HW 2 writeup

Aaron Li (al2633)
Perlmutter Username: aaronhli

Data Structures & Approach

To achieve linear scaling, I partitioned the grid into a grid of square cells whose side length was 16x the size of the cutoff distance. To avoid repeatedly scanning neighboring cells, I used ghost cells, where particles near cell boundaries are replicated into neighboring cells within a width equal to the cutoff.

Cells were stored using a packed array representation, in a somewhat comparable manner to CSR. I stored:

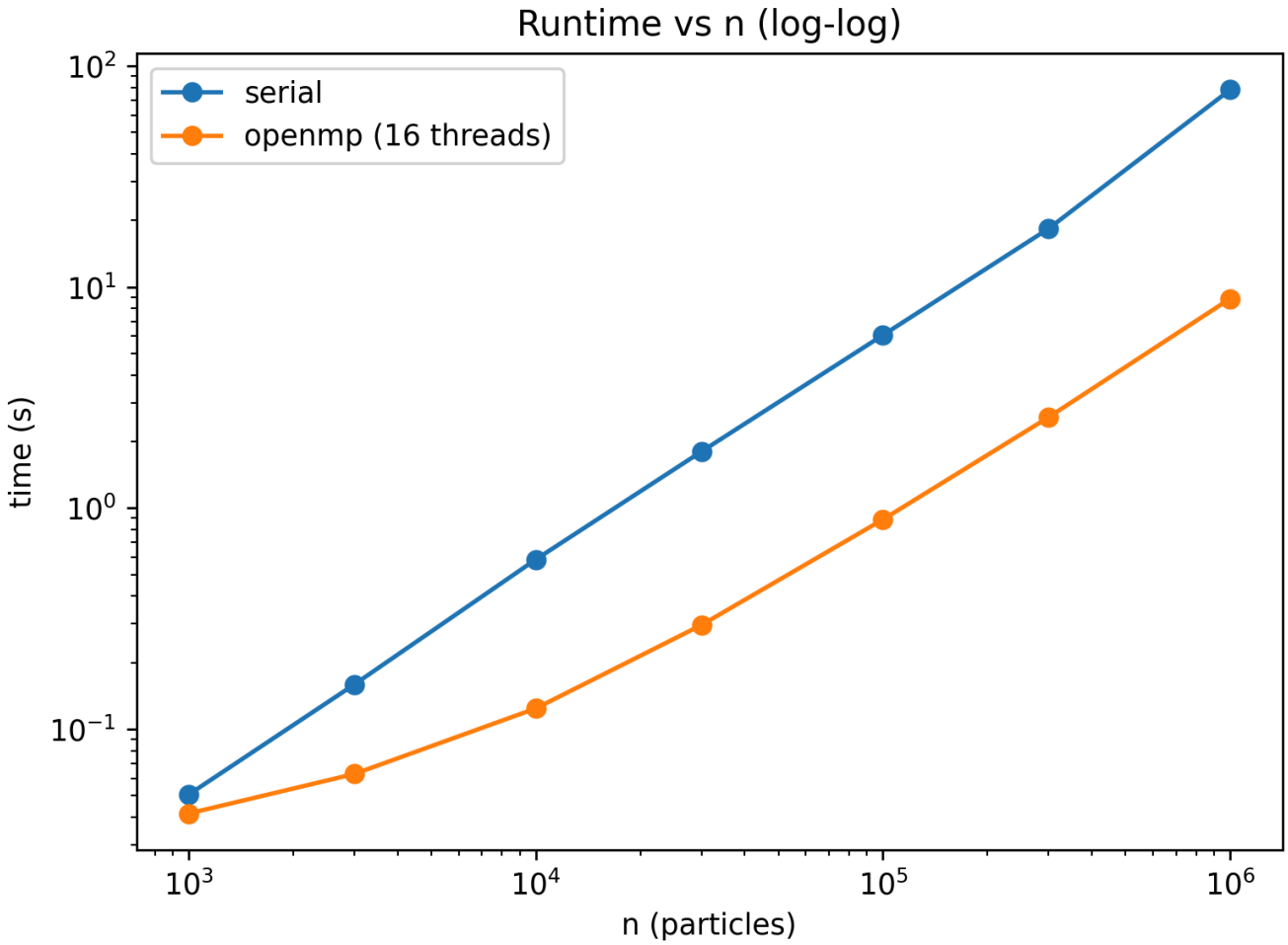
- The actual particle data in a contiguous array
- The number of particles in each cell in
- The start index of each cell

Force computation is easily parallelized over grid cells. Since each cell computes its own local interactions, and ghost particles are read-only, the force loops parallelize cleanly across cells.

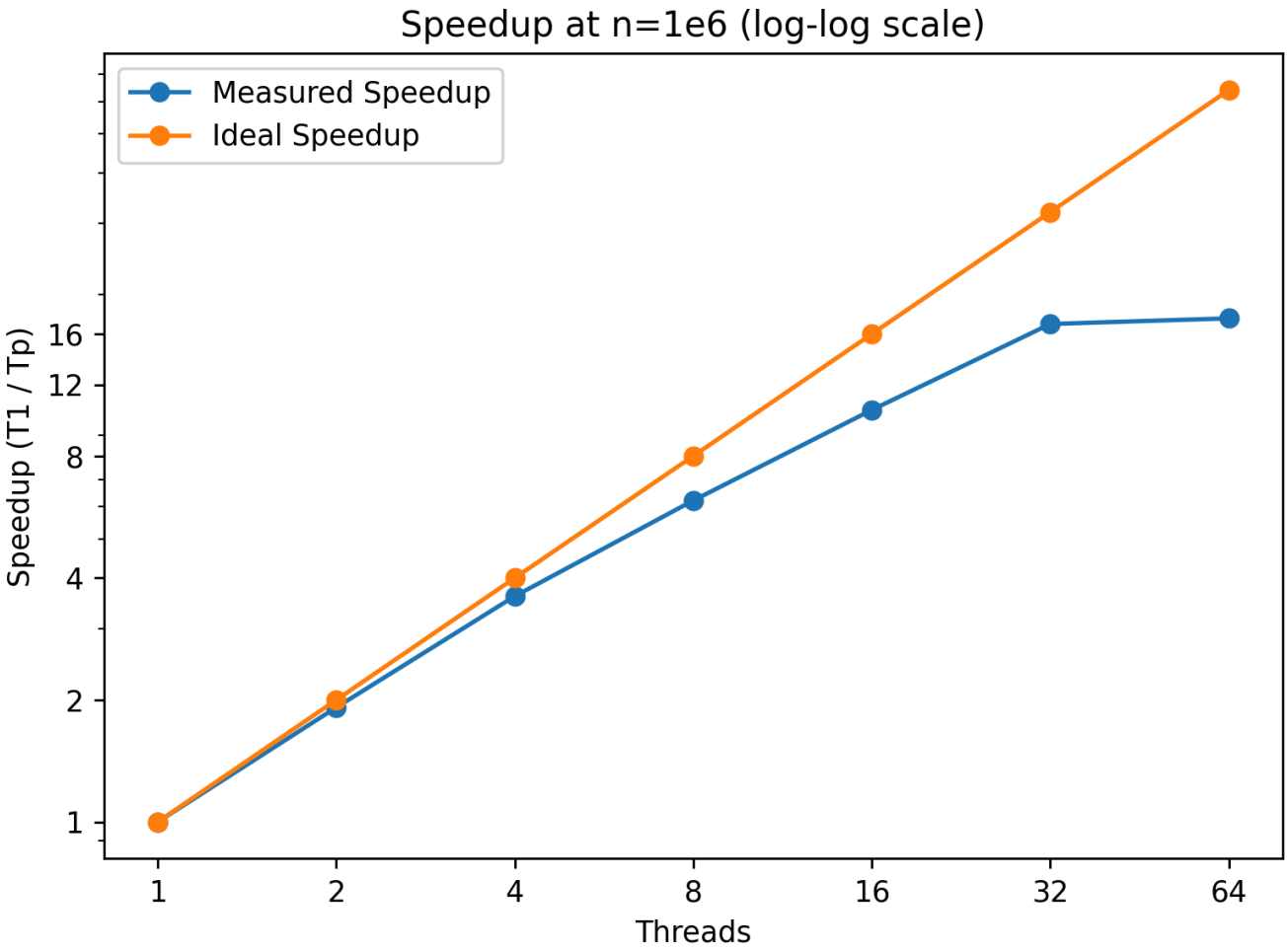
To parallelize the construction of the cell data structure, I use per-thread temporary count arrays. Each thread counts how many particles fall into each cell using its own private counters. These per-thread counts are then combined to compute the global counts and prefix offsets for each cell. From these offsets, each thread is assigned a disjoint region of the packed array for each cell, so threads can write particle indices into the final array without locks or atomics. This also avoids running into cache coherence problems, as the vast majority of the computation is performed on thread-local scratch.

Results:

The log–log runtime plot shows both the serial and OpenMP implementations scale approximately linearly with the number of particles.



The strong scaling plot at $n = 10^6$ shows near-linear speedup up to moderate thread counts, but with diminishing returns after that. **Note: I found around 30% speedup at higher thread counts by using** `OMP_PROC_BIND=close`



GenAI:

I used AI to help me with implementation of the compressed format, and it helped suggest that I add a few intermediate buffers for performance and simplicity. I also had it help me understand the various options I had with regards to openMP. It suggested using `OMP_PROC_BIND=close` for performance in parallel testing. I also used it to assist with plots and formatting.