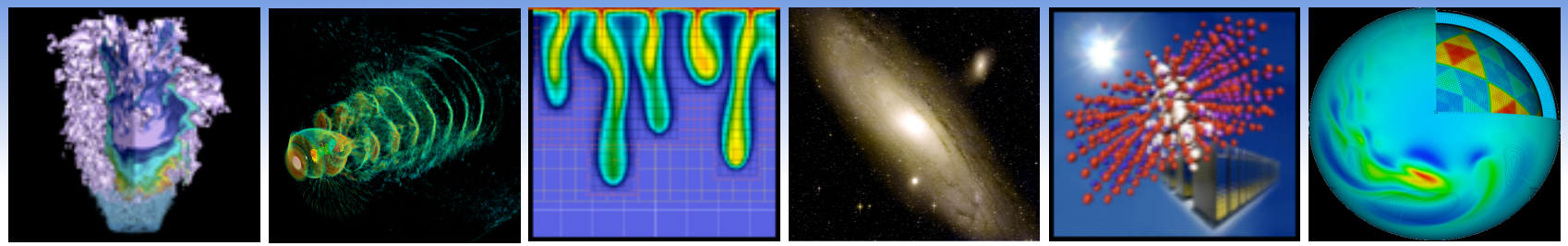


GPU Monte Carlo Algorithms for Molecules within a Microporous Framework

Jihan Kim

ICCS Workshop 1/26/11

U.S. DEPARTMENT OF ENERGY | Office of Science

NERSC

BERKELEY LAB

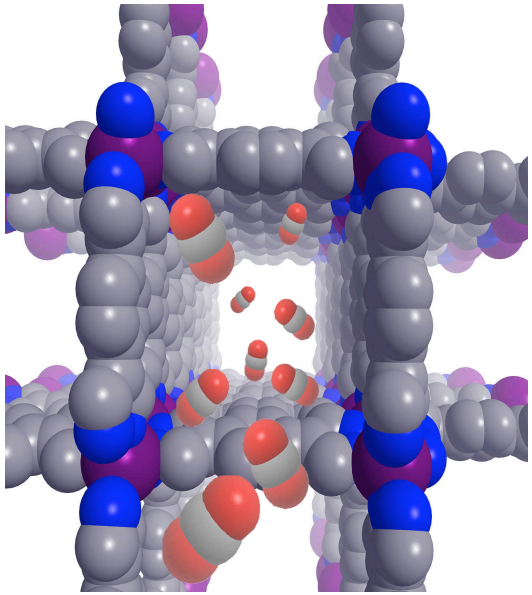
NATIONAL ENERGY RESEARCH SCIENTIFIC COMPUTING CENTER

Outline

- Introduction
- Computational Setup (Methane – MFI system)
- GPU Monte Carlo Algorithms
- Results
- Conclusion and Future Work

Carbon Capture and Sequestration

- Goal: separate CO_2 molecules from power plant flue gases and bury them underground
- 46 Energy Frontier Research Centers (EFRC) established by DOE to tackle this and other energy issues: $777 million dollars over five years
- Collaboration with Prof. Berend Smit (UC Berkeley Chemical Engineering): simulate mobile molecules contained in host frameworks
- Millions of frameworks (need fast simulation)



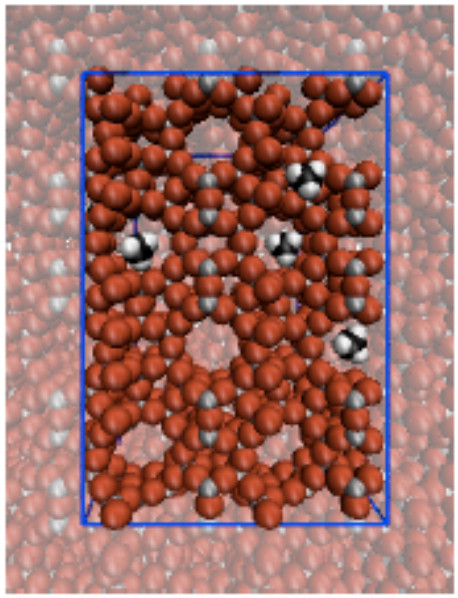
CO_2 molecules: red-grey-red rods and metal-organic framework
Consisting of cobalt atoms (purple)
Linked by an organic bridging ligand
(D. M. D'Alessandro, B. Smit, J. Long "Carbon Dioxide Capture: Prospects for New Materials)

Focus on Canonical Monte Carlo

- Constant number (N), volume (V), and temperature (T): compute total energy
- Mobile methane molecules within an immobile MFI framework
- Interaction modeled with Lennard-Jones potential with cutoff radius, R = 12 Å

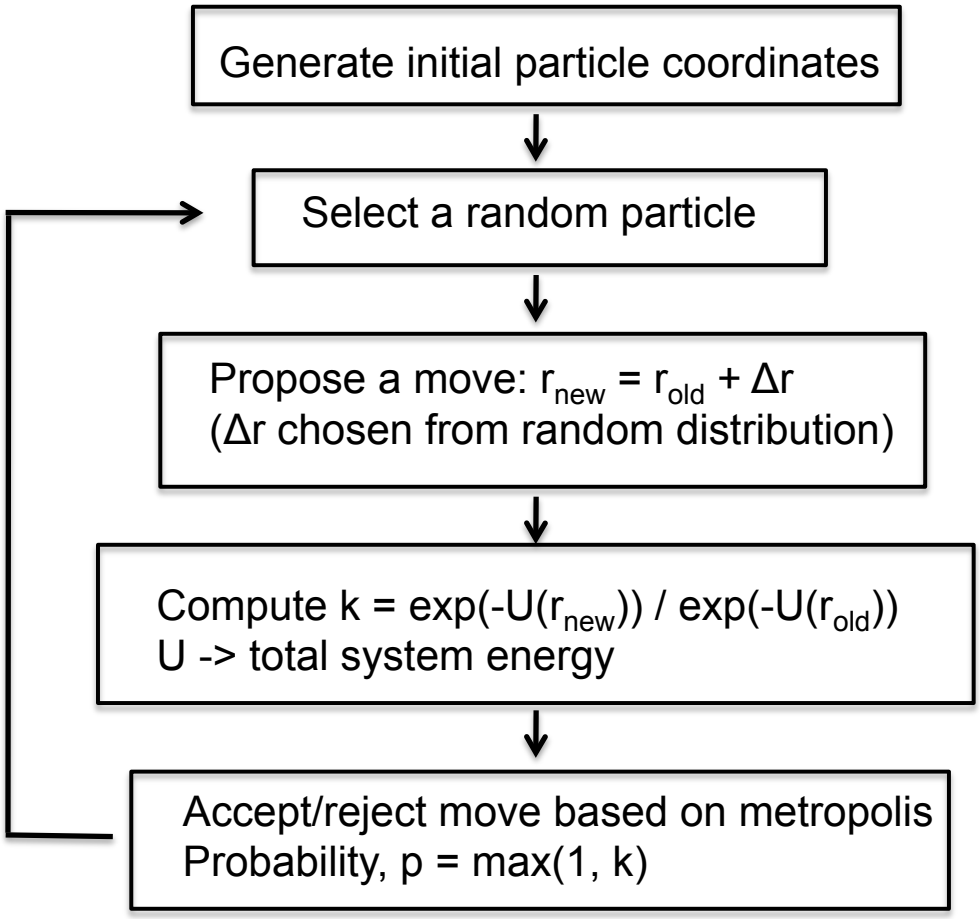
$$U_{jk}(r_{jk}) = 4\epsilon[(\frac{\sigma}{r_{jk}})^{12} - (\frac{\sigma}{r_{jk}})^{6}], \qquad r_{jk} < R$$
$$U_{jk}(r_{jk}) = 0, \qquad r_{jk} \geq R$$

- Periodic boundary condition – small system size (V = 40 x 40 x 26 $Å^3$, hundreds of methane molecules)
- Energy grid (512x512x256) used to compute gas-framework interaction – use interpolation functions



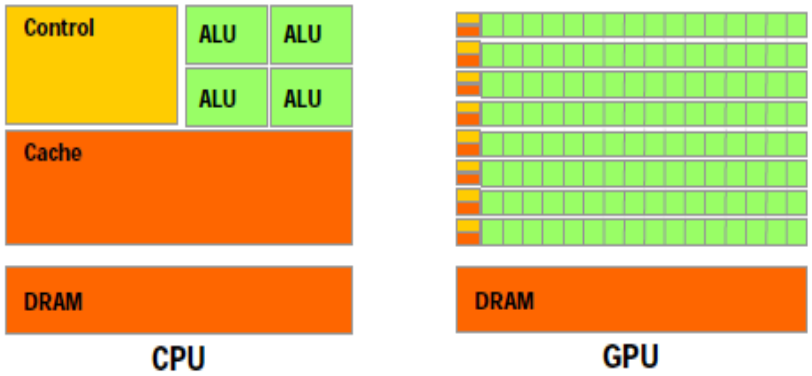
2D graphic illustration of the methane (silver and black), MFI (red) system
Image courtesy of Dr. Jocelyn Rodgers

Quick Summary of Markov Chain Monte Carlo Algorithm



Use GPUs to obtain Speedup

- GPU (graphics processing units) : More transistors devoted to data computation (CPU: cache, loop control)
- GPU simulations: Dirac GPU Cluster at NERSC (44 Fermi Tesla C2050 GPU cards - 448 CUDA cores, 3 GB GDDR5 memory, PCIe x16 Gen2), double-precision, CUDA C
- CUDA 3.2 (CURAND Library)
- CPU simulations: Carver cluster at NERSC (2 quad-core Intel Nehalem 2.67 GHz), Intel 11.1 Compiler

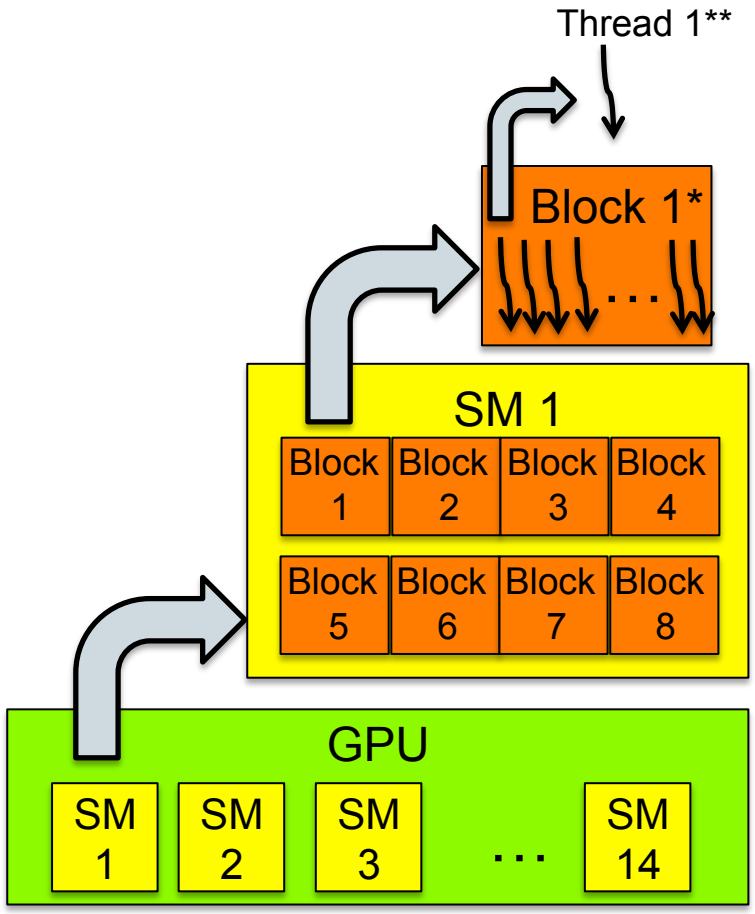


NVIDIA CUDA Programming Guide

Dirac GPU Cluster (NERSC)

Thousands of CUDA Threads: Parallelization Strategies?

Small system size: conduct multiple, independent MC simulations side-by-side



(1) CUDA block per system*

- CUDA threads work together to process the same system: (a) parallel Lennard-Jones (b) waste recycling Monte Carlo
- Total number of independent methane – MFI system: (# CUDA blocks per SM) x (# of SM)
- Utilize fast GPU memory
- Large system size

(2) CUDA thread per system**

- Embarrassingly parallel problem: no communication between threads
- Total number of independent methane – MFI system: (# CUDA threads per block) x (# CUDA blocks per SM) x (# of SM)
- Cannot utilize fast GPU memory
- Small system size (large DRAM usage)

Method 1: Parallel Lennard-Jones MC

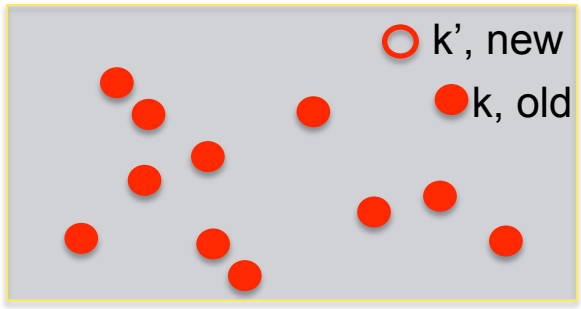
- **Lennard-Jones (LJ) pair potential kernel: bottleneck routine**

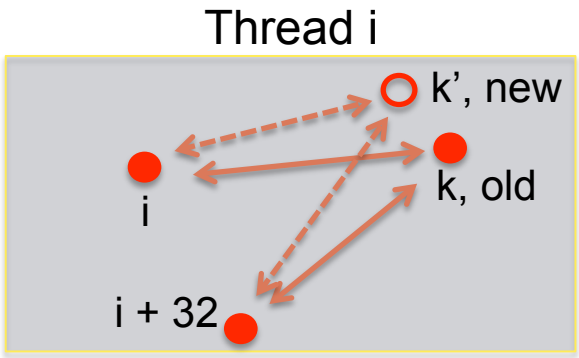
$$U_{jk}(r_{jk}) = 4\epsilon[(\frac{\sigma}{r_{jk}})^{12} - (\frac{\sigma}{r_{jk}})^{6}], \qquad r_{jk} < R$$
$$U_{jk}(r_{jk}) = 0, \qquad r_{jk} \geq R$$

- **Threads in the same CUDA block share work to parallelize LJ**

$$E_{\text{total,new}} = E_{\text{total,old}} + \sum^{N_{\text{tot}}} U_{jk'}(r_{jk'}) - \sum^{N_{\text{tot}}} U_{jk}(r_{jk}) + E_{\text{grid,k'}} - E_{\text{grid,k}}$$

- **Thread i is responsible for pair potential calculation of particle i, i +32, i+2*32, … to ensure memory coalescing**
- **Combine partial results at the end to obtain new total energy using shared memory**





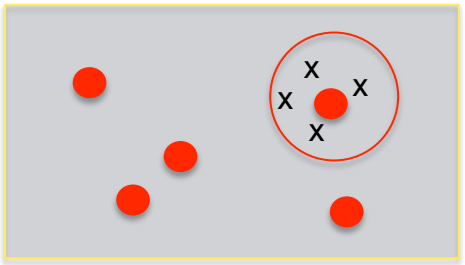
Method 2: Waste Recycling MC*

- **Multi-proposal Monte Carlo – each thread generates its own displacement moves**
- **Only one of these moves (including old state) will be accepted**
- **Utilize "waste" by incorporating information gathered from other, rejected proposals**

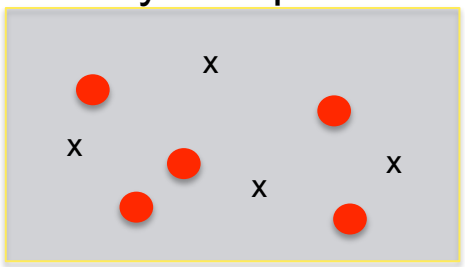
$$S_E \rightarrow S_E + \frac{\sum_i w_i E_i}{\sum_i w_i} \qquad w_i = \exp(-\beta E_i)$$

- **Two kinds of waste recycling MC: (1) displacement, (2) uniformly sampled**

Displacement WRMC



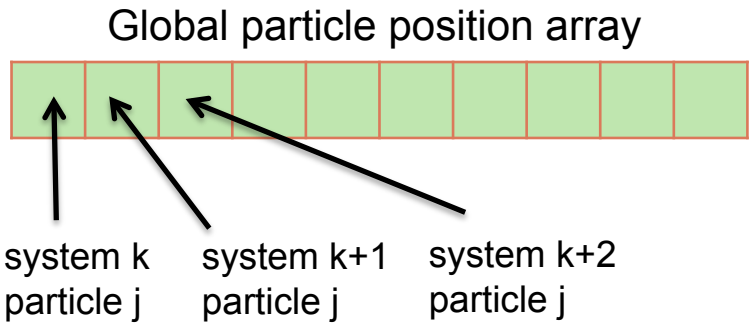
Uniformly Sampled WRMC



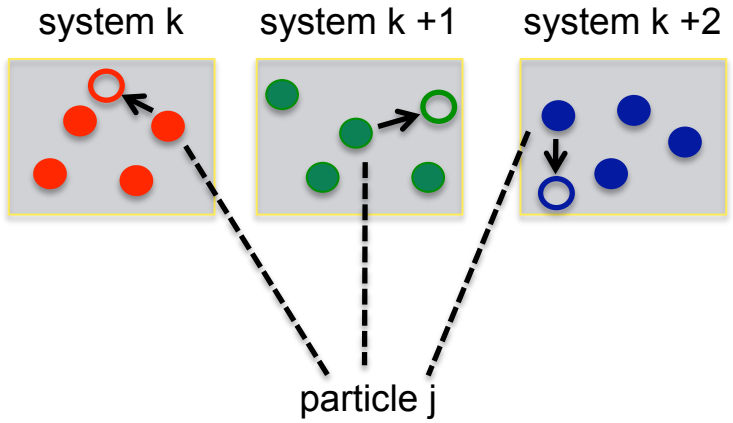
*D. Frenkel. "Speed-up of Monte Carlo simulations by sampling of rejected States" Proceedings of the National Academy of Sciences of the United States of America 101.5 (Dec. 2004) pp. 17571-17575.

Method 3: Embarrassingly Parallel MC

- Each CUDA thread is responsible for conducting its own independent methane – MFI system
- Memory coalescing strategy (1): particle position data layout, avoid warp divergence for LJ memory transactions

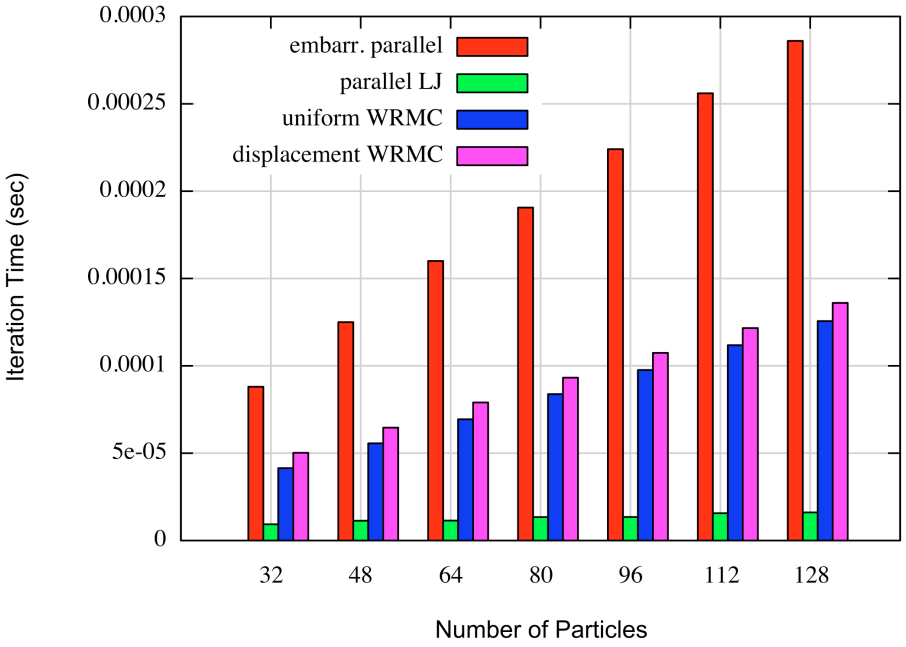


- Memory coalescing strategy (2): choose the same particle index for all CUDA threads in each MC step, one memory transaction for particle translation proposals

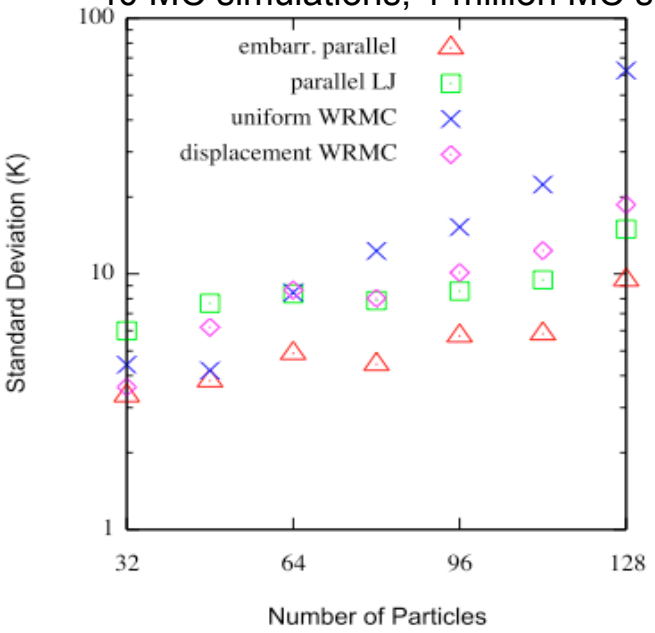


Results

CUDA Block size = 32, one block per SM



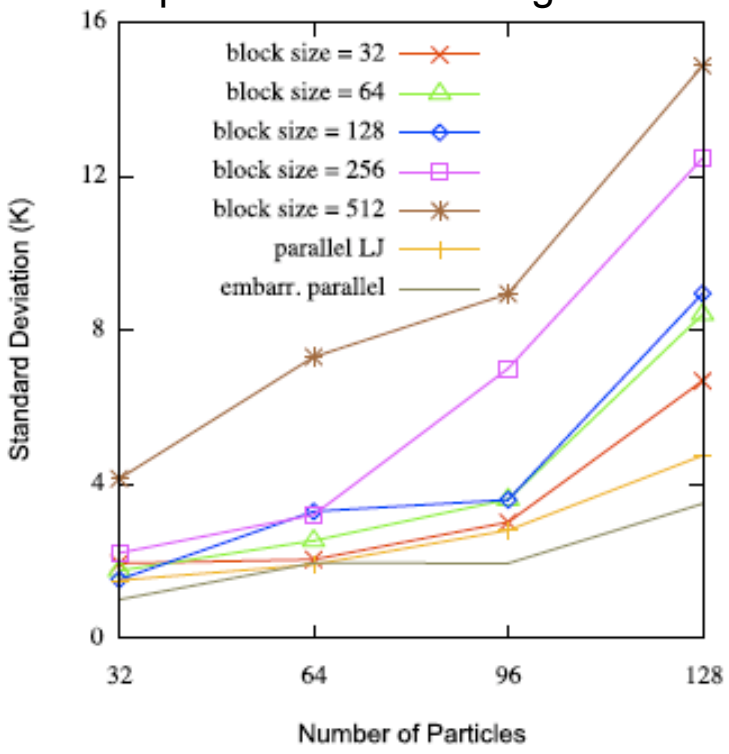
40 MC simulations, 1 million MC step



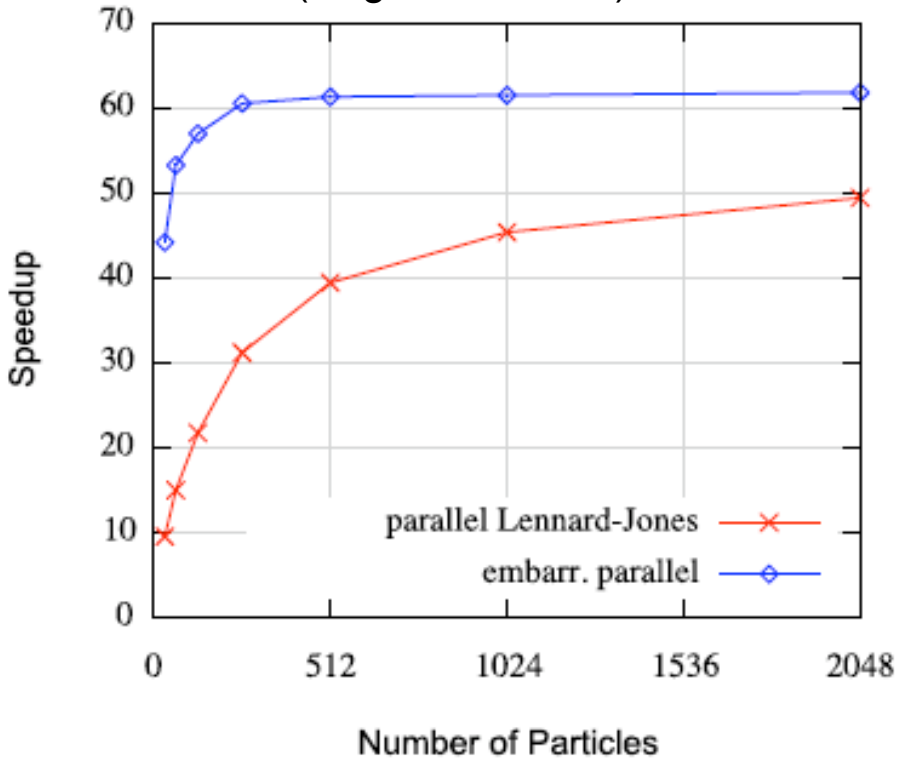
- **Embarrassingly parallel MC: best performance**
- **Displacement WRMC > Uniform WRMC at denser systems**
- **Performance of parallel LJ MC should become similar to embarrassingly parallel MC at larger system size**

Optimized Results and Speedup over CPU

Optimized Block configurations



CPU (single CPU core) vs GPU



Speedup = GPU iteration time / CPU iteration time

NATIONAL ENERGY RESEARCH SCIENTIFIC COMPUTING CENTER

Conclusion and Future Work

- Three approaches to GPU canonical Monte Carlo simulations: (1) parallel LJ, (2) waste-recycling (3) embarrassingly parallel
- Embarrassingly parallel: best performance (limitation is at large system size, not enough GPU DRAM)
- GPU Grand canonical Monte Carlo (GCMC): vary the number of particles during MC simulation (add insertion/deletion moves)
- Obtain adsorption isotherm (number of molecules as a function of pressure)

Main Collaborators

- Prof. Berend Smit (UC Berkeley)
- Dr. Jocelyn Rodgers (LBNL)
- Weekly meetings (Tuesday 2pm @ Gilman Hall – UC Berkeley)