Lisa Felberg
CS 267
02.07.13

Assignment 0

About me:

I am a first year graduate student in the Chemical Engineering department. My research project focuses on designing optimal polymer electrolyte membranes (PEMs) for use in fuel cells. To do this, in-house code has been developed, integrating a linearized Poisson-Boltzmann equation (LPBE) solver into a Brownian Dynamics (BD) program. This code has previously been used to simulate complex protein interactions in a salty environment.

Most of my programming experience has been in Matlab and Perl, but the in-house code has been written in C++ and needs work to be parallelized and optimized. For these ends, I would like to learn C and C++ and learn how to make the LPBE code more efficient by parallelization.

Application: High-throughput PEMFC design

Background

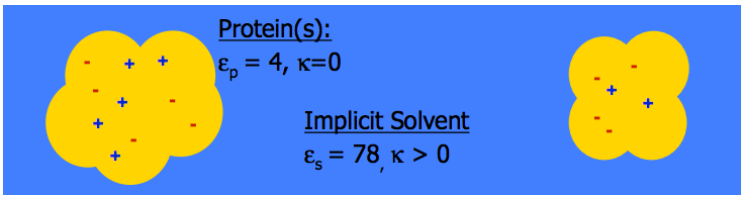
The invention of the Polymer Electrolyte Membrane Fuel Cell (PEMFC) dates back to the 1950s. The components of a PEMFC are similar to a common fuel cell, consisting of an anode, at which a reduction reaction occurs, a cathode, at which an oxidation reaction occurs, and a PEM, through which ions created in one half-cell diffuse to the other half-cell to create an electrochemical gradient. In the late 1960s, Dow introduced Nafion®, which remains to date the gold standard of proton exchange membranes[1]. Potential applications of PEMFCs range from handheld electronics to automobiles to the military; their benefits when compared to other energy sources include: portability, efficiency, renewability, a lack of emissions, and a zero-carbon footprint[2].

Although the PEMFC has been extensively studied in the past sixty years, fundamental issues in the performance properties of PEMs, particularly in regards to water flooding and insufficient ion transport, have prevented the widespread commercialization of this technology as a replacement for the common battery. While ion transport across the PEM can occur through both natural diffusion and reactive pathways[3], unwanted water is transported via an effect known as electro-osmotic drag (EOD), wherein the charged ions amass clusters of water that then pass through the PEM to flood the fuel cell.

Molecular level studies have lent insight into explaining the deficiencies in current PEMFC

technology, but little effort has been made to propose more general design rules of the PEM material, based on simple, generalized concepts of polymer structure and charge distributions. To do this, I will simulate a higher level of interaction between the polymer and the electrolyte solution, using a numerical solution to the LPBE that solves for electrostatic forces between macromolecules in a crowded, charged environment, introduced by Head-Gordon[4,5]. The LPBE solver, called the Poisson-Boltzmann semi-analytical method (PB-SAM), combines boundary elements and previous analytical formalism[5] to generate the complex molecular charge distributions more efficiently and effectively than commercial software packages. Simply, the method describes molecular boundaries and extended surfaces as a cluster of spheres, each with its own complex charge distribution, see *Figure I.*

Figure I: Schematic of PB-SAM simulation environment[6]



The LPBE solver has been incorporated into a Brownian Dynamics (BD) scheme, which will be used to simulate transport properties of water-complexed ions through regions of hydrophilic charged polymer environment. To test the novel mesoscale theoretical framework, polymers with extensively studied structures, like Nafion® and Hyflon®, will be modeled using the LPBE-BD approach, and benchmarks like radial distribution function plots and diffusion coefficients will be compared to both simulation and experimental results. After the method has been rigorously compared to literature, it will then be extended to develop systems of novel shapes, such as cylindrical, laminar, or amorphous hydrophilic domains[7]. Arbitrary charge distributions will also be introduced as a point of focus as well.

Computing Language and platforms

This project is intended to be another application of code developed by a previous lab member, Eng Hui Yap. Yap took CS 267 in Spring 2009, so my efforts will be an extension of her progress. From reviewing the results of her first assignment and her final project[6,8], I have found the following:

Application Details - language, parallel platforms, apparent bottlenecks

The code was developed in object-oriented C++, using MPI for message passing. The solver involves extensive dense matrix operation and requires BLAS and LAPACK.

The ultimate application of the code is to simulate a large number of molecules, which each in turn can be subdivided into many spheres. This indicates a use for parallelization at many layers within the code. For this, Eng Hui utilized both OpenMP and MPI in her final project.

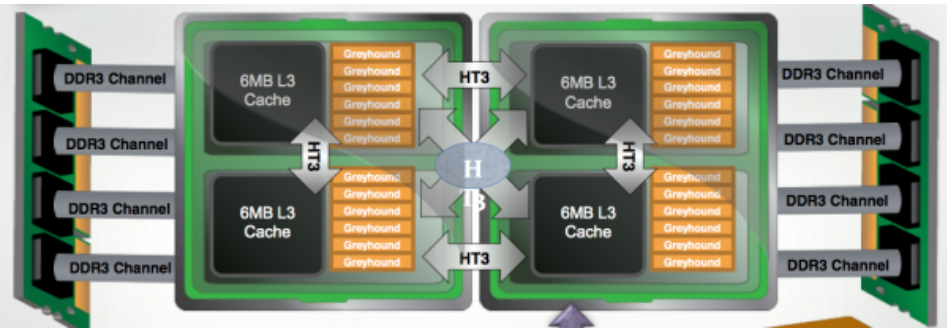
A potential bottleneck might occur during mutual polarization, as each sphere requires information about the polarization status of all other spheres. As an extension of Eng Hui's code, this bottleneck may be reduced by replacing the all-body mutual polarization calculations with a 3-body approximation in the algorithm.

Further tests must be performed and modifications made before PB-SAM can be benchmarked and compared with similar algorithms for fuel cell applications. Scaling benchmarks are also unknown at this time.

Computational Resources

PB-SAM is intended to run on both group managed clusters and the NERSC *Hopper* cluster, which is ranked 19th on the November 2012 Top500 list. *Hopper* is a Cray XE6, massively parallel processing (MPP) system with 6,384 compute nodes, each with two twelve-core AMD 'Magny Cours' 2.1-GHz processors, see *Figure II*. Each core has at least 32 GB of memory. It supports MPI message passing protocols. The Linpack performance of Hopper is about 1054.0 TFlop/s.

Figure II: Schematic of a Magny Cours Processor[9]



References

[1] Smitha, B.; Sridhar, S.; Khan, A. A. *J. Membrane Sci.* **2005**, 259, 10.
[2] Spohr, E.; Commer, P.;Kornyshev, A. A. *J. Phys. Chem. B.* **2002**, 106, 10560.
[3] Paddison, S. J.; Paul, R. *Phys. Chem. Chem. Phys.* **2002**, 4, 1158.
[4] Yap, E.-H.; Head-Gordon, T. *J. Chem. Theo. Comp.* **2010**, 6, 2214.
[5] Lotan, I.; Head-Gordon, T. *J. Chem. Theo. Comp.* **2006**, 2, 541.
[6] Yap. E.-H. CS267, *Final Project.* **2009**. http://goo.gl/B7kMs
[7] Knox, C.; Voth, G. A. *J. Phys. Chem. B.* **2010**, 114, 3205.
[8] Yap, E.-H. CS267, *Assignment 0.* **2009**. http://goo.gl/L1QJ3
[9] NERSC. *Computational Systems: Hopper*. **2013**. http://goo.gl/9l1Tq