Molecular Dynamics

Alex Melnick and Maura Mulligan

EC 527 Final Project

Looking at Memory

LJ Approximation for Liquid Argon:

$$\ddot{\vec{r}}_k(t) = \vec{a}_k(t) = \sum_{i<j} \vec{r}_{ij}(t) \left(-\frac{1}{r}\frac{du}{dr} \right)_{r=r_{ij}(t)} \left(\delta_{ik} - \delta_{jk} \right)$$

$$-\frac{1}{r}\frac{du}{dr} = \frac{48}{r^2}\left(\frac{1}{r^{12}} - \frac{1}{2r^6} \right)$$

Velocity - Verlet Algorithm:

Given $\left(\vec{r}_i(t), \vec{v}_i(t)\right)$,

1. Compute $\vec{a}_i(t)$ as a function of $\{\vec{r}_i(t)\}$,
2. $\vec{v}_i(t+\frac{\Delta}{2}) \leftarrow \vec{v}_i(t) + \frac{\Delta}{2}\vec{a}_i(t)$,
3. $\vec{r}_i(t+\Delta) \leftarrow \vec{r}_i(t) + \vec{v}_i(t+\frac{\Delta}{2})\Delta$,
4. Compute $\vec{a}_i(t+\Delta)$ as a function of $\{\vec{r}_i(t+\Delta)\}$,
5. $\vec{v}_i(t+\Delta) \leftarrow \vec{v}_i(t+\frac{\Delta}{2}) + \frac{\Delta}{2}\vec{a}_i(t+\Delta)$

Data Structures:

r[nAtoms][3];
rv[nAtoms][3];
ra[nAtoms][3];

“Array of Atom Structs” would be wasting ⅓ of cache block for each calculation.

Serial Baseline Implementation - $O(n^2)$

Initialization: atomic positions in a lattice formation, velocity is randomized, acceleration calculated

For each step in time (and each dimension):

1. Update half of velocity // O(n)
2. Update position // O(n)
3. **Calculate acceleration** **// $O(n^2)$**
   a. **Reset acceleration to zero**
   b. **Accumulate force contribution of every other atom (AI = 6)**
4. Update half of velocity // O(n)

	Baseline	N3L	Cells	OpenMP - Base	OpenMP - N3L	GPU - Base
Runtime(s)	12.2161					

Newton's 3rd Law - $O(n^2)$

Newton's 3rd Law: "for every action, there is an equal and opposite reaction"

- When we calculate the LJ Potential for one atom in a pair, we get the other for free
- Simply initialize inner for-loop to value of outer for-loop (i < j)

Other Optimizations:

- Compare squared distance between atoms to squared cut-off distance (significantly reduces square root calculations)
- Pulled calculations out of conditionals/loops when possible
- Read and write to local variables when possible

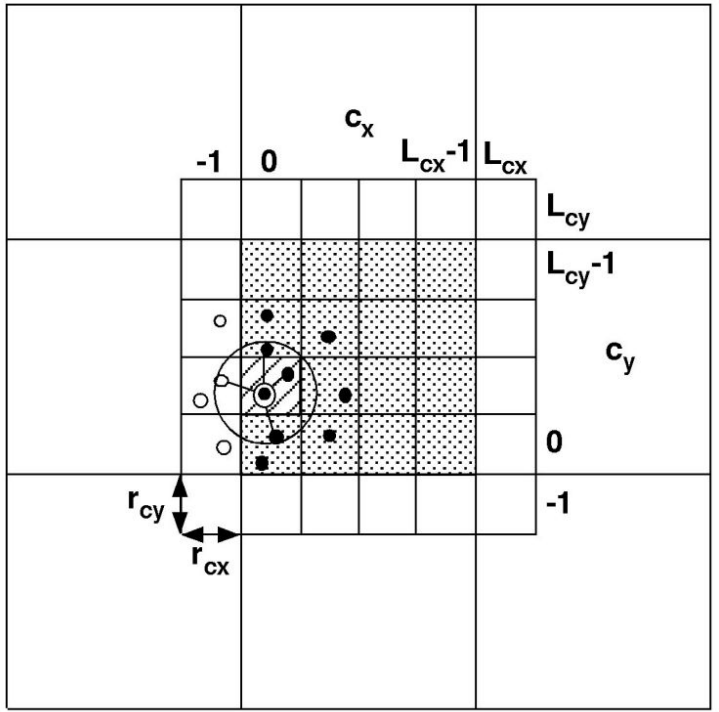
	Baseline	N3L	Cells	OpenMP - Base	OpenMP - N3L	GPU - Base
Runtime(s)	12.2161	5.7002				

Cell “List” - O(n)

Make each “cell” just larger than the cutoff, then we only need to check 27 neighboring cells

To maximize memory efficiency:

- Pseudo linked-list with “head” and “tail” pointer (index) stored in an array
- Each cell “head” starts 2x an even distribution of atoms across cells from previous
  - Gives room for movement of atoms each iteration while leveraging sequential memory of arrays
- Compress array every 5 steps
  - Trade-off for not using linked-lists



	Baseline	N3L	Cells	OpenMP - Base	OpenMP - N3L	GPU - Base
Runtime(s)	12.2161	5.7002	0.4142			

Parallelization - Multithreading

Parallelized with OpenMP - gives us the flexibility to parallelize many different techniques

- Utilized private acceleration array for each thread to avoid race conditions or slow atomic operations
- Threads were synchronized after each timestep to combine acceleration values (before using to update other variables)
- All other data partitioned automatically by OpenMP (Strip-Mining)

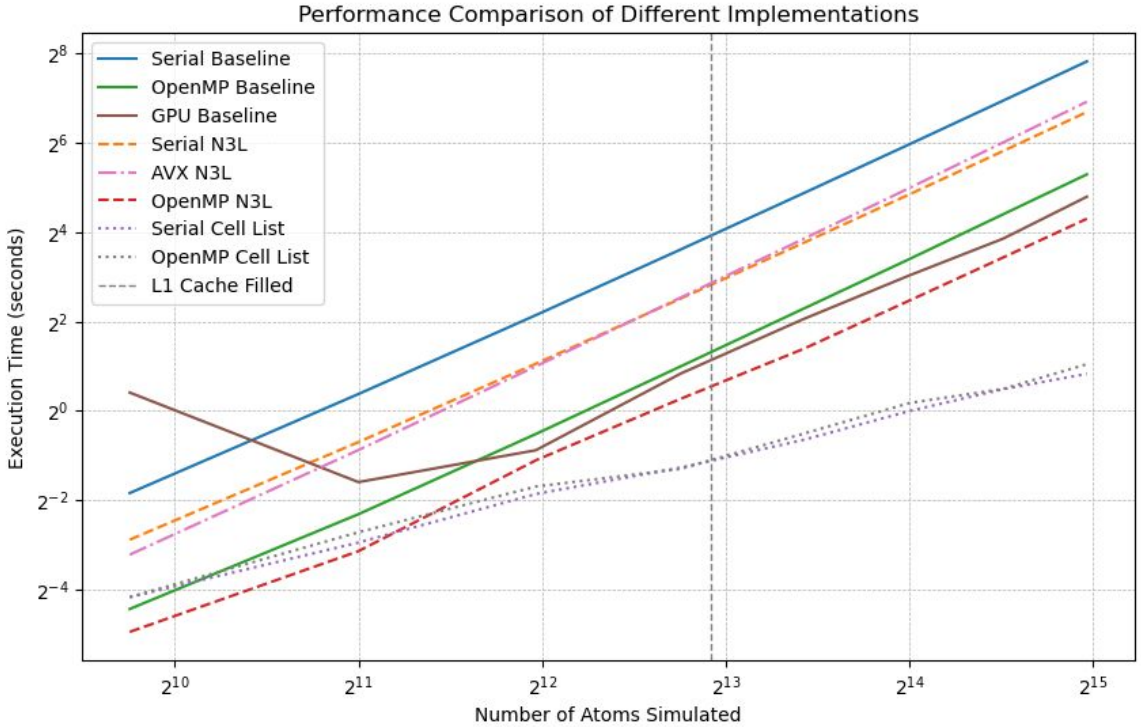
	Baseline	N3L	Cells	OpenMP - Base	OpenMP - N3L	GPU - Base
Runtime(s)	12.2161	5.7002	0.4142	1.9928	1.2048	

Parallelization - GPU

- Utilized multiple kernels to parallelize all steps - no need to transfer data back to CPU between timesteps or kernel calls
  - Sync blocks between each step in algorithm
- Constant 1-D block length, 1-D grid length determined by the number of atoms in the simulation
- All data allocated via Strip Mining

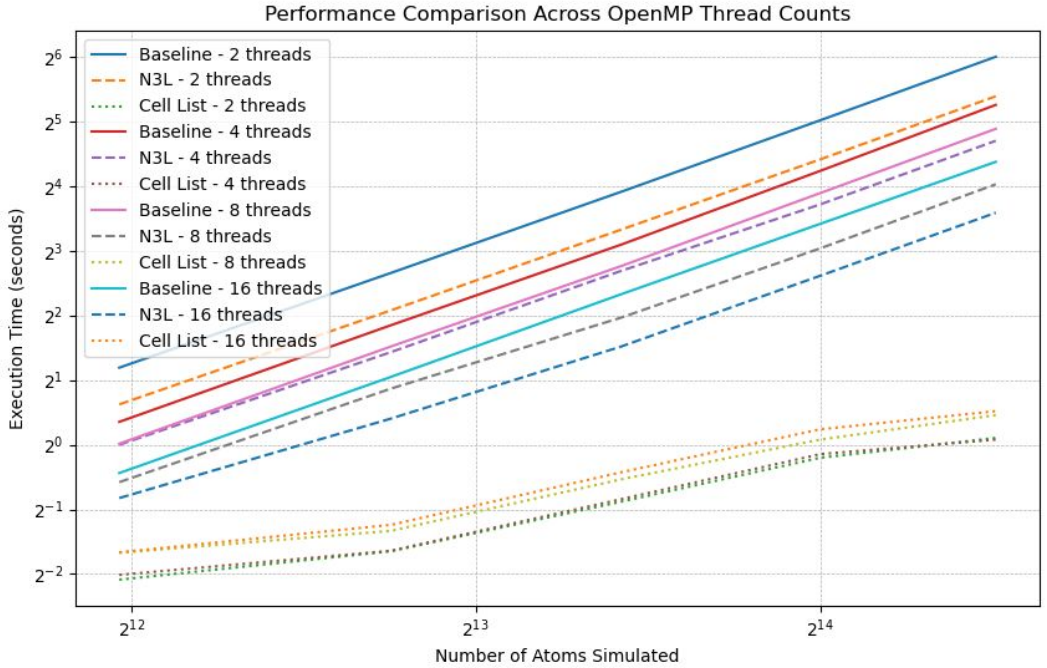
	Baseline	N3L	Cells	OpenMP - Base	OpenMP - N3L	GPU - Base
Runtime(s)	12.2161	5.7002	0.4142	1.9928	1.2048	1.7827

Results of Varying Array Size



*864 to 32,000 Atoms (constant density ⇒ increase overall lattice size) - 72 Bytes / Atom
**All tests ran on Dell XPS 9520 with Intel i7-12700H CPU (20 Threads) and NVIDIA RTX 3050TI Mobile

Results of Varying Thread Counts



*4000 to 23328 Atoms (constant density ⇒ increase overall lattice size) - 72 Bytes / Atom
**All tests ran on Dell XPS 9520 with Intel i7-12700H CPU and NVIDIA RTX 3050TI Mobile

References

[1] A. Nakano (2024). University of Southern California CSCI 596 Scientific Computing and Visualization: Molecular Dynamics Basics [Lecture notes]. Available: https://aiichironakano.github.io/cs596/01MD.pdf

[2] A. Nakano (2024). University of Southern California CSCI 596 Scientific Computing and Visualization: Linked-List Cell Molecular Dynamics [Lecture notes]. Available: https://aiichironakano.github.io/cs596/01-1LinkedListCell.pdf