# High-throughput Characterization of Porous Materials Using Graphics Processing Units

Jihan Kim,[†] Richard L. Martin,[‡] Oliver Rübel,[‡] Maciej Haranczyk,[‡] and Berend[¶]

[†]Lawrence Berkeley National Laboratory, Material Sciences Division,
Berkeley, CA 94720
[‡] Lawrence Berkeley National Laboratory, Computational Research Division,
Berkeley, CA 94720
[¶] Departments of Chemical and Biomolecular Engineering and Chemistry,
University of California, Berkeley, CA 94720

E-mail: jihankim@lbl.gov

**Acknowledgements:** 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.
J.K., R.L.M., O.R and M.H. 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. In addition, R.L.M. and O.R. were supported by the Scientific Discovery through Advanced Computing (SciDAC) program's Visualization and Analytics Center for Enabling Technologies (VACET). M.H. was supported jointly by DOE Office of Basic Energy Sciences and the Office of Advanced Scientific Computing Research through SciDAC project \#CSNEW918 entitled ``Knowledge guided screening tools for identification of porous materials for CO2 separations''. B.S. was supported as part of the Center for Gas Separations Relevant to Clean Energy Technologies, an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Award Number DE-SC0001015.

# High-throughput Characterization of Porous Materials Using Graphics Processing Units

Jihan Kim,[*,†] Richard L. Martin,[‡] Oliver Rübel,[‡] Maciej Haranczyk,[‡] and Berend Smit[¶]

*Lawrence Berkeley National Laboratory, Material Sciences Division, Berkeley, CA 94720, Lawrence Berkeley National Laboratory, Computational Research Division, Berkeley, CA 94720, and Departments of Chemical and Biomolecular Engineering and Chemistry, University of California, Berkeley, CA 94720*

E-mail: jihankim@lbl.gov

**Abstract**

We have developed a high-throughput graphics processing units (GPU) code that can characterize a large database of crystalline porous materials. In our algorithm, the GPU is utilized to accelerate energy grid calculations where the grid values represent interactions (i.e., Lennard-Jones + Coulomb potentials) between gas molecules (i.e., $CH_4$ and $CO_2$) and material's framework atoms. Using a parallel flood fill CPU algorithm, inaccessible regions inside the framework structures are identified and blocked based on their energy profiles. Finally, we compute the Henry coefficients and heats of adsorption through statistical Widom insertion Monte Carlo moves in the domain restricted to the accessible space. The code offers significant speedup over a single core CPU code and allows us to characterize a set of porous materials

[*]To whom correspondence should be addressed
[†]Lawrence Berkeley National Laboratory, Material Sciences Division, Berkeley, CA 94720
[‡]Lawrence Berkeley National Laboratory, Computational Research Division, Berkeley, CA 94720
[¶]Departments of Chemical and Biomolecular Engineering and Chemistry, University of California, Berkeley, CA 94720

at least an order of magnitude larger than ones considered in earlier studies. For structures selected from such a prescreening algorithm, full adsorption isotherms can be calculated by conducting multiple grand canonical Monte Carlo simulations concurrently within the GPU.

## 1 Introduction

Porous materials, such as zeolites and metal organic frameworks (MOFs), have been exploited in many current technologies and are considered to be a very important class of materials for many new industrial applications. For example, zeolites are commonly used as chemical catalysts, in particular as cracking catalysts in oil refinement, membranes for separations, and water softeners.[1–4] Additionally, there is an increasing interest in utilizing zeolites as membranes or adsorbents for $CO_2$ capture applications.[5–8] Other materials, such as MOFs[9,10] and their subfamily of zeolitic imidazolate frameworks (ZIFs),[11] have enormous potential for gas separations and storage as well.[12,13]

A key factor that determines the utility of any nanoporous material is its optimal pore topology along with the chemical composition for given conditions in a particular application. There are approximately 190 known unique zeolite frameworks[14] in more than 1400 zeolite crystals of various chemical composition and geometry.[15] However, these experimentally known zeolites constitute only a very small fraction of more than 2.7 million structures that are feasible on theoretical grounds.[16,17] Of these, between 300,000 and 600,000 are predicted to be thermodynamically accessible as aluminosilicates, with the remainder also potentially accessible via elemental substitution.[18,19] All of the zeolite structures in this research work are comprised of silicon and oxygen atoms, making these materials much more simple in terms of their chemical composition than ZIFs or MOFs. The chemical composition of zeolite structures can be altered by replacing some of the silicon atoms with aluminum (or other) atoms, and then adding cations (e.g. $Na^+$) to impose charge neutrality. Changing the chemical makeup of zeolite materials in this way further increases the number of possible structures. Structure sets of similar or even greater size are expected for other nanoporous materials such as ZIFs,[7] which offer greater flexibility in the choice of building blocks.

In an attempt to identify optimal materials for various applications, such as gas separations,[5–7] researchers have started to screen large databases of porous materials. Molecular simulation techniques, such as the grand canonical Monte Carlo (GCMC) method, are often used in numerical simulations to accurately predict properties of materials and their guest-adsorption characteristics expressed as an experimentally verified adsorption isotherm.[20–22] However, the computational cost of molecular simulations is high, significantly limiting the number of structures that can be analyzed. To avoid the high computational costs, most of the screening strategies rely on thorough pre-screening of materials, which filters out structures based on easily obtainable structural properties, such as pore diameters and framework energy. As an example, Haldouplis et al. screened 250,000 zeolites while only about 8,000 were characterized using molecular simulations.[6] Nonetheless, it is important to note important developments in algorithms that allow high-throughput characterization of porous materials via generation of structural parameters used for pre-screening[5,23–25] or database sampling approaches.[26]

The importance of these screening strategies motivated us to approach this task from a high-performance computing point of view and utilize fast molecular simulation techniques that allow us to characterize a very large set of porous materials – more than an order of magnitude larger than what has been reported previously by other researchers. Additionally, we addressed a practical complication limiting the characterization of large sets of materials, which is determination whether pores ina material are accessible. This step typically involves visual inspection,[35] which becomes cumbersome with large number of structures. Our approach integrates an automatic analysis of topology of the materials' void space, and accordingly, the simulation domain can be defined within the accessible void space of a material and this domain reflects the space available to molecules in experiments.

In order to process large sets of materials within reasonable time, our molecular simulation tool utilizes graphical processing units (GPUs) to efficiently conduct parallel calculations. GPUs are hardware accelerators initially developed to accelerate graphics related tasks. With the advent of NVIDIA's CUDA (compute unified device architecture) and subsequent development of the CUDA software interface, general purpose GPU (GPGPU) programming has become more prevalent and commonplace in the scientific community.[27] Unlike conventional CPUs, GPUs have many more transistors devoted to data processing and as such, can provide significant performance improvement in computational problems that can be easily mapped into its multi-threaded hardware. In the context of molecular simulations, GPGPU computing has been mainly used to accelerate molecular dynamics (MD) and Monte Carlo (MC) simulations.[28–31] Our molecular simulation GPU code takes advantage of the fact that the computationally intensive bottleneck routines can easily be mapped into a SIMD (same instruction multiple data) format, making it ideal to port the code from the CPU to the GPU. Although this article focuses on GPU simulation results of the zeolite structures, the code can be easily extended to process other important classes of porous materials such as MOFs or ZIFs and can accelerate characterization in those materials as well. As such, the techniques described in this work can generalize well to many other systems.

The manuscript is organized as follows. In Section 2, we discuss the algorithmic details of our hybrid GPU + CPU characterization/screening code. In Section 3, we analyze the performance of our implementation and present results obtained using our GPU code. In Section 4, we summarize the important findings in our work and discuss avenues for future work.

## 2 Algorithm for Characterizing Porous Materials

In this work, we focus on $CO_2$ and $CH_4$ gas molecules as they comprise representative examples for the behavior of molecules with, respectively, partial atomic charges and no charges. However, the techniques described in this work can readily extend to other gas molecules such as $N_2$, He, and $H_2O$, as well. The zeolite framework is assumed to be rigid, which is a reasonable approximation[3] and as such, only the gas-framework and the gas-gas interactions are considered. As a means to characterize the zeolite structures, we compute the Henry coefficient ($K_H$) and the heats of adsorption ($\Delta h_\mathrm{i}$) values of $CO_2$ and $CH_4$. These quantities characterize adsorption of the gas molecules in porous materials. $K_\mathrm{H}$ is a basic constant that relates the equilibrium between the gas and the adsorbed phase ($\rho = K_\mathrm{H}$ P), and hence describes adsorption at very low pressure regime. In the context of $CO_2$ capture, an ideal material would exhibit a large $CO_2$ $K_\mathrm{H}$ compared to the $K_\mathrm{H}$ values of other flue gases, resulting in high selectivity for $CO_2$. For carbon capture of flue gases, the pressure values are relatively low and for most systems, $K_\mathrm{H}$ serves as an important quantity to characterize large material databases. $K_\mathrm{H}$ and $\Delta h_\mathrm{i}$ can be computed from a Monte Carlo simulation:

$$K_\mathrm{H} = \beta \langle \exp(-\beta U_\mathrm{ins}) \rangle_\mathrm{test} \tag{1}$$

and

$$\Delta h_\mathrm{i} = \frac{\langle U_\mathrm{ins} \exp(-\beta U_\mathrm{ins}) \rangle}{\langle \exp(-\beta U_\mathrm{ins}) \rangle}, \tag{2}$$

with $\beta = 1 / (k_\mathrm{B} T)$ with $k_\mathrm{B}$ representing the Boltzmann constant and $T$ indicating the temperature of the system. $T$ is fixed at 300K in all simulations reported in this work. $U_\mathrm{ins}$ represents the test particle energy of the gas molecule at a random position in the material. $\langle \ \rangle_\mathrm{test}$ indicates that in the ensemble average, the test particle does contribute to the energy of the system. Upon taking a sufficiently large number of Monte Carlo Widom insertions, which consist of randomized insertions of the gas molecules in the simulated volume, the $K_\mathrm{H}$ and the $\Delta h_\mathrm{i}$ of the gas molecules can be computed with a great deal of accuracy.

The $K_\mathrm{H}$ and the $\Delta h_\mathrm{i}$ computational algorithm for a given porous material consists of the following three important steps:

1. construct an energy grid that stores the energy values of the test gas molecule at discrete positions of the structure's unit cell ( 2.1),

2. automatically identify inaccessible regions within the structure utilizing the energy grid values from the previous calculation ( 2.2), and

3. conduct Widom insertion Monte Carlo moves to compute the Henry coefficient $K_\mathrm{H}$ and the

heats of adsorption $\Delta h_{\mathrm{i}}$ of the gas molecule ( 2.3).

This algorithm is utilized to characterize all materials in the theoretical zeolite database. We describe the algorithm outlined above in detail in the subsequent subsections. In 2.4, we briefly describe the grand canonical Monte Carlo algorithm used to compute the adsorption isotherms inside the GPU, which can provide adsorption properties of the materials at higher pressure regimes.

## 2.1 Energy Grid Construction

All of the zeolite materials in our simulations are crystalline structures and accordingly, the adsorption properties of each of these materials can be accurately characterized by examining a small number of unit cells and imposing a periodic boundary condition. Inside the numerical domain of a single unit cell, we construct a three-dimensional energy grid for each of the zeolite structures. The grid points of the energy grid each represent the sum of the Lennard-Jones and the Coulomb potentials between the gas molecule and all of the framework atoms that make relevant contribution to the interaction. The spacing of the energy grid is fixed to be 0.1Å along all three spatial dimensions. The Lennard-Jones potential and the Coulomb potentials are defined as follows:

$$U_{\mathrm{LJ}}(r) = 4\varepsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right] \tag{3}$$

and

$$U_{\mathrm{coul}}(r) = \frac{q_i q_j}{4\pi \varepsilon_o r}. \tag{4}$$

Here, $r$ represents the distance between two particles, $\varepsilon$ indicates the depth of the potential well, $\sigma$ represents the effective core size of the particles with the potential well located at $2^{1/6}\sigma$, $q_i$ and $q_j$ indicate charges of the two particles, and $\varepsilon_o$ represents the material permittivity. For all pairwise interactions, a Lennard-Jones cutoff radius $R_{\mathrm{C}}$ = 12.0Å is imposed such that the interaction is shifted to zero for $r > R_{\mathrm{C}}$ by subtraction of $U_{\mathrm{LJ}}(R_{\mathrm{C}})$ from $U_{\mathrm{LJ}}(r)$ for all $r \leq R_{\mathrm{C}}$. In our code, the long-range Coulomb interactions between the charged particles are approximated by the Ewald summation method. The work to compute the periodic Coulomb potentials is divided into summations in both real and Fourier components to accelerate convergence. All of the force fields used in this work come from Garcia-Perez et al. and Dubbeldam et al.,[32,33] which have been shown to reproduce the experimental adsorption for various zeolite structures.

Given that a very large number of energy grid points must be computed (over ten million for most zeolite structures), GPU hardware architecture can be efficiently utilized by employing thousands of CUDA threads to concurrently compute the energy values at these points in parallel. In our algorithm, the total number of CUDA threads is set to be equal to the total number of energy grid points, creating a one-to-one mapping between threads and grid points. Accordingly, each CUDA thread computes the interaction between the gas molecule at the specified grid point and all the framework atoms. The number of framework atoms, $N_{\text{tot}}$, is in most zeolites small enough ($N_{\text{tot}} < 2000$ atoms) such that all data required fits into the 64kB constant memory of the Tesla C2050 Fermi cards used, minimizing the number of GPU DRAM transactions. For $N_{\text{tot}} > 2000$, the data containing the framework atom positions is stored in the slower GPU DRAM, which leads to a decrease in performance. In practice this performance deterioration is small and has no major impact on our experiments. Also, in the hypothetical zeolite database, the number of structures with $N_{\text{tot}} > 2000$ is relatively small and, thus, the overall wall time to process an entire database is not affected significantly by the reduced memory bandwidth. Since the constant memory bandwidth is greater than the shared/L1 cache memory and given that we only require read operations from the framework atoms, the decision was made to choose constant memory over other fast memory available in the GPU. For a linear molecule like $CO_2$, we compute separate carbon and oxygen grids for the Lennard-Jones interaction while computing only one Coulomb grid (e.g., carbon atom Coulomb grid). The Coulomb interaction values for the second atom (i.e., oxygen) can be obtained by multiplying the grid point values of the carbon Coulomb grid by a pre-factor corresponding to the ratio of the charges of the two atoms. This strategy does not only reduce computation time, but it also reduces the amount of GPU DRAM required; this is important given the 3GB DRAM constraint of the Tesla C2050 cards.

In order to determine inaccessible regions within the unit cell of the structure (described in

2.2), an additional energy grid is required. This additional grid is constructed to encode the total energy of the gas molecule at each of the grid points, and accordingly approximately maps to the occupation probability of the gas molecule at that position (i.e., a higher total energy corresponds to a lower probability of occupancy). For molecules such as $CH_4$, which are modeled as point particles, this total energy grid is equivalent to the individual energy grid computed in the energy grid construction routine. However, for linear molecules, such as $CO_2$ and $N_2$, or non-linear molecules, such as $H_2O$, the total energy values need to be obtained separately from the individual energy grids computed by the grid construction routine. We obtain approximate values of energies of these molecules and store the result in a single grid point by conducting a large number of test rotation moves about the grid point and computing its average energy. In $CO_2$, for example, the carbon atom is positioned to coincide with each grid point in turn, and $N_{\mathrm{rot}} = 100$ random rotations are conducted on the two oxygen atoms about that point. Because rotations can be conducted independently and in parallel, the test rotation routine maps well to the GPU. We utilize the CUDA CURAND library to generate the random numbers that determine the gas molecule orientations. The energy value for the carbon atom is sampled directly from the Lennard-Jones and the Coulomb energy grids, whereas the energy values for the two randomized oxygen atoms are obtained using linear interpolation functions from the energy grid values. The trilinear interpolation function samples eight nearest neighbor points in the energy grid from the rectangular voxel that encompasses the sampled point. In cases where the identification of inaccessible regions within a structure is unnecessary, the test rotation routine can be skipped in order to improve computational performance.

## 2.2 Pocket Blocking Algorithm

In some materials, the arrangement of atoms in the structure is such that there exist regions of space which a guest molecule could occupy, but which it cannot access, due to the positions of surrounding atoms. These regions constitute ‘inaccessible pockets’ of void space, and are contrasted with accessible regions of space, i.e. ‘channels’. In computer calculations it is critical to account for these positions such that they are not considered in, for example, the calculation of guest-accessible volumes or surface areas, or the prediction of adsorption properties using molecular simulation techniques.[34,35] It is typical to detect inaccessible pockets through visual inspection of so-called pore landscapes, and subsequently to block them with exclusion spheres.[36] However, it is not practical to perform a visualization-based analysis on the very large quantity of materials we are concerned with. Accordingly, we have recently developed algorithms for the automatic segmentation of void space (into accessible and inaccessible regions) and exclusion of pockets.[37]

The original algorithms[37] relied on partial differential equation (PDE)-based front propagation techniques, by which the grid representing the guest-accessible positions (or, in the case of complex non-spherical probes, guest accessible orientations at each position[38]) can be segmented. There are numerous advantages to a PDE-based segmentation algorithm, including the approximation of paths of least resistance between certain positions within the structure, which can give insights regarding diffusion. However, for the purposes of high-throughput characterization - of the order of hundreds of thousands of structures or more - the additional information obtained by solving the PDE is generally not of critical importance. Therefore, in order to accelerate this process, we perform a more simplistic segmentation using a parallel flood fill (also known as seed fill, boundary fill or bucket fill) algorithm described in Ref.[39] Flood fill is a recursive algorithm to determine the bounds of a connected region, and so like PDE-based methods, can segment a grid into distinct, connected regions of guest-accessible space. Following this process, regions which connect across the periodic boundary are detected and merged. In our parallel implementation, each CPU thread is assigned a separate subdomain of the unit cell, upon which it performs segmentation using flood fill. Connections across subdomain boundaries are considered in the same manner as those which cross the periodic boundary of the cell. We then proceed to identify and block inaccessible regions as discussed in the following paragraphs.

In this study, we represent the material with a three-dimensional energy-grid. 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($-\beta E_{\mathrm{i}}$) with

$E_\mathrm{i}$ representing the energy of the $i^\mathrm{th}$ grid point. We interpret this grid in a binary fashion, as containing grid points which can or cannot be occupied by the guest molecule in the timeframe of our application. We set:

$$if(E_i < (p * T)) \rightarrow accessible; else \rightarrow inaccessible; \tag{5}$$

where $T$ 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 gas separation application we set $p = 15$ such that a point is accessible if exp(-$E_\mathrm{i}$) $<$ exp(-15$\times T$). This number was chosen large enough such that in a typical zeolite crystal structure, these forbidden regions are considered to be diffusively inaccessible on an experimental timescale. From this binary interpretation of the energy grid, the material's unit cell is segmented into disconnected, non-periodic regions of void space using parallel flood fill. Each of these distinct regions is then analyzed to determine whether it forms a channel or an inaccessible pocket. We examine the positions where each region reaches a face of the unit cell, and inspect their periodic neighbors for accessibility, connecting these regions; the boundaries between each CPU thread's flood fill domain are inspected in the same manner. Using this method we classify regions as channels if they constitute a loop through the void space, else as pockets (see Figure 1).

As discussed above, it is important to exclude inaccessible pockets prior to performing techniques such as Monte Carlo sampling in order to avoid false contributions of the energy term within inaccessible regions to the measured behavior of the overall system. We have devised two techniques for the exclusion of these inaccessible regions. The first approach is to generate a set of exclusion spheres which span each pocket, without interfering with other regions. The algorithm for generating these spheres is described in Ref.[37] In the subsequent Monte Carlo step, moves which are within an exclusion sphere are rejected. The obtained set of blocking spheres can be visualized or used in other molecular simulation packages. The second approach, which is utilised in our high-throughput analysis, is simply to mark grid points which are within inaccessible space by setting their energy-grid values to be prohibitively large. This saves time by bypassing the generation of blocking spheres, and in the Widom insertion step the high values at these points are sufficient to identify the inaccessible space. This constitutes a further performance improvement with respect to our earlier approaches.[37,39] Both of these techniques are implemented in our tool as multicore CPU functions, wherein each thread works independently to exclude an individual pocket.

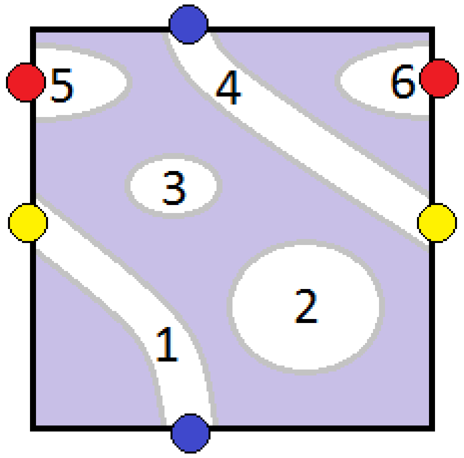


Figure 1: A two-dimensional illustration of the pocket blocking technique. Having segmented the energy-grid into distinct regions of occupiable space, we examine the positions where regions touch the periodic boundary. Hence, we merge regions 1 and 4 and find that they form a channel; regions 2 and 3 are pockets since they do not reach the boundary; and regions 5 and 6 are merged to form a pocket which crosses the periodic boundary.

## 2.3 Widom Insertion Monte Carlo

Utilizing the energy grids, the $K_\mathrm{H}$ and the $\Delta h_\mathrm{i}$ of gas molecules inside porous materials can be calculated via Widom insertion Monte Carlo moves. Similar to the test rotation routine, Widom insertion is conducted inside the GPU, again utilizing the CUDA CURAND library for random number generation. For $CH_4$, the insertion algorithm entails choosing random particle positions uniformly sampled from the entire simulation box and repeating this process while sampling the

Lennard-Jones potential values via interpolation using values from the energy grid. For $CO_2$, the carbon atom is randomly placed uniformly sampled from the entire simulation volume, similar to the case of $CH_4$, while both the Lennard-Jones and the Coulomb interaction energies of the carbon atom are sampled from the energy grids via linear interpolation functions. In zeolite structures that have non-orthogonal unit cells, the randomly generated position value might fall outside of the unit cell, which causes the point to be rejected without making contribution to the Monte Carlo statistics. Also, if the position of the carbon atom falls inside the inaccessible region, the sampled energy value would be very high due to the high energy values set in the CPU pocket blocking routine. The energy value is set high enough (i.e. 1 million Kelvin) such that the entire $CO_2$ molecule would possess large energy regardless of the values sampled from the other two oxygen atoms. The same holds true in case that any of the atoms falls inside the inaccessible region. Once the insertion of the carbon atom is finished, the algorithm proceeds to insert the first oxygen atom by randomly sampling from a surface of a sphere with a radius equal to the bond length of $CO_2$, i.e., 1.16Å. If the oxygen atom falls outside of the unit cell, then the periodic boundary condition is used to move the position of the atom back inside the unit cell via appropriate displacements. Finally, the second oxygen atom is placed in a position such that all three atomic positions are collinear to one another. The placement of the final atom in the linear molecule does not require generation of random numbers due to zero degrees of freedom. Inside the code, variables that store the total energy and the Boltzmann factors are updated at each iteration of the Widom insertion Monte Carlo cycles. For the GPU thread configurations, $16 \times 14 = 224$ CUDA blocks are generated with a block size of 64 in the CUDA kernel for the Widom insertions. Each of the CUDA threads conduct 1000 independent Widom Monte Carlo cycles, resulting in a total of $\approx$ 14 million insertion moves. Unless the CUDA occupancy is set to be too low, the block size and the number of blocks can be changed easily without disrupting the code's functionality or performance.

Rather than computing the adsorption properties for the entire structure, the code can also calculate the local $K_{\mathrm{H}}$ and the local $\Delta h_{\mathrm{i}}$ of the gas molecule within a specified subset region of our simulation volume. This capability allows us to focus on different regions within the porous material to obtain a better understanding on the local adsorption properties of the regions of interest. In the simulation code, the subset region is described by the union of a number of spheres at different positions and various radii. In the local $K_{\mathrm{H}}$ and $\Delta h_{\mathrm{i}}$ calculations, the Widom insertion algorithm first checks to see whether the $CO_2$ molecule is inside any of the spheres that describe the local region and rejects any other insertion moves. Accordingly, in the limiting case where the spheres cover the entire simulation volume, the problem simply reduces back to the original Widom insertion algorithm. If the total volume of the spheres is much smaller than the volume of the unit cell, most of the Widom insertion moves will be rejected as the probability of sampling the location regions will be small. Having a large number of rejected moves can slow down the code significantly due to the warp divergence that occurs within the GPU threads and due to the high cost of generating random numbers inside the GPU. However, because the local structural properties analysis is something that we are interested in for only a small number of structures at this point, we have yet to optimize this part of the code. In the results section, we elaborate on some of the findings that come from sampling local regions of the zeolite structures.

## 2.4 Adsorption Isotherm Calculations

Grand canonical Monte Carlo simulations are utilized to compute adsorption isotherms. Given the relatively small number of gas particles found within zeolite structures, there exists insufficient amount of work that can be efficiently parallelized via thousands of threads within the GPU. In order to circumvent this issue, we propose a parallelization strategy in which multiple Monte Carlo simulations are conducted in parallel inside the GPU.[28] Specifically, the number of different pressure values for the GCMC simulation is set to be fourteen, which is equal to the number of streaming multiprocessors (SMs) found in the Tesla C2050 card and each SM is responsible for conducting a single GCMC simulation. The threads within the CUDA blocks can work together to parallelize the different pair-wise interaction contributions and a reduction kernel can be used to sum up the contributions from each of the individual threads. The energy grids computed in the previous steps are still utilized to remove the need to explicitly compute the gas-host interactions at each step of the Monte Carlo cycles. Furthermore, there can be an additional speedup by tabulating the Fourier components of the gas-gas Ewald summation interactions in another energy grid to circumvent the need to iterate over all of the $k$ vectors for each of the pair-wise interactions during the Monte Carlo cycles.[40] Similar to the gas-host energy grid, the linear interpolation functions are utilized to estimate the gas-gas interaction. Within the periodic boundary conditions, the Fourier components are only a function of the vector distance between two particles and thus we do not lose much accuracy upon utilizing the tabular grid.

## 3 Results

In the simulation results, we initially focus our attention on the experimentally verified 188 IZA zeolite structures and later extend our analysis to a much larger hypothetical zeolite database. All of the numerical simulations were performed on the Dirac and the Carver clusters, located at the National Energy Research Scientific Computing Center (NERSC). Dirac is a testbed GPU cluster consisting of 48 nodes (44 Tesla C2050 Fermi GPU cards and 4 Tesla 1060 GPU cards). Each node contains two Quad core Intel Nehalem 5530 2.4 GHz processors with 8 MB cache, 5.86 GT/sec QPI. The Fermi GPUs have 448 CUDA cores, a PCIe x16 Gen2 system interface, and 3 GB of GDDR5 memory —where a portion of the memory (12.5%) is dedicated to error correction code (ECC) bits— yielding 2.625 GB of user available memory. According to the NVidia's Tesla C2050 specifications, double (single) precision floating point performance number peaks at 515 GFLOPS (1.03 TFLOPS) while the memory bandwidth is indicated to be 144 GB/sec. In order to utilize as many GPUs as possible on the Dirac cluster, we use a simple MPI+CUDA multi-GPU version of the code which supports static as well as dynamic scheduling of the material structures onto the MPI tasks for processing. The CPU simulations were performed on the Carver cluster, which consists of 800 Intel Nehalem 2.67 GHz quad-core processors with 24 GB DDR3 1333 MHz memory. Finally, we used the CUDA Toolkit 3.2, the CURAND Library for random number generations, and gcc 4.4.2 compiler with full optimizations in all of our simulations

## 3.1 Performance Analysis

### 3.1.1 Overall Performance

Figure 2 shows the timing results for the $CH_4$ and the $CO_2$ $K_H$ and $\Delta h_i$ calculations for the IZA structures. Given that the two quantities are computed simultaneously in the Widom insertion routine, we focus only on the results for the $K_H$ in our performance analysis.

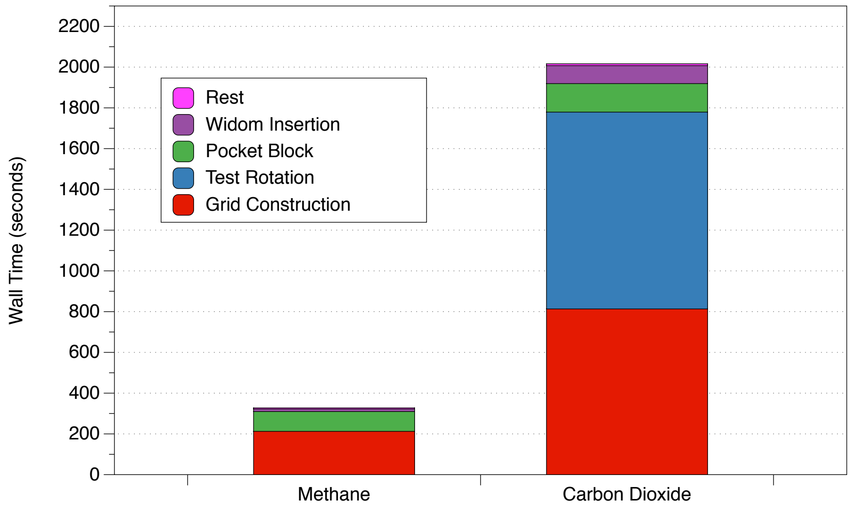


Figure 2: Distribution of wall time for 188 IZA structures with methane and carbon dioxide gas molecules. The energy grid size is set to be 0.1Å and the number of test rotations about the grid points at 100, and roughly 10 million Widom insertion Monte Carlo moves. The total computational time is divided by the major routines that comprise the simulation code: (1) GPU energy grid construction, (2) GPU test rotation, (3) CPU pocket blocking, and (4) GPU Widom insertion moves.

The total computational wall time for all of the 188 IZA $CH_4$ and $CO_2$ simulations are measured to be 327.59 and 2016.28 seconds, respectively. Accordingly, the GPU code takes about 1.74 seconds ($CH_4$) and 10.72 seconds ($CO_2$) to compute a single value of $K_H$ and $\Delta h_i$, making the $CH_4$ calculations about 6.2$\times$ faster than the $CO_2$ calculations. In general, the $CO_2$ calculations require longer wall time largely due to: a) presence of charge in the carbon and the oxygen atoms of the $CO_2$ molecules, necessitating the expensive Ewald summation computations, and b) invoking calls to the test rotation routine. Breaking down the performance of individual routines in further detail, we see that the $CH_4$ ($CO_2$) energy grid construction routine takes 212.52 (813.15) seconds, making the $CO_2$ calculations 3.83$\times$ more expensive due to the presence of Coulomb interaction terms. The number of terms in the Fourier space calculations for the Ewald summation scales with $O(k_{\text{max}}^3)$ with $k_{\text{max}}$ representing the maximum number of $k$ vectors. We can accelerate the Fourier space calculations inside the GPU by nearly two-fold by replacing the sine and the cosine terms with the *sincos* function for terms located in the inner-most loop of the nested $k$ vector. In the $CO_2$ calculations, 47.9% of the wall time is spent in the test rotation routine as opposed to 0% for $CH_4$ since the rotations are only required for multi-atom, linear and non-linear molecules. Later on in the paper, we explore the performance impact of varying the number of test rotations in this routine. The amount of time spent in pocket blocking for $CH_4$ ($CO_2$) is 97.11 (139.93) seconds, making the performance for the different guest molecules comparable. The additional Coulomb interaction term in the $CO_2$ calculations often produce more complicated energy profiles in the simulation box, which can increase the time spent in the flood fill step. Finally, the wall time spent in the Widom insertions are proportionally small compared to the overall wall time in both $CH_4$ (12.51 seconds) and $CO_2$ (87.91 seconds) calculations. This is not surprising, given that computational intensity is low in the Widom insertion moves as the routine involves mostly reading off pre-computed energy grid values from an array stored inside the GPU. The performance numbers for the Widom insertions indicate that the wall time is 7.02$\times$ larger in the case of $CO_2$ compared to $CH_4$, which can be explained by the following. $CO_2$ calculations require two random and one non-random insertion, as well as interpolation of values from both the Lennard-Jones and the Coulomb grid, to obtain an energy value for a single configuration. On the other hand, the $CH_4$ molecule entails one random insertion and one interpolation from the Lennard-Jones grid for a single configuration. The number of Monte Carlo cycles for the Widom insertion routine can be changed depending on the level of accuracy desired for the $K_H$ and the $\Delta h_i$ results, which can affect the proportional wall time spent in this routine as its wall time scales linearly with respect to the number of cycles.

### 3.1.2 Numerical Accuracy: Energy Grid Size

Next, we analyze the relationship between the energy grid size and code performance. In general, the computational wall time for the $K_H$ and the $\Delta h_i$ calculations can be reduced by increasing the mesh size of the energy grid. In practice, the grid size should be chosen small enough to ensure numerical accuracy of the $K_H$ and $\Delta h_i$ calculations performed using the energy grid. In earlier work on GPU waste recycling Monte Carlo,[28] we have demonstrated that in the zeolite MFI structure for $CH_4$ molecules, utilizing the energy grid provides average energy values within 0.05% of the ones obtained from direct Lennard-Jones potentials without the grid, providing good justification of using the grid.

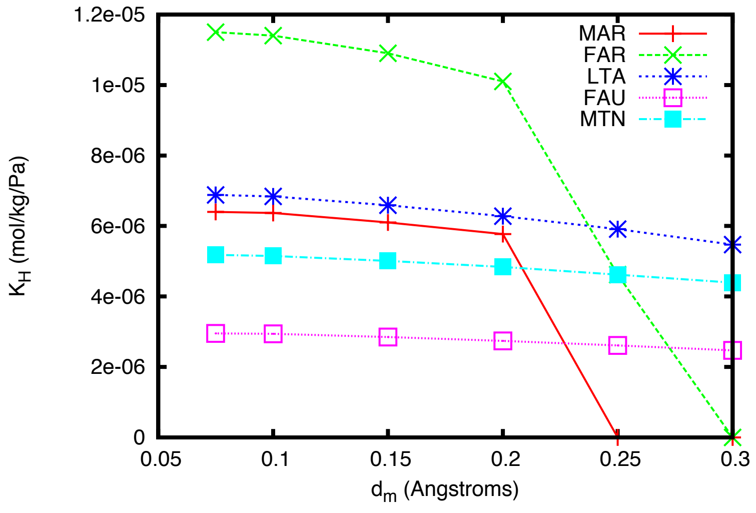


Figure 3: $CO_2$ $K_H$ values for five IZA blocking set structures for $d_m$ = 0.075, 0.10, 0.15, 0.20, 0.25, and 0.30Å. The sudden jump in the Henry coefficient values for MAR and FAR at $d_m$ = 0.25 and 0.30Å signify the effect of erroneous results due to insufficient accuracy of the energy grid.

In Figure 3, the mean values of $CO_2$ $K_H$ of five IZA structures (i.e. MAR, FAR, LTA, FAU, and MTN) are plotted for different mesh sizes ($d_m$ = 0.075, 0.10, 0.15, 0.20, 0.25, and 0.30Å). These five structures are part of twenty-one blocking set in IZA, which includes zeolite structures that possess inaccessible regions for the $CO_2$ molecules; this set comprises of about 11% (21/188) of the entire IZA database. It is not necessary to plot the curves for all the structures in the blocking set as these five are sufficient to show the general trends for all blocking structures with respect

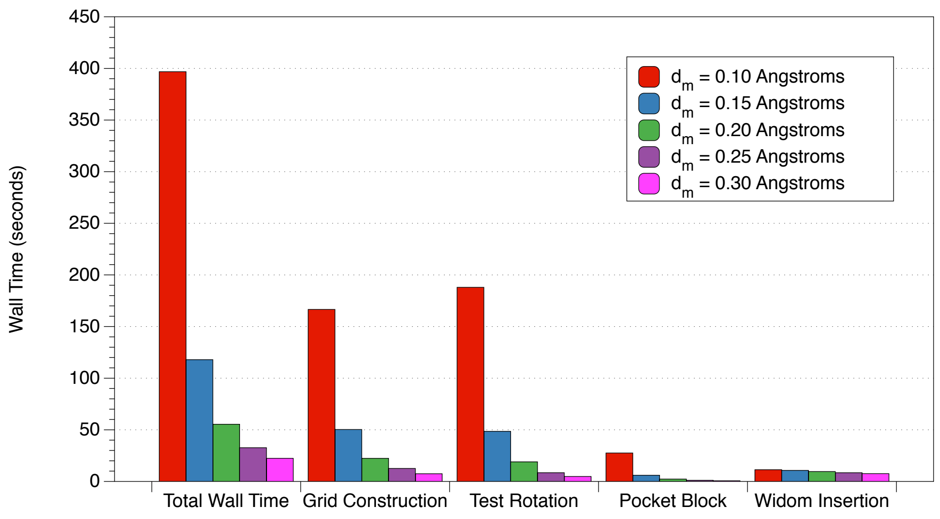


Figure 4: $CO_2$ $K_H$ computational wall time for twenty-one IZA zeolite structures as a function of energy grid mesh size. The computational intensity scales $O(n^3)$ for all routines except for the Widom insertion where $n$ is the number of energy grid points (inversely related to mesh size).

to changes in the grid size. As can be seen from Figure 3, the $K_H$ values overall decrease upon increasing $d_m$. Most of the $K_H$ contributions in the porous materials come from small regions with relatively low energy values due to the exponential Boltzmann term found in the formulation of the Henry coefficient in 1. Because we use linear interpolation functions, it is not possible for the interpolated values to have energy values lower than the sampled grid points (as opposed to another form of interpolating functions such as cubic splines where it is possible for interpolated values to be lower). Thus, for smaller $d_m$, the likelihood of sampling lower energy points increases due to general positive concavity of the energy profile around the local energy minimum regions. The choice of the linear interpolation functions was largely made to reduce the time spent in reading and interpolating the energy grid values. As can be seen from Figure 3, for small grid size, the $K_H$ values do not change much as the percentage difference between $K_H$ at $d_m$ = 0.075Å and at 0.1Å is less than 1 percent. For grid sizes smaller than $d_m$ = 0.075Å, many of the IZA structures suffer from GPU memory allocation errors as the device is bounded by the 3GB DRAM. For zeolites MAR and FAR, the mesh size makes a significant difference as at lower grid resolutions of $d_m$ = 0.25Å, these structures are errorneously considered to be entirely inaccessible, reducing their $K_H$

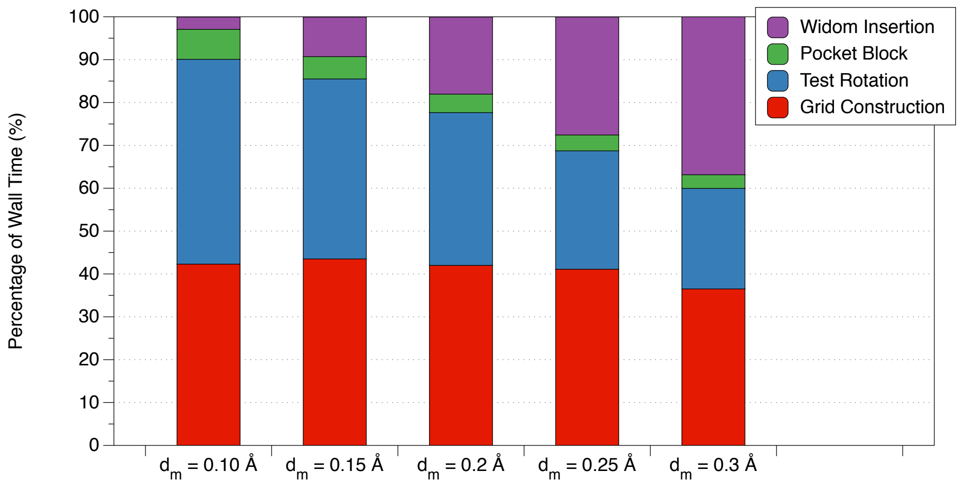


Figure 5: Proportional wall time spent in each of the main simulation routines for twenty-one $CO_2$ $K_H$ calculations at different grid size. From top to bottom are Widom insertion routine, pocket blocking routine, test rotation routine, and energy grid construction routine.

values to nearly zero. The sudden increase in the deviation for these structures demonstrates the importance of setting $d_m$ sufficiently small to avoid inaccurate results. For experimentally known structures, the simulation $K_H$ values can be compared to values derived from the experimental adsorption isotherm data at very low pressure values.

### 3.1.3 Performance: Energy Grid Size

Figure 4 summarizes the computational wall times for twenty-one $CO_2$ $K_H$ and $\Delta h_i$ calculations for different values of $d_m$. As can be seen from Figure 4, the total wall time, $T_{tot}$ decreases significantly upon changing the grid size. At $d_m$ = 0.10, 0.15, 0.20, 0.25, and 0.30Å, $T_{tot}$ = 396.82, 117.93, 55.37, 32.67, and 22.4 seconds, respectively. There is 7.17× wall time improvement upon increasing $d_m$ by two-fold going from 0.10Å to 0.20Å, which is a reasonable number given that the number of grid points reduces by eight-fold, affecting the energy grid, test rotation, and the pocket blocking routine. For the most part, the wall time for Widom insertion routine is independent of the grid size, which is reflected in Figure 5, which shows the proportional wall time spent in each of the routines for different $d_m$. According to Figure 5, the proportional wall time spent in Widom insertion routine increases for larger $d_m$ while its absolute time remains relatively the same (11.4, 10.7, 9.59, 8.44, and 7.55 seconds for $d_m$ = 0.10, 0.15, 0.20, 0.25, and 0.30Å). From Figure 5, it can also be seen that the proportional wall time for the test rotation and the pocket blocking reduces much more significantly (42.4× and 39.2×, respectively) going from $d_m$ = 0.10 to 0.30Å compared to the energy grid routine (22.3×) despite the same reduction in the number of energy grid points in all three routines. It turns out that in the energy grid construction routine, there exists separate computational terms within the Ewald summation that do not scale with the number of grid points and, therefore, the speedup improvement is smaller here compared to the other two routines. Not shown in this work are the performance results for $CH_4$, where similar speedup numbers are obtained by increasing $d_m$. However, given that the test rotation routine is not called in the $CH_4$ calculations, the proportional time spent in the Widom insertions is far larger for $CH_4$ compared to the $CO_2$ calculations for all $d_m$ but especially at large $d_m$.

#### 3.1.4 Performance: Number of Test Rotations $N_{rot}$

Next, we analyze the effect of changing the number of test rotations, $N_{rot}$. As mentioned earlier, point particles such as $CH_4$ do not require rotation moves and thus are omitted in this analysis. We separately analyze the $CO_2$ $K_H$ performance for the 21-member blocking set and the remaining 167-member non-blocking set (i.e., zeolites in which all regions are inaccessible) for IZA structures at $d_m$ = 0.10Å. Figure 6a and 6b show the computational wall times as a function of $N_{rot}$ for the two sets, respectively. In both sets, the qualitative behavior of the curves remains the same: the wall time starts high at small $N_{rot}$, decreases for larger $N_{rot}$ until a minimum wall time ($T_{tot}$ = 328.96 seconds for blocking set and $T_{tot}$ = 1270.43 seconds for non-blocking set) is reached, and then increases monotonically for even larger $N_{rot}$. The minimum wall times, located at $N_{rot}$ = 40 in both sets, constitutes 17.1% and 26.6% reduction from the default value of $N_{rot}$ = 100 rotations. The general behavior of the two curves can be explained by observing the changes in the two routines affected by the number of test rotations: a) pocket blocking and b) test rotation routines. For small $N_{rot}$ – due to the small number of terms that inaccurately captures the true total energy landscape – the energy profile becomes less smooth and results in more disconnected regions inside the simulation box and more pockets. Accordingly, the pocket blocking routine spends more time in the flood fill algorithm and blocking pockets, resulting in longer wall time for the routine at small $N_{rot}$. For larger $N_{rot}$, the energy profile becomes smoother and less time is spent in the flood fill algorithm. Accordingly, the overall pocket blocking routine decreases monotonically with respect to $N_{rot}$. By contrast, the wall time for test rotation predictably scales linearly with respect to $N_{rot}$, and the different behaviors of these two routines contribute to the overall shape of the total wall time curves depicted in 6a and 6b. Additionally, there are small differences between the performance of the blocking and that of the non-blocking sets. In general, the proportional wall time spent in finding and detecting pockets is smaller in the non-blocking set compared to the blocking set; while the flood fill step is performed in both cases, only the blocking set is found to have regions which require blocking. Subsequently, one observes a steeper descent in the wall time curve at smaller numbers of test rotations in 6a compared to 6b as pocket blocking contributes relatively little of the proportional wall time for the non-blocking set.

At the optimal value of $N_{rot} = 40$, the $K_H$ results agree very well (within 0.1%) from results obtained at $N_{rot} = 100$ for both blocking and non-blocking sets that have relatively high $K_H$. For zeolite structures that can be characterized by a very low $K_H$ values (i.e. $K_H$ < 1e-18), the relative $K_H$ difference between $N_{rot} = 40$ and $N_{rot} = 100$ becomes large, but the difference here is uninteresting and ultimately meaningless in practice as very small values of $K_H$ all indicate poor adsorption properties of $CO_2$ regardless of the exact number. Thus, there can be a performance gain by setting $N_{rot} = 40$ from the default value of 100 without having to sacrifice meaningful accuracy in the $K_H$ results.

### 3.1.5 Performance: Pocket Blocking

Next, we discuss the performance details of the pocket blocking routine as a function of the number of CPU cores. Because the pocket blocking is the only routine of this algorithm that takes place entirely on the CPU, we utilize Pthreads to generate multiple CPU threads that work in parallel to accelerate the routine. Based on the profiling reported in Ref,[39] larger numbers of CPU threads

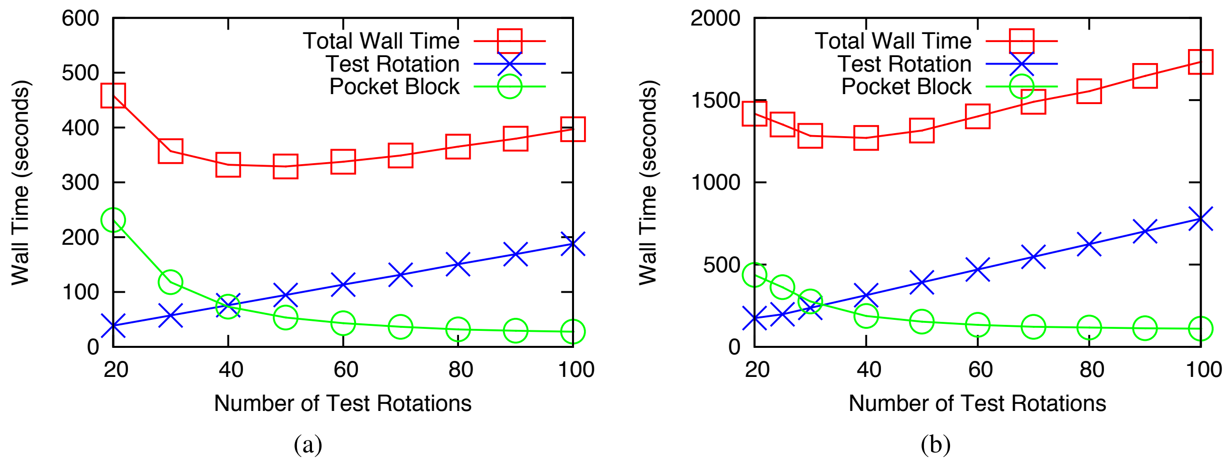


Figure 6: Computational wall time as a function of number of test rotations, $N_{rot}$, for the $CO_2$ $K_H$ calculations for the (a) 21 IZA blocking set and (b) 167 IZA non-blocking set. The optimal number of test rotations that minimizes wall time occurs at $N_{rot} = 40$.

are found to be generally advantageous for larger or more complex zeolite structures, but there exists performance degradation in smaller or more simplistic structures. The relationship between number of CPU threads and the pocket blocking wall time is illustrated in Figure 7. As expected, the wall time decreases monotonically with respect to the number of CPU threads for both the $CH_4$ and the $CO_2$ simulations and saturation point is reached near 4-8 CPU threads. For $CO_2$ simulations, the performance scales nearly linearly going from 1 to 2 CPU threads as a speedup of $1.96\times$ is observed (in $CH_4$, speedup is $1.53\times$).

### 3.1.6 Performance: GPU vs. CPU

Finally, we assess performance differences between the CPU and the GPU. For the $CH_4$ energy grid routine, we observe speedups of around $50\times$ compared to an optimized single CPU core simulation without SSE from simulation results from the Carver cluster at NERSC. The number is reasonable given that the energy grid construction is compute-bound due to its high compute intensity and most of the memory transactions are handled inside the fast constant memory of the GPU. We have not written a CPU version of the code for the $CO_2$ energy grid routine, but we expect the speedup numbers to be similar given the similar compute flop intensity and similar memory references

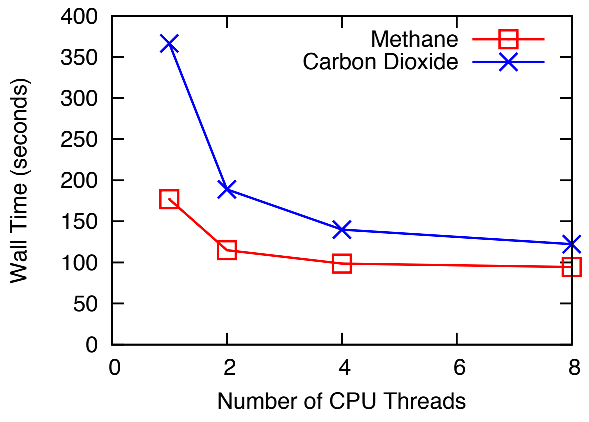
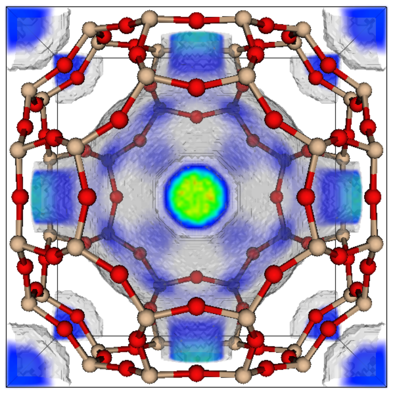


Figure 7: $CH_4$ and $CO_2$ pocket block routine wall time as a function of number of CPU threads for the 188 IZA zeolite structures. The performance gain begins to saturate going from 4 to 8 CPU threads.

and coalescing patterns. Moreover, the inclusion of fast, pocket blocking routine accelerates the detection of inaccessible regions compared to slow methods based on molecular dynamics.

## 3.2 Characterization of Zeolites

Utilizing the total energy grid, we can create another grid (i.e. local $K_{\mathrm{H}}$ grid) that contains the Boltzmann factors at each of the points. The local $K_{\mathrm{H}}$ grid can be used to attain information about the locations likely to be occupied by the gas molecules, providing another means to characterize the structures in the post-processing stage after the simulation. In Figure 8, we include an example illustration that displays the effect of pocket blocking in the zeolite LTA structure. A heat mapping represents the local $CO_2$ $K_{\mathrm{H}}$ regions, with warmer colors denoting larger $K_{\mathrm{H}}$ contributions, and as can be seen, the distribution of the local $K_{\mathrm{H}}$ changes upon enabling/disabling the pocket blocking routine. Specifically in the case of LTA, the corner regions in the unit cell become inaccessible as $CO_2$ cannot enter into this region within a reasonable experimental timescale. Accordingly, the $CO_2$ $K_{\mathrm{H}}$ values inside LTA with or without pocket blocking are $9.59\times10^{-6}$ and $6.85\times10^{-6}$ mol/kg/Pa as the $K_{\mathrm{H}}$ value decrease upon setting the energy grid values inside the inaccessible region to be very high. In general, structures that contain inaccessible regions always show reduction in $K_{\mathrm{H}}$ values as proper inclusion of pocket blocking raises the energy values at these regions. We illustrate by this example the importance of pocket blocking in avoiding the overestimation of the level of adsorption in a given porous material.


(a) LTA pore landscape without blocking

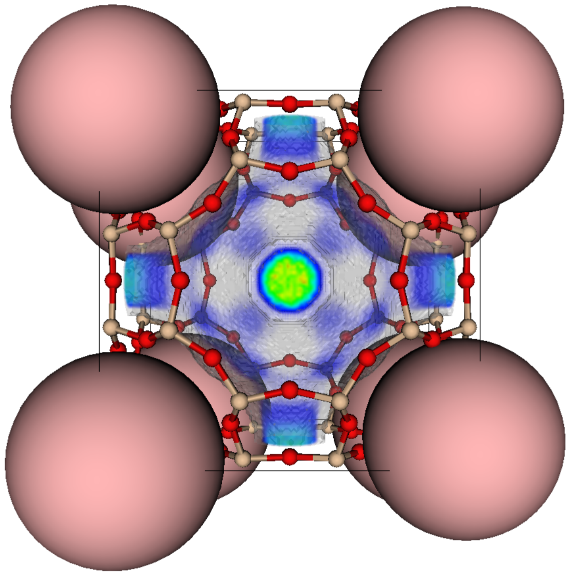

(b) LTA pore landscape with blocking

Figure 8: Snapshots of LTA zeolite with pore landscape (a) without blocking and (b) with blocking. The silicon and oxygen atoms of the framework are shown in tan and red respectively. The local Henry coefficient contributions are shown as a heat map; warmer colors indicate higher likelihood of the guest $CO_2$ molecule occupying these positions. The inaccessible pocket region is shown to be fully excluded by the pink spheres, located on the eight corners of the unit cell. Upon enabling the pocket blocking routine, the energy grid points in the blocking region are set to very high energy and the $CO_2$ molecules are forced away from these regions.

Up until now, the analysis focused on the experimentally verified IZA structures, which comprise of a very small subset of the entire database of the zeolite structures. The algorithm explained earlier can be utilized unchanged to easily extend our simulation code to process 135,222 hypothetical zeolites in the database. Figure 9 plots the histogram of both the $CO_2$ and $CH_4$ $K_H$ values for all of the zeolite structures. Given that zeolites are seen as one of the ideal candidates for carbon capture, it is not surprising that the $K_H$ values for $CO_2$ are in general higher than that of $CH_4$. The broader distributions of the $CO_2$ $K_H$ also indicates that the range of possible structures with different $CO_2$ adsorption properties remains large compared to $CH_4$. Extrapolating from the simulation times obtained from the IZA structures, we can obtain the $CO_2$ $K_H$ values of the entire hypothetical zeolite structures in about 50 hours of total wall time, utilizing 8 Tesla C2050 GPUs in the Dirac cluster. At the end of the $K_H$ calculations, selected structures from a large database that are deemed to have shown good adsorption properties can be analyzed in-depth by utilizing the GPU grand canonical Monte Carlo simulations whose algorithm is described in 2.4. The predicted adsorption properties obtained from the simulation code can provide valuable insights to experimentalists interested in synthesizing materials inside a large porous materials database.

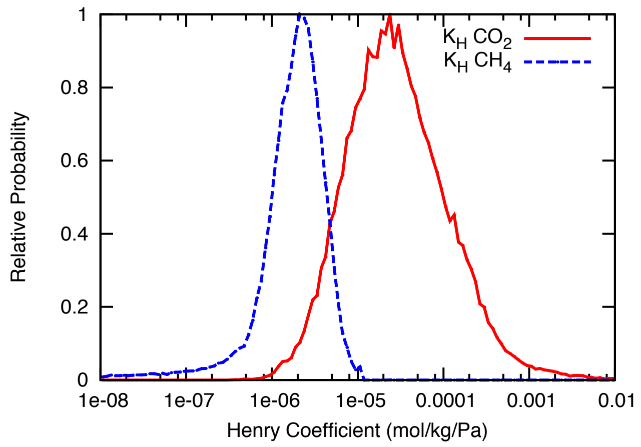


Figure 9: Distribution of $CO_2$ and $CH_4$ Henry coefficient values for 135,222 hypothetical zeolite structures. In general, the $CO_2$ $K_H$ values are larger than the $CH_4$ $K_H$, indicating that zeolites have high selectivity for $CO_2$.

## 4 Summary and Future Work

We have developed a GPU-based simulation code that can characterize and pre-screen a large database of porous materials. Our code has the capability to quickly compute the Henry coefficient and the heats of adsorption values for many different gas molecules immersed inside a porous material. We can further analyze individual structures in the simulations by visualizing local Henry coefficient values to determine local adsorption property of a given material. Although the simulation results presented in this work pertain to zeolites, the code can be easily extended to process other classes of microporous materials. For future work, we plan to analyze the effect of adding cations to zeolites and to characterize a large database of other porous materials such as MOF and ZIF structures.

## Acknowledgements

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.

J.K., R.L.M., O.R and M.H. 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. In addition, R.L.M. and O.R. were supported by the Scientific Discovery through Advanced Computing (SciDAC) program's Visualization and Analytics Center for Enabling Technologies (VACET). M.H. was supported jointly by DOE Office of Basic Energy Sciences and the Office of Advanced Scientific Computing Research through SciDAC project #CSNEW918 entitled "Knowledge guided screening tools for identification of porous materials for CO2 separations". B.S. was supported as part of the Center for Gas Separations Relevant to Clean Energy Technologies, an Energy Frontier Research Center funded by the U.S. Department of Energy, Office of Science, Office of Basic Energy Sciences under Award Number DE-SC0001015.

## References

(1) Auerbach, S.M.; Carrado, K.A.; Dutta, P. K. Handbook of Zeolite Science and Technology; Marcel Dekker: New York, USA, 2004.

(2) Smit, B.; Maesen, T.L.M. *Nature* **2008**, *457*, 671-677.

(3) Smit, B.; Maesen, T.L.M. *Chem. Rev.* **2008**, *108*, 4125-4184.

(4) Krishna, R.; van Baten, J.M. *Chem. Eng. J.* **2007**, *133*, 121-131.

(5) Haldoupis, E.; Nair, S.; Sholl, D.S. *J. Am. Chem. Soc.* **2010**, *132*, 7528-7539.

(6) Haldoupis, E.; Nair, S.; Sholl, D.S. *Phys. Chem. Chem. Phys.* **2011**, *13*, 5053-5060.

(7) Lin, L.-C.; Berger, A.; Martin, R.L.; Kim, J.; Swisher, J.; Jariwala, K.; Rycroft, C.H.; Bhown, A.; Deem, M.W.; Haranczyk, M.; Smit, B. *Nature Materials* submitted.

(8) D'Alessandro, D.M.; Smit, B.; Long, J.R. *Angew. Chem. -Int. Edit.* **2010**, *49*, 6058-6082.

(9) Millward, A.R.; Yaghi, O.M. *J. Am. Chem. Soc.* **2005**, *127*, 17998-17999.

(10) Walton, K.S.; Millward, A.R.; Dubbeldam, D.; Frost, H.; Low, J.J.; Yaghi, O.M.; Snurr, R.Q. *J. Am. Chem. Soc.* **2008**, *130*, 406-407.

(11) Banerjee, R.; Phan, A.; Wang, B.; Knobler, C.; Furukawa, H.; O'Keeffe, M.; Yaghi, O.M. *Science* **2008**, *319*, 939-943.

(12) Sumida, K.; Hill, M.R.; Horike, S.; Dailly, A.; Long, J.R. *J. Am. Chem. Soc.* **2009**, *131*, 15120-15121.

(13) Choi, H.J.; Dinca, M.; Long, J.R. *J. Am. Chem. Soc.* **2008**, *130*, 7848-7850.

(14) (a) Baerlocher, C.; Meier, W.M.; Olson, D.H. Atlas of Zeolite Framework Types, seventh edition. Elsevier: Amsterdam, NL, 2007. (b) http://www.iza-online.org/ (accessed Jan 1, 2010).

(15) (a) Lach-hab, M.; Yang, S.; Vaisman, I.I.; Blaisten-Barojas, E. *Mol. Inf.* **2010**, *29*, 297-301; (b) Carr, D.A.; Lach-hab, M.; Yang, S.; Vaisman, I.I.; Blaisten-Barojas, E. *Micropor. Mesopor. Mat.* **2009**, *117*, 339-349.

(16) Foster, M.D.; Treacy, M.M.J. http://www.hypotheticalzeolites.net (accessed Nov 13, 2009)

(17) Earl, D.J.; Deem, M.W. *Ind. Eng. Chem. Res.* **2006**, *45*, 5449-5454.

(18) Deem, M.W.; Pophale, R.; Cheeseman, P.A.; Earl, D.J. *J. Phys. Chem. C* **2009**, *113*, 21353-21360.

(19) Pophale, R.; Cheeseman, P.A.; Deem, M.W. *Phys. Chem. Chem. Phys.* **2001**, *13*, 12407-12412.

(20) Maginn, E. J.; Bell, A.T.; Theodorou, D.N.; *J. Phys. Chem.* **1995**, *99*, 2057-2079.

(21) Macedonia, M. D.; Maginn, E.J.; *Mol. Phys.* **1999**, *96*, 1375-1390.

(22) Smit, B.; Krishna, R. *Curr. Opin. Solid State Mater. Sci.* **2001**, *5*, 455-461.

(23) Willems, T.F.; Rycroft, C.H.; Kazi, M.; Meza, J.C.; Haranczyk, M. *Micropor. Mesopor. Mat.* **2012** *149*, 134-141.

(24) Wei, J.; Floudas, C.A.; Gounaris, C.E.; Somorjai, G.A. *Catal. Lett.* **2009** *133*, 234-241.

(25) Foster, M.D.; Rivin, I.; Treacy, M.M.J.; Friedrichs, O.D. *Micropor. Mesopor. Mat.* **2006**, *90*, 32-38.

(26) Martin, R.L.; Smit, B.; Haranczyk, M. *J. Chem. Inf. Model.* **2012**, *52*, 308-318.

(27) Owens, J.D.; Luebke, D.; Govindaraju, H.; Harris, M; Krüger, J; Lefohn, A.E.; Purcell, T. *Computer Graphics Forum* **2007**, *26(1)*, 80-113.

(28) Kim, J.; Rodgers, J.; Athènes M.; Smit B. *J. Chem. Theory Comput.* **2011**, *7*, 3208-3222.

(29) Preis, T.; Virnau, P.; Schneider, J.; *J. Comput. Phys.* **2009**, *228*, 4468-4477.

(30) Li, H.; Petzold, L.; *Int. J. High Perform. Comput.* **2010**, *24*, 107-116.

(31) Anderson, A.; Goddard, W.; Schroder, P. *Comput. Phys. Commun.* **2007**, *177*, 298-306.

(32) Garcia-Perez, E.; Parra, J.B.; Ania, C.O.; Garcia-Sanchez, A.; van Baten, J.M.; Krishna, R.; Dubbeldam, D.; Calero, S. *Adsorption* **2007**, *13*, 469-476.

(33) Dubbeldam, D.; Calero, S.; Vlugt, T.; Krishna, R.; Maesen, T.; Beerdsen, E.; Smit, B. *Phys. Rev. Lett.* **2004**, *93*, 088302.

(34) (a) Dubbeldam, D. and Smit, B. (2003). J. Phys. Chem. B. **107**:12138.; (b) Bates, S.P., v. Well, W.J.M., v. Santen, R.A. and Smit, B. (1996). *J. Am. Chem. Soc.* **118**:6753.

(35) Krishna, R.; van Baten, J.M. *Langmuir* **2010**, *26*, 2975-2978.

(36) Keffer, D.; Gupta, V.; Kim, D.; Lenz, E.; Davis, H. T.; McCormick, A. V. *J. Mol. Graph.* **1996**, *14*, 108-116.

(37) Haranczyk, M.; Sethian, J.A. *J. Chem. Theory Comput.* **2010**, *6*, 3472-3480.

(38) Haranczyk, M.; Sethian, J.A. *Proc. Natl. Acad. Sci. USA (PNAS).* **2009**, *106*, 21472-21477.

(39) Martin, R.L.; Prabhat; Donofrio, D.; Sethian, J.A.; Haranczyk, M. *J. High Perform. Comput. Appl.* in press, DOI: 10.1177/1094342011431591.

(40) Toukmaji, A.Y.; Board, J.A.; *Comput. Phys. Commun.* **1996**, *95*, 73-92.