Performance Comparison Between AMOEBA (Polarizable) and AMBER (Static Charge) Force Fields

Yong Hwan Kim, Salvador Aguiñaga, Yiji Zhang
Department of Computer Science and Engineering
University of Notre Dame · Notre Dame, Indiana, USA
{*ykim15, saguinag, yzhang20*}*@nd.edu*

***Abstract*—AMOEBA is a polarizable atomic multipole force field recently developed in molecular dynamics to more accurately calculate descriptive trajectories at the atomic level. This model is believed to enable more accurate description of the molecular properties, but is computationally expensive and many researchers lean towards the use of static-charge force field models. In this work, we characterize complexity and computational cost of AMOEBA's simulation efficiency and compare the results to simulations using AMBER, a static charge model. We focus on one central question: how do system resources contribute to simulation efficiency? The metrics used to analyze system effects included simulation efficiency, virtual memory utilization, and network traffic. Our results suggest that a balance between system resources and the molecular dynamics simulation system size (i.e. molecule type and number of atoms) must be achieved, otherwise it is not realistic to conduct MD simulations using the AMOEBA force field whose extremely slow performance outweighs its value of accurate calculations.**

***Keywords*-molecular dynamics, force field, operating system, cluster computing, AMBER, AMOEBA**

I. Introduction

Molecular dynamics (MD), a computer simulation of complex systems, is now an accepted and integral part of contemporary biochemistry, biophysics, and materials science to study the behavior of atoms or their physical movement at the molecular level [1]. MD simulation is useful for examining protein-folding pathways, which may help scientists understand the nature of the odd behavior of proteins sometimes progressing to disease. This understanding is enabled by the structural, kinetic, and thermodynamic calculations within MD simulations towards providing insight and new biological hypotheses for these mechanisms. Specifically, those that can be experimentally tested and used to further enhance the models iteratively.

A great shift in biological research has led to the emergence of high-throughput technologies for measuring different aspects of biological mechanisms in a cell. The development of techniques for measuring protein-protein interactions (PPI) is an example of this technological shift. Many large-scale assays measuring PPIs in a variety of organisms (i.e. bacteria, yeast, worm, etc.) has lead to an explosion of data that is being assessed and accumulated in databases such as DIP and BioGRID [2]. The problem is that data remains noisy and incomplete. MD simulations is a technique well suited to help denoise and fill in the gaps, thus extending the knowledge base.

There are two aspects regarding the evaluation of a MD simulation, accuracy and performance. Simulation efficient is a function of CPU performance and various factors influence it from a biochemical perspective. Our work considers the following factors: force field, system size, and molecule type (benchmark). *System size* refers to the molecule (in terms of number of atoms, see Table II) plus its surrounding (explicit) solvent or the lack of (implicit). In our study, proteins are our molecules of interest and their surrounding solvent is water molecules. The second factor we consider in our study is protein type. Different proteins lead to different performance in simulation depending on their structure. Figure 1 shows a diagram of of the input parameters to our simulation setup. In MD, a ***force field*** indicates the form and parameters of mathematical functions used to describe the potential energy of a system of atoms or molecules [3]. A trend-change in force field choice has taken place over the past several years [4]. Some researchers have began to shift their focus from well-established and well-tuned, but intrinsically limited, fixed point charge models to more complicated and computationally expensive polarizable models that allow a more accurate description of molecular properties [4]. AMBER (Assisted Model Building and Energy Refinement) is one of the most popular non-polarizable force fields used today. AMOEBA (Atomic Multipole Optimized Energetics for Bio-molecular Applications) is a polarized force field model that has been showing stable performance, but has not been extensively explored.

Problem Statement. Our study focuses on is how computationally expensive is the use of the AMOEBA model as compared to using the AMBER force field? To study the difference between these two force fields, we measured the effect of the AMOEBA and AMBER on the performance of different types of simulations and eventually on how each force field calculates and generates results that can answer specific scientific questions. The results, analyzed in Section III, show how MD simulations with AMOEBA compare to those with AMBER.

Our Contribution. This work contributes to the field of MD by characterizing the simulation efficiency of **AMOEBA**

force field, a promising model designed to more accurately describe the interactions at the atomic level, on a range of typical systems (molecule types) using threaded and parallelized computational environments. Because AMOEBA has not been fully benchmarked or thoroughly analyzed, our contribution focuses on how the AMOEBA simulation efficiency compares to the more established and thoroughly tested AMBER force field. Our aim is to gain a better understanding of how AMOEBA simulation efficiency stacks up to AMBER's. These results will help us or others in the field to investigate and explore further (in the near future) how to optimize hardware and software system resources to encourage the use of polarizable force field models to be used on a more regular basis.

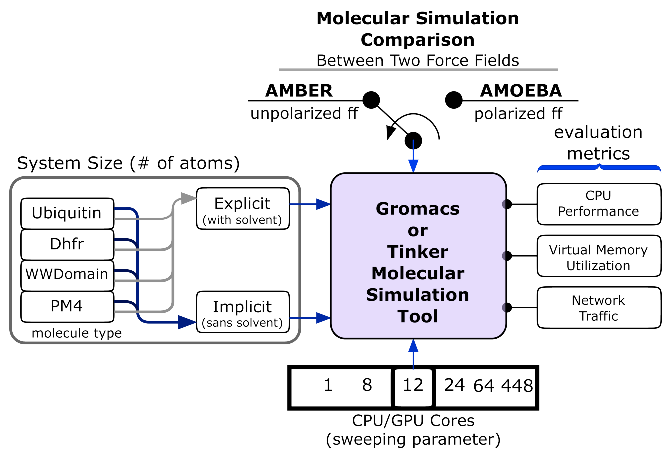


Figure 1: Molecular simulation design and configuration setup

Related Work. Several publications have discussed MD simulations using AMOEBA force field [5] [6]. Pande group at Stanford University is one of the groups that has done many examinations for simulations using polarizable force field. Recently, Ponder et al. provided the current status of AMOEBA force field compared to fixed charge models for small molecules [4]. According to this paper, the researchers showed that AMOEBA is especially successful on the interactions between protein and ligand and computational X-ray crystallography where polarization and accurate electrostatics are crucial. Although this paper examines the various aspects of the performance of AMOEBA force field, this paper did not directly compare the performance of AMOEBA force field to that of other regular force fields such as AMBER during the MD simulations on the protein.

MD simulation on GPU systems has been under active study [7], [8]. Work in [9] explored the best conditions where each MD simulations can perform the best using GPU and CPU. However, MD simulations using polarizable force field were not examined in this work. In their work, they did not study memory usage, power consumption, nor network performance.

A few years after the paper on the stability of AMOEBA force field was published, a group of researchers at Simbios, a NIH center at Stanford, did further work on simulations using AMOEBA force field by running several benchmarks on proteins using AMOEBA force fields on GPUs and CPUs [10]. Our work will be different in that we show the relationship between the number of cores and scalability. Also, this benchmark used two different MD packages which are openMM and TINKER to explore AMOEBA force fields. That may not result in major difference but we cannot rule out the possibility that the results might have been influenced by using two different MD software packages. In addition, the work at Simbios compared performance, but did not explain the details behind the simulations nor provided the computing-system details. Our study will use TINKER for simulations using AMOEBA force field in both GPUs and CPUs.

II. Methods

A. Experimental Design

We consider three factors of MD simulation efficiency, which are system size, force field, and hardware configuration. We carried out a set of experiments (molecular simulations) to explore how each of these factor affects the simulation efficiency. The experiment design is shown in Figure 1 and Table I includes the different parameters considered for each factor. The goal of experiment 1 is

Table I: Experiment designs

Exp.	Fixed Factor	Changed Factor
Simulation Duration: 20ps		
1	AMOEBA; machine	system size (implicit/explicit)
2	machine; system size	force field (AMOEBA/ AMBER)
3	AMOEBA; system size	machine

to evaluate MD simulation efficiency of different system sizes with AMOEBA. As introduced in Section I, a *system* consists of a protein and its surrounding solvent. To achieve the goal, we conducted experiments on CPU and GPU machines respectively. On each machine, we run simulations with AMOEBA on each machine of two different system sizes, a bigger one with water solvent, and another one with non-water solvent. The protein molecules used are show in Table II.

The second goal of our work, which is corresponding to experiment 2 is to compare the performance of MD simulation with AMOEBA and AMBER running on the same machine with the same MD designs. AMBER, one of the most popular force fields, is a fixed charge model and thus computationally cheap. While there are many review

Table II: System Size

Biological Molecule	Implicit Solvent Number of atoms	Explicit Solvent Number of Atoms
Alanine dipeptide	13	Unavailable
WWdomain	537	4034
Proinsulin	1293	18738
Plasmepsin 4	5157	31769
Ubiquitin	1692	9374
DHFR	2557	24805

articles comparing different non-polarizable models, different effects between computationally expensive polarizable force field and non-polarizable force field in MD have not been studied to the best of our knowledge. To investigate the performance, we ran MD simulations on CPU and GPU machines respectively, using the number of cores from 2 to 10 under the same MD design (with solvent or without solvent).

The third experiment is to investigate how different processor and grid system affect MD simulations performance using AMOEBA. GPU with its highly parallel structure performs more effective than general-purpose CPU, while the time and power efficiency per core has not been studied when running MD simulations with AMOEBA. We measured performance information for simulations running on CPU and GPU, compared and analyzed the runtime and memory utilization per core. We will also study the network flow information as future work. The next section specifies the hardware and software configurations used.

B. System configuration

Hardware configuration. The hardware configuration of the experiment is as follows:

- The University of Notre Dame Center for Research Computing (CRC) provided a cluster of CPU nodes for this research project. Jobs were submitted through the Sun Grid Engine system. These nodes typically have: four 6-core Intel Xeon E7540 processors operating at 2.00 GHz and 128GB of RAM.
- CRC provided a set of nodes with GPU cores. These nodes typically have two NVIDIA Tesla M2050 graphics cards, each with 2.5GB of RAM, 448 cores, and a clock rate of 1.15 GHz, two 6-core Intel Xeon X5650 processors running at 2.67 GHz, and 48GB of RAM.

This work leveraged CRCs computing resources. Their machine systems are available for 'fair share,' which means there was no guarantee that all of the resources would be available any given project at any time. Multi-node jobs were stuck in a long job queue for days. Their facility encountered power failures and all jobs had to be cancelled and resubmitted.

Software configuration. In this study we utilized two simulation software packages: Gromacs (www.gromacs.org) and TINKER (dasher.wustl.edu/tinker). Table III summarizes both simulation and profiling software packages used.

Table III: Software Packages and Tools Used

Software Tool	Description and Notes
Gromacs	CRC's threaded version used primarily for use with the AMBER forcefield We compiled a MPI-enabled version and made it available as a module
TINKER	Primarily used for the AMOEBA MD simulations
TotalView	A profiling package, provided by CRC to debug memory
qsub	A scheduling tool to run simulations on single and cluster nodes
Xymon	A tool for monitoring servers, applications and networks

C. Metrics

We use three metrics to assess the performance, which are simulation rate, memory utilization, and network traffic of the grid computation system. The techniques that is used to evaluate MD simulations with AMOEBA and AMBER against these metrics are outlined in the following subsections.

Simulation Efficiency. *Simulation efficiency* is commonly measured in nanosecond per day (ns/day). The Ns/day unit refers to the physical movements of atoms and molecules computed by simulation takes one day while in reality it takes N nanoseconds by the actual molecule. Higher simulation efficiency indicates better performance.

We extract simulation efficiency information from the log file generated by the simulation software. We expect to see a weak linear relationship between the number of cores and the performance for all the simulations. For the simulations with variations in system size, we expect to see a weak linear relationship in MD simulations using AMOEBA. Lastly, we expect to see a linear relationship in MD simulation using AMBER force field.

Memory Utilization. The second metric we look at for evaluation is the *total virtual memory* utilization. We expect to see larger memory utilization when we use a larger size system, which means we experiment with solvated molecules. Keeping constraints and non-bonded interactions might affect the computational cost. Memory utilization is obtained in the following ways:

1) Submitting jobs via **`qsub`** allowed us to obtain Maximum Virtual Memory (max_vmem)
2) We extracted memory utilization using **xymon**

Network Traffic. Interprocessor communication is another measure of performance. In this work we measure network communication as a function of system size, force field model, and number of cores during simulation.

III. Results

A. Simulation efficiency

We extract data on simulation efficiency in ***ns/day*** with respect to the number of cores and system size.

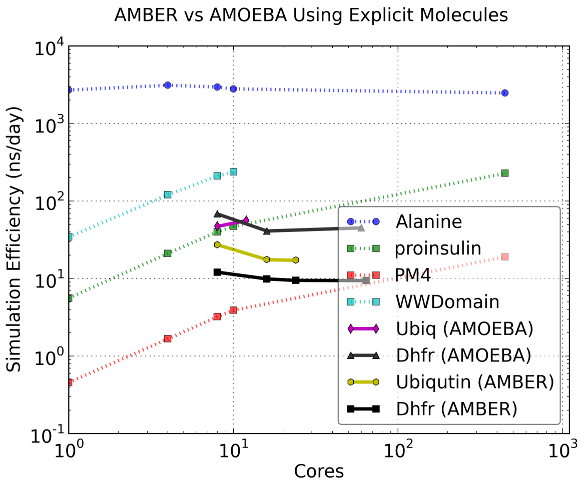
Table IV: Comprehensive system-level comparison between AMOEBA and AMBER MD simulations

AMOEBA	AMBER
Simulation Duration: 20ps	
40 CPU cores	
Software: TINKER	
System size: number of atoms of explicit Dhfr model	
Simulation runtime: 10 hours	1.67 hours
Avr VMem: 7.92 G	7.392 G
Network Traffic: 7.06 Mbits/s	31.08 Mbit/s

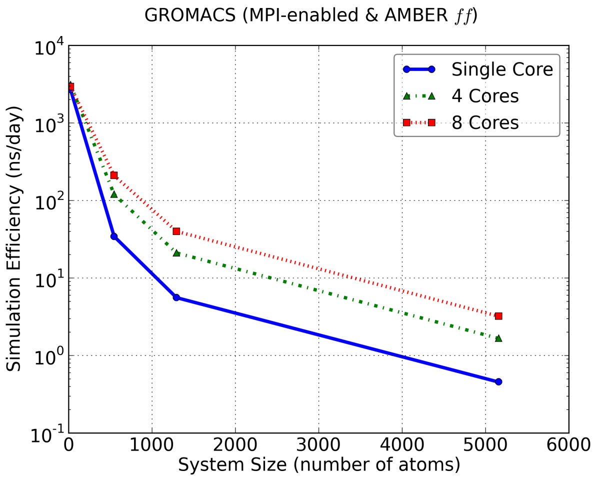
1) Number of Cores: Figure 2a and 2b indicates that using more cores leads to higher simulation efficiency. An exception is that, for Alanine, the performance decreases while using more cores. This is a reasonable and expected result because the molecule is too small, and I/O bottleneck makes the parallelization useless, even harmful, to small size jobs.

When parallelizing the job, the number of I/O operations increases and therefore the amount of time of I/O grows. The time consumed on I/O can not be discarded because the speed of I/O, being limited by the speed of mechanical components like disks and arms, has not been increasing at a same rate as the speed of CPU, memory and bus in recent years, and it is a key limiting factor in the performance of the system. Therefore, I/O-bound problem becomes the major problem in the distributed system especially when processing small jobs, because there is much less computation time in this case, which makes the I/O time weights more and has negative influence on the performance of the computation.

Another observation is that the performance of small size job degrades more badly using threaded version GROMACS than using MPI version. The reason is that the threaded GROMACS parallelizes the job in a way that every part of the job on each core is independent with each other, and there is no core-to-core network traffic . So the main core needs to first separate the jobs into independent components and sends to each core. In this way, the time consumed for separating tasks is not helpful but a waste for small size jobs which do not need to be parallelized. Whereas in MPI-enabled GROMACS, network traffic is allowed so that each core can communicate with other cores if necessary, and the time consumed on separating and assigning the job onto each individual core is therefore less, which makes MPI-enabled GROMACS more efficient than threaded GROMACS.



(a) Simulation Efficiency vs CPU Cores



(b) Simulation Efficiency vs System Size

Figure 2: Simulation efficiency as a function of cores, top figure, shows data for Ubiqutin and Dhfr obtained using both forcefields.

2) System Size: Figure 2b shows that the simulation efficiency is higher for smaller system sizes, which is the number of atoms for all four benchmark molecules. This is an expected results because the larger the system, the heavier the job, and the more time is needed to process the job. And we expect the same results for simulations using implicit solvent molecules.

The increasing runtime with respect to larger system is an obvious intuition and we saw that the degree of runtime increase when we used dramatically more cores. For GPU simulations in non-polarizable forcefield, it showed better performance up to twelve times faster than that of simulations using eight cores in CPU. Improvement in performance was more pronounced in large molecules such as proinsulin and PM4.

B. Memory Utilization

In this article, we present our preliminary results on memory utilization. Priority access in CRC resources has limited simulation runtime. Figure 3 shows maximum virtual memory utilized for the entire duration of each simulation. The profiler employed was TotalView, a highly scalable debugger available from CRC.

Figure 3 illustrates two facts, which are simulations with larger system size (total size of molecular and solvent) and/or using more cores in the distributed system require more memory. These are general rules for regular computation. But one interesting observation is that .

Besides the hardware resource balance issue, the results also indicate that in order to ensure the performance of the simulation, we have to allocate enough memory space to MD simulations. For AMOEBA, we suggest to allocate around 10 times more memory space compared to AMBER in each case (molecular and number of CPU cores) to ensure the performance, because the trend shows that the virtual memory used is a factor of the system size when using certain number of cores.

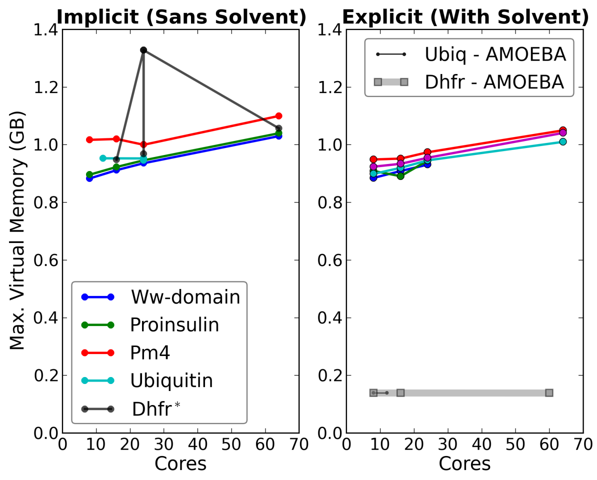


Figure 3: Virtual memory utilization as a function of system size and number of CPU cores

C. Network Traffic

We measured inbound and outbound network traffic in Mb/s using profiler Xymon. Figure 4, 5, 6, and 7 show the inbound and outbound network traffic using implicit and explicit model with AMBER, and there are Three observations. First, the trends of inbound and outbound network traffic is the same for the same model. This indicates that MD simulation is a balanced computation in terms of inbound and outbound traffic, and the workload does not change this fact. Therefore, network balance is not an issue in the event of MD simulation.

The second observation is that implicit model used less network traffic than explicit model. This is an expected result because explicit model has larger size than implicit model because of the solvent, and thus it requires more computation and more network traffic. Beyong this intuition, we can get a conclusion that network traffic may be the bottleneck of the performance as the size of the molecule increases.

The third fact is that the network traffic will reduce with more CPU cores if the size of the molecule is less than a certain degree. And using more cores will increase the network traffic in two circumstances, 1) the molecule is too small, or 2) the molecule is too large. Network traffic does not reflect simulation performance directly. And the difference between these two cases is that distributed computing harms the total performance in the former one and it actually helps the later case. Therefore, network traffic is one way to judge whether using more cores will help or harm the simulation performance.

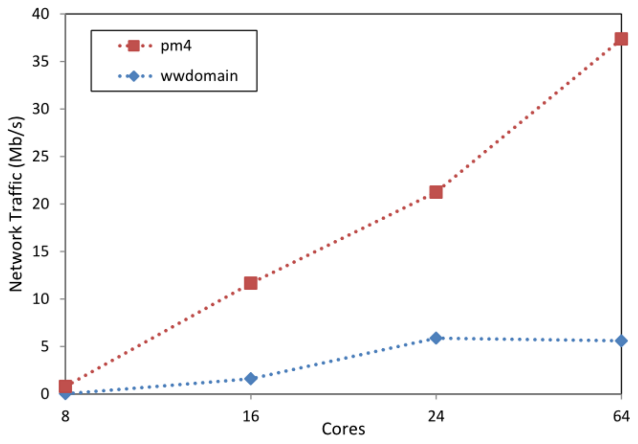


Figure 4: Inbound network traffic using impllicit model as a function of number of CPU cores

IV. Discussion and Conclussions

As we have demonstrated, we get higher simulation efficiency as we use more resources but that may not always happen which suggests that there is a specific condition that a simulation can perform its best. Our results confirm that the cost of AMOEBA forcefield is significant but we can improve simulation at the system level by finding its best condition to perform. In our tests, we had difficulty building a model for TINKER simulation due to lack of help availability. Also, we couldn't figure out how to set the number of cores for TINKER simulations which resulted in lack of data on simulations varying the number of cores using. If we can figure out a way to set the number of cores as well as how to build a model without crashing during the simulation, we could then have more data on the AMOEBA simulation. Forcefields are the central components of all

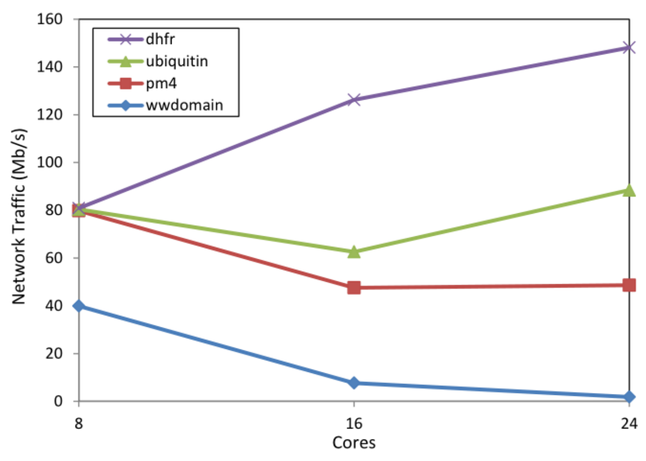


Figure 5: Inbound network traffic using expllicit model as a function of number of CPU cores

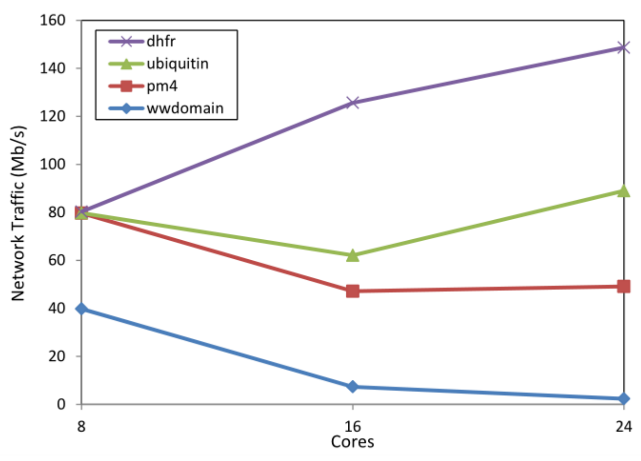


Figure 7: Outbound network traffic using expllicit model as a function of number of CPU cores

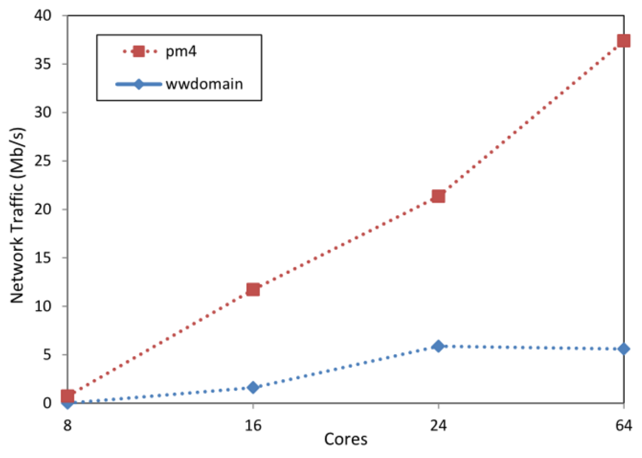


Figure 6: Outbound network traffic using impllicit model as a function of number of CPU cores

biomolecular simulations. AMOEBA force field is a new polarizable forcefield that allows more accurate description of molecular properties with its expensive computational cost. Even though AMOEBA can generate descriptive calculations, our results suggest that without using a super computer, it will not be realistic to conduct molecular dynamic simulations using currently available AMOEBA forcefield.

References

[1] J. Durrant and J. A. McCammon, "Molecular dynamics simulations and drug discovery," *BMC Biology*, vol. 9, no. 1, p. 71, 2011. [Online]. Available: http://www.biomedcentral.com/1741-7007/9/71

[2] N. Atias and R. Sharan, "Comparative analysis of protein networks: hard problems, practical solutions," *Commun. ACM*, vol. 55, no. 5, pp. 88–97, May 2012. [Online]. Available: http://doi.acm.org/10.1145/2160718.2160738

[3] J. W. Ponder and D. A. Case, "Force fields for protein simulations." *Advances in protein chemistry*, vol. 66, pp. 27–85, 2003. [Online]. Available: http://view.ncbi.nlm.nih.gov/pubmed/14631816

[4] J. W. Ponder, C. Wu, P. Ren, V. S. Pande, J. D. Chodera, M. J. Schnieders, I. Haque, D. L. Mobley, D. S. Lambrecht, R. A. DiStasio, M. Head-Gordon, G. N. I. Clark, M. E. Johnson, and T. Head-Gordon, "Current status of the amoeba polarizable force field," *The Journal of Physical Chemistry B*, vol. 114, no. 8, pp. 2549–2564, 2010, pMID: 20136072. [Online]. Available: http://pubs.acs.org/doi/abs/10.1021/jp910674d

[5] A. R. P. Grossfield and P. J. W., "Ion solvation thermodynamics from simulation with a polarizable force field," *Journal of American Chemical Society*, vol. 125, pp. 15 671–15 682, 2003.

[6] P. Ren and J. W. Ponder, "Consistent treatment of inter- and intamolecular polarization in molecular mechanics calculations," *Journal of Computational Chemistry*, vol. 23, pp. 1497–1506, 2002.

[7] M. S. Friedrichs, P. Eastman, V. Vaidyanathan, M. Houston, S. Legrand, A. L. Beberg, D. L. Ensign, C. M. Bruns, and V. S. Pande, "Accelerating molecular dynamic simulation on graphics processing units," *Journal of Computational Chemistry*, vol. 30, pp. 864–872, 2009.

[8] W. Liu, B. Schmidt, G. Voss, and W. Mlleru-Wittig, "Accelerating molecular dynamics simulations using graphics processing units with CUDA," *Computer Physics Communications*, vol. 179, no. 9, pp. 634 – 641, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0010465508002191

[9] R. N. Badi Abdul-Wahid, Haoyun Feng and J. Sweet, "Final report: Determining the effect of architecture on molecular dynamics simulations," *Unpublished*, 2011.

[10] "Simbios — advancing physics-based simulation of biological structures," simbios.stanford.edu, Oct 2012. [Online]. Available: http://simbios.stanford.edu