GPU Computational Screening of Carbon Capture Materials

J. Kim[1], A Koniges[1], R. Martin[1], M. Haranczyk[1], J. Swisher[2], and B. Smit[1,2]

[1] Lawrence Berkeley National Laboratory, Berkeley, CA 94720

[2] Department of Chemical Engineering, University of California, Berkeley, Berkeley, CA 94720

E-mail: jihankim@lbl.gov

Abstract. In order to reduce the current costs associated with carbon capture technologies, novel materials such as zeolites and metal-organic frameworks that are based on microporous networks are being studied. We have developed a GPU code that can characterize and screen a large database of zeolite structures and help identify the most efficient structures for carbon capture. The interactions between the atoms that constitute the zeolite structures and the gas molecules such as carbon dioxide and methane are described by the Lennard-Jones and Coulomb potentials. We have also developed a CPU algorithm that identifies inaccessible regions inside the zeolite structures based on their energy profiles. Putting all this together, we can compute the selectivity for separating carbon dioxide molecules from other flue gases in each of the zeolite structures.

1. Introduction

The implementation of current technology of carbon capture based on scrubbing carbon dioxide with amine solutions may decrease the efficiency of a power plant by as much as 30–40% [1]. An alternative solution with lower energy costs such as solid-state adsorbent based on porous materials is being developed. In this work, we focus on zeolite structures, which are among the most widely reported adsorbents for carbon capture [2]. Finding the optimal zeolite is an arduous task, since the number of possible pore topologies is extremely large. Approximately 190 unique zeolite frameworks are known to exist today in more than 1,400 zeolite crystals of various chemical composition. However, these experimentally known zeolites constitute only a very small fraction of more than 2.7 million structures that are feasible on theoretical grounds [3],[4], of which between 314,000 and 585,000 are predicted to be thermodynamically accessible as aluminosilicates, with the remainder potentially accessible via elemental substitution [5],[6]. Databases of similar or greater magnitude can be developed for other nanoporous materials such as metal organic frameworks (MOFs) or ZIFs.

To characterize and later screen a large database of zeolite structures within a reasonable computational time, we developed a GPU code to accelerate our molecular simulations. Unlike conventional CPUs, GPUs have many more transistors devoted to data processing and, as such, can provide significant performance improvement in problems that can be easily mapped into its multithreaded hardware. In our code, the bottleneck routines can easily be abstracted into a SIMD (single instruction, multiple data) format, making it ideal for GPU acceleration.

The paper is organized as follows. In Section 2, we outline the algorithm behind our hybrid GPU + CPU screening code. In Section 3, we provide both the performance and the Henry coefficient results obtained from the code and briefly mention future work regarding our project.

2. Algorithm for Characterizing Zeolites

Various computational techniques in molecular simulations can be used to predict adsorption and diffusion properties of the gas molecules adsorbed in zeolites [7]. In this work, we compute the Henry coefficients (K_{H}) of CH_4 and CO_2 gas molecules, which are basic constants describing equilibrium between the gas phase and adsorbed phase. The ratio of Henry coefficients characterizes the selectivity of the material at low pressure and serves as a useful quantity for screening large number of adsorbent frameworks. The mathematical expression for the Henry coefficient is as follows [8]:

$$K_{\mathrm{H}} = \beta\langle\exp(-\beta U_{\mathrm{ins}})\rangle,$$

where $\beta = 1 \,/\, (k_{\mathrm{B}}T)$ with k_{B} representing the Boltzmann constant and T indicating temperature. U_{ins} is the energy of the test molecule inserted at a random position in the materials, that is, a Widom insertion move. This formula is frequently used in molecular simulations to compute the chemical potential of a particular component. In general, ideal zeolite structures that are suitable for carbon dioxide separation will possess a sufficiently large ratio between the CO_2 Henry coefficient and the Henry coefficients of other flue gases.

Our GPU algorithm for simulating a single zeolite structure is summarized as follows: (1) construct an energy grid where each grid point maps to the total energy value of a single gas molecule (either CH_4 or CO_2 in this work), (2) automatically detect inaccessible regions within the zeolites utilizing the energy grid constructed from the previous step, and (3) conduct a large number of Monte Carlo Widom insertion moves in the accessible space to compute the average Henry coefficient of the gas molecule inside the zeolite structure.

2.1. Energy grid construction

The purpose behind constructing the energy grid is twofold: (1) to obtain the test molecule insertion energies via interpolating energy values from the grid in the Widom insertion moves and (2) to automatically detect the inaccessible regions within the zeolites via marching through the grid points so that the subsequent Monte Carlo insertions are rejected at these positions. This is important because in our calculation of the Henry coefficient, we insert at random positions and, therefore, inaccessible cavities should be excluded from the sampling.

Because zeolites are crystalline structures, we can characterize various properties of the zeolite by analyzing just a single unit cell and applying the periodic boundary condition. In the GPU algorithm, an energy grid with a mesh size of $0.1\mathring{A}$ along the x, y, and z directions is superimposed on top of the entire simulation volume of a single zeolite unit cell. In each of the energy grid points, we compute the interaction between the guest gas molecule and all of the framework atoms using pairwise potentials (for noncharged molecules such as CH_4, the Lennard-Jones potential, and for charged molecules such as CO_2, the combination of Lennard-Jones and Coulomb potentials). We use the Ewald summation to compute the periodic Coulomb potential values by replacing the summation of interaction energies from the real to the Fourier space for faster convergence.

Given the large number of energy grid points (over a million for most zeolite structures), we can efficiently utilize the thousands of lightweight GPU threads present within the GPU architecture to accelerate the energy grid construction kernel. From the hardware perspective, each CUDA thread maps to a single energy grid point, and the pairwise potential calculations are conducted in parallel inside the GPU. We minimize the number of global memory transactions by moving the framework atom data into the fast, constant GPU memory before the energy

computation kernel launch. Other optimization techniques (e.g., reducing the number of division operators and precomputing shared terms among the CUDA threads) are used to further accelerate the routine. At the end of the routine, the array containing the energy grid values is transferred from the GPU to the CPU and used as an input to the CPU pocket blocking function, which will be explained next.

2.2. Pocket blocking

We have adopted algorithms for analysis of the void space in porous materials and detection of inaccessible pockets proposed in [9],[10]. The original algorithms relied on front propagation techniques based on partial differential equations (PDEs). In particular, the problem of determination of accessibility was casted as an Hamilton-Jacobi-type Eikonal equation in configuration space describing a guest molecule inside a material:

$$|\nabla U| = C(x).$$

Here, U is the minimal total cost, and $C(x)$ is a cost function defined at each point x in the domain and corresponds to its ability to be occupied. Abstractly, this cost function is defined at the beginning of the problem, and the solution $U(x)$ to the above problem represents the total cost, which is the smallest obtainable integral of $C(x)$, considered over all possible trajectories throughout the computational domain from a start point to finish point. The latter feature can be then used to construct practical approaches, allowing one not only to obtain shortest paths but also to predict accessibility of sections of the void space (e.g., detect inaccessible pockets).

In the current project, we have made two modifications to this algorithm. First, we have used the energy criterion to determine cost and the related accessibility of a particular configuration/point in the domain. Second, we have substituted solving of the Eikonal equation with a computationally cheaper flood fill algorithm.

The energy terms calculated at each discrete grid point can be interpreted as conveying the probability of the guest molecule occupying that position in the form of the Boltzmann factor, exp(-βE_i) with E_i representing the energy of the ith grid point. We interpret this grid in a binary fashion, as containing grid points that can or cannot be occupied by the guest molecule in the timeframe of our application. We set the following.

$$if(E_i < (p * T)) \rightarrow accessible;$$
$$else \rightarrow inaccessible;$$

Here T is the temperature, and p relates to the probability of the position being occupied. At long timescales, for instance in geologic applications, high barriers can be overcome, and so p can take a high value; however, for our carbon capture application we set $p = 15$ such that a point is accessible if exp(-E_i) > exp(-15$\times T$). Given this binary interpretation of the energy grid, we segment the grid into disconnected nonperiodic regions (marches) using the traversal algorithm described above. Each of these distinct marches is then analyzed periodically to determine whether it forms a channel through the void space or an inaccessible pocket. We perform this analysis by examining the positions where each march reaches a face of the unit cell and inspecting their periodic neighbors for accessibility, connecting these marches. By this method, marches that do not reach a face are pockets, as are marches that cross the periodic boundary but do not form a loop. It is important to exclude these inaccessible pockets prior to performing Monte Carlo sampling; otherwise the energy terms calculated within the pockets will contribute to the measured behavior of the system, even though in reality the guest molecule cannot access these positions. We overcome this problem by generating blocking spheres for each march that is flagged as being part of a pocket; and, in the following Monte Carlo step, moves that are within a blocking sphere are rejected. We generate spheres large enough that the

Table 1. Simululation timing results (seconds) for 193 IZA zeolite structures. The wall times spent in three main routines are tabulated for both CH_4 and CO_2 molecules.

Gas	GPU Grid Construction	CPU Pocket Blocking	GPU Widom Insertions	Total Wall Time
CH_4	139.95	274.84	5.63	426.61
CO_2	1521.42	4439.91	65.73	6137.51

entire march is excluded, without interfering with other marches. The algorithm for generating these spheres is described in [10].

2.3. Monte Carlo Widom insertion

Utilizing the energy grid and the blocking spheres from the previous subsections, we can calculate the Henry coefficients for a given zeolite structure using the Monte Carlo Widom insertion moves. We test insert a single, guest gas molecule inside our simulation box and compute the Boltzmann factor (i.e., $\exp(-\beta U_{\text{ins}})$), where U_{ins} is the insertion energy of the test molecule obtained from the interpolating energy values from the energy grid. In our algorithm, we set the total number of Monte Carlo Widom insertion moves to be 4.3 million. Because these insertions can be conducted independently, we use the GPU to accelerate the insertion routine and utilize the CUDA CURAND library to generate random numbers uniformly sampled from the unit cell for the Widom insertion moves. Zero contribution to the Henry coefficient is added upon sampling regions inside the blocking sphere. For linear molecules such as carbon dioxide, multiple insertions for each atom (e.g., C, O, O) are conducted in sequence, and the energy for each of the atoms is added to obtain the total U_{ins}.

3. Performance Results

We utilize the NERSC Dirac GPU cluster (Figure 1) to run our GPU zeolite simulations. Dirac consists of 44 NVidia Fermi Tesla c2050 GPU cards with each node having 2 Intel 5530 2.4 GHz Nehalem quad cores. All the results here are based on single-precision floating numbers, since the format has sufficient precision to accurately model our stochastic simulations. Simulation timing results for computing the CH_4 and the CO_2 Henry coefficients of 193 IZA zeolite structures are tabulated in Table 1.

As can be seen from the data, the total wall time spent in CO_2 calculations is around 11.01× longer compared with the total wall time spent in CH_4 calculations as the long-range Coulomb interactions entail summing interactions over multiple copies of the unit cells in both the real and the Fourier spaces in the Ewald summation. The Coulomb term required to perform the CO_2 energy grid calculation has the additional effect of causing many more isolated regions of low energy than are detected in the CH_4 energy grid. The impact of this is to generally increase the quantity of pockets that need to be excluded in the pocket blocking routine. Overall, 64.4% (CH_4) and 72.3% (CO_2) of total wall time is spent in the single CPU core pocket blocking routine. This is not surprising considering that this portion of the code has not been optimized and there remains a great deal of scope for acceleration. In the future, we intend to implement various multicore approaches (e.g., OpenMP and Pthreads) to treat the pockets independently and exclude them in parallel. The quantity of pockets detected is generally in the tens or hundreds, and so the blocking step is not likely to benefit fom a GPU application. Therefore, this part of the code is not the emphasis of this work.

The CH_4 and the CO_2 IZA Henry coefficients data are shown in Figure 2 using histograms with a bin size of $2.5e^{-6}$ mol/gram/Pa. The Henry coefficient results agree well with the CPU results (not included in the figure), and the performance improvement makes it possible to simulate an entire hypothetical database of zeolite structures in a reasonable time. As the

Figure 1. The NERSC Dirac GPU cluster rack.

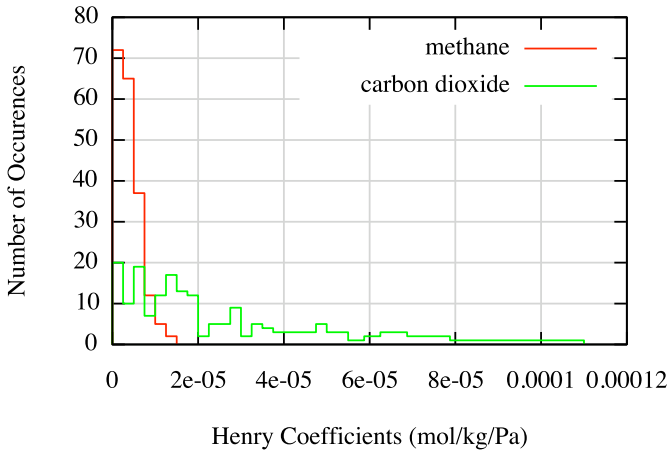

Figure 2. Histogram of CH_4 and CO_2 Henry coefficients for the 193 IZA zeolite structures.

next step, we plan to incorporate additional features to our code to simulate other framework structures such as metal organic frameworks.

Acknowledgments

RLM and MH were supported by the Director, Office of Advanced Scientific Computing Research, Office of Science, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 through the SciDAC-e project Accelerating Discovery of New Materials for Energy-related Gas Separations through PDE-based Mathematical and Geometrical Algorithms and Advanced Visualization Tools. This research used resources of the National Energy Research Scientific Computing Center, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231. JK and AK were supported by the Director, Office of Science, Advanced Scientific Computing Research, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231 and JK is part of the Petascale Initiative in Computational Science and Engineering at LBNL/NERSC.

References

[1] IPCC, IPCC Special Report on Carbon Dioxide Capture and Storage, Cambridge University Press, Cambridge, 2005.
[2] S. Choi, J. H. Drese, C. W. Jones, ChemSusChem 2 (2009) 796-854.
[3] M.D. Foster, M.M.J. Treacy, http://www.hypotheticalzeolites.net (accessed Nov. 13, 2009).
[4] D.J. Earl, M.W. Deem, Ind. Eng. Chem. 45 (2006) 5449-5454.
[5] M.W. Deem, R. Pophale, P.A. Cheeseman, D.J. Earl, J. Phys. Chem. C 113 (2009) 21353-21360.
[6] R. Pophale, P.A. Cheeseman, M.W. Deem, Phys. Chem. Chem. Phys. DOI: 10.1039/c0cp02255a.
[7] D.M. D'Alessandro, B. Smit, J. Long, Angew. Chem. Int Ed. 49 (2010) 6058-6082.
[8] D. Frenkel and B. Smit, Understanding Molecular Simulation, ASecond Edition: From Algorithms to Applications, Academic Press, 2001.
[9] M. Haranczyk and J.A. Sethian, J.A. (2009). Proc. Natl. Acad. Sci. USA (PNAS) 106:21472-21477.
[10] M. Haranczyk and J.A. Sethian, J. Chem. Theory Comput. 6, no. 11 (2010) 3472-3480.