. Minary, M. E. Tuckerman, and G. J. Martyna. Long time molecular dynamics for enhanced conformational sampling in biomolecular systems. *Phys. Rev. Lett.*, 93(15):150201.1–150201.4, 2004. 860KD Times Cited:5 Cited References Count:45.

[60] J. Norberg and L. Nilsson. On the truncation of long-range electrostatic interactions in dna. *Biophys J*, 79(3):1537–1553, Sep 2000.

[61] A. Okur, B. Strockbine, V. Hornak, and C. Simmerling. Using pc clusters to evaluate the transferability of molecular mechanics force fields for proteins. *J Comput Chem*, 24(1):21–31, Jan 2003.

[62] J. W. Perram, H. G. Petersen, and S. W. Deleeuw. An algorithm for the simulation of condensed matter which grows as the 3/2 power of the number of particles. *Molecular Physics*, 65(4):875–893, 1988. R5253 Times Cited:83 Cited References Count:10.

[63] H. G. Petersen. Accuracy and efficiency of the particle mesh ewald method. *J. Chem. Phys.*, 103(9):3668–3679, 1995. Rt923 Times Cited:65 Cited References Count:22.

[64] J. C. Phillips, R. Braun, W. Wang, J. Gumbart, E. Tajkhorshid, E. Villa, C. Chipot, R. D. Skeel, L. Kal, and K. Schulten. Scalable molecular dynamics with namd. *J. Comput. Chem.*, 26(16):1781–1802, December 2005.

[65] J. C. Phillips, G. Zheng, S. Kumar, and L. V. Kalé;. Namd: biomolecular simulation on thousands of processors. In *Supercomputing '02: Proceedings of the 2002 ACM/IEEE conference on Supercomputing*, pages 1–18, Los Alamitos, CA, USA, 2002. IEEE Computer Society Press.

[66] S. Plimpton. Fast Parallel Algorithms for Short-Range Molecular-Dynamics. *J. Comput. Phys*, 117(1):1–19, 1995. Qk448 Times Cited:251 Cited References Count:56.

[67] E. L. Pollock and J. Glosli. Comments on p(3)m, fmm, and the ewald method for large periodic coulombic systems. *Comput. Phys. Commun.*, 95(2-3):93–110, 1996. Uq973 Times Cited:62 Cited References Count:18.

[68] P. Procacci, M. Marchi, and G. J. Martyna. Electrostatic calculations and multiple time scales in molecular dynamics simulation of flexible molecular systems. *J. Chem. Phys.*, 108(21):8799–8803, 1998. 108FR Times Cited:25 Cited References Count:15.

[69] P. Rissland. *MDoC: A Molecular Dynamics application framework for the QC-DOC supercomputer.* PhD thesis, Stony Brook University, 2006.

[70] D. R. Roe, V. Hornak, and C. Simmerling. Folding cooperativity in a three-stranded beta-sheet model. *J Mol Biol*, 352(2):370–381, Sep 2005.

[71] M. J. L. Sangster and M. Dixon. Interionic potentials in alkali-halides and their use in simulations of molten-salts. *Advances in Physics*, 25(3):247–342, 1976. Ca195 Times Cited:456 Cited References Count:169.

[72] I. J. Schoenberg. *Cardinal Spline Interpolation*. Regional conference series in applied mathematics, 12. Society for Industrial and Applied Mathematics, Philadelphia,, 1973.

[73] A. Y. Toukmaji and J. A. Board. Ewald summation techniques in perspective: A survey. *Comput. Phys. Commun.*, 95(2-3):73–92, 1996. Uq973 Times Cited:76 Cited References Count:54.

[74] M. Tuckerman, B. J. Berne, and G. J. Martyna. Reversible multiple time scale molecular-dynamics. *J. Chem. Phys.*, 97(3):1990–2001, 1992. Je891 Times Cited:725 Cited References Count:19.

[75] R. V. Vadali, Y. Shi, S. Kumar, L. V. Kale, M. E. Tuckerman, and G. J. Martyna. Scalable fine-grained parallelization of plane-wave-based ab initio molecular dynamics for large supercomputers. *J. Comput. Chem.*, 25(16):2006–2022, 2004. 868OI Times Cited:3 Cited References Count:70.

[76] C. Van Loan, S. for Industrial, and A. Mathematics. *Computational Frameworks for the Fast Fourier Transform*. SIAM, Philadelphia, 1992.

[77] J. C. Venter, M. D. Adams, E. W. Myers, P. W. Li, R. J. Mural, G. G. Sutton, H. O. Smith, M. Yandell, C. A. Evans, R. A. Holt, J. D. Gocayne, P. Amanatides, R. M. Ballew, D. H. Huson, J. R. Wortman, Q. Zhang, C. D. Kodira, X. Q. H. Zheng, L. Chen, M. Skupski, G. Subramanian, P. D. Thomas, J. H. Zhang, G. L. G. Miklos, C. Nelson, S. Broder, A. G. Clark, C. Nadeau, V. A. McKusick, N. Zinder, A. J. Levine, R. J. Roberts, M. Simon, C. Slayman, M. Hunkapiller, R. Bolanos, A. Delcher, I. Dew, D. Fasulo, M. Flanigan, L. Florea, A. Halpern, S. Hannenhalli, S. Kravitz, S. Levy, C. Mobarry, K. Reinert, K. Remington, J. Abu-Threideh, E. Beasley, K. Biddick, V. Bonazzi, R. Brandon, M. Cargill, I. Chandramouliswaran, R. Charlab, K. Chaturvedi, Z. M. Deng, V. Di Francesco, P. Dunn, K. Eilbeck, C. Evangelista, A. E. Gabrielian, W. Gan, W. M. Ge, F. C. Gong, Z. P. Gu, P. Guan, T. J. Heiman, M. E. Higgins, R. R. Ji, Z. X. Ke, K. A. Ketchum, Z. W. Lai, Y. D. Lei, Z. Y. Li, J. Y. Li, Y. Liang, X. Y. Lin, F. Lu, G. V. Merkulov, N. Milshina, H. M. Moore, A. K. Naik, V. A. Narayan, B. Neelam, D. Nusskern, D. B. Rusch, S. Salzberg, W. Shao, B. X. Shue, J. T. Sun, Z. Y. Wang, A. H. Wang, X. Wang, J. Wang, M. H. Wei, R. Wides, C. L. Xiao, C. H. Yan, et al. The sequence of the human genome. *Science*, 291(5507):1304–1351, 2001. 402MX Times Cited:3755 Cited References Count:192.

[78] P. K. Weiner and P. A. Kollman. Amber - assisted model-building with energy refinement - a general program for modeling molecules and their interactions. *J. Comput. Chem.*, 2(3):287–303, 1981. Mb222 Times Cited:814 Cited References Count:42.

[79] D. A. Yarne, M. E. Tuckerman, and G. J. Martyna. A dual length scale method for plane-wave-based, simulation studies of chemical systems modeled using mixed ab initio/empirical force field descriptions. *J. Chem. Phys.*, 115(8):3531–3539, 2001. 462AP Times Cited:14 Cited References Count:28.

[80] E. L. Zapata, F. F. Rivera, I. Benavides, J. M. Carazo, and R. Peskin. Multidimensional Fast Fourier-Transform into Simd Hypercubes. *IEE. Proc-E*, 137(4):253–260, 1990. Dq028 Times Cited:5 Cited References Count:30.

[81] R. H. Zhou, E. Harder, H. F. Xu, and B. J. Berne. Efficient multiple time step method for use with Ewald and particle mesh Ewald for large biomolecular systems. *J. Chem. Phys.*, 115(5):2348–2358, 2001. 453QQ Times Cited:29 Cited References Count:29.