Princeton GPU Hackathon

EcoSLIM

Members

Jun Zhang, The University of Arizona
Hoang Tran, Princeton University
Chen Yang, Princeton University

Mentors

Troy Comi, Princeton University
Carl Ponder, NVIDIA

Progress and Goals

- **What have you accomplished since June 8?**

➢ Fixed the race condition by splitting the compaction kernel to two.

➢ Tested the serial code and the updated parallel code for different number of particles

➢ Tested the code with MPI

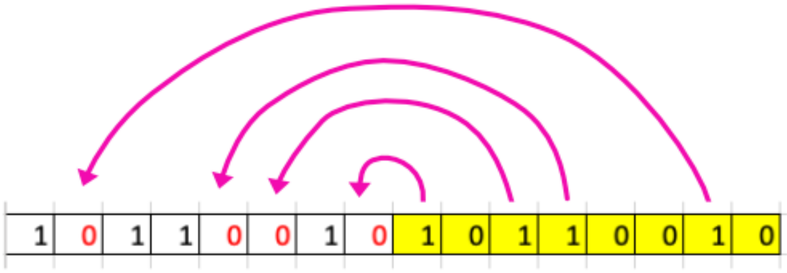
➢ Used cmake and figured out the new structure of the whole code.

- **What are your goals for the day?**

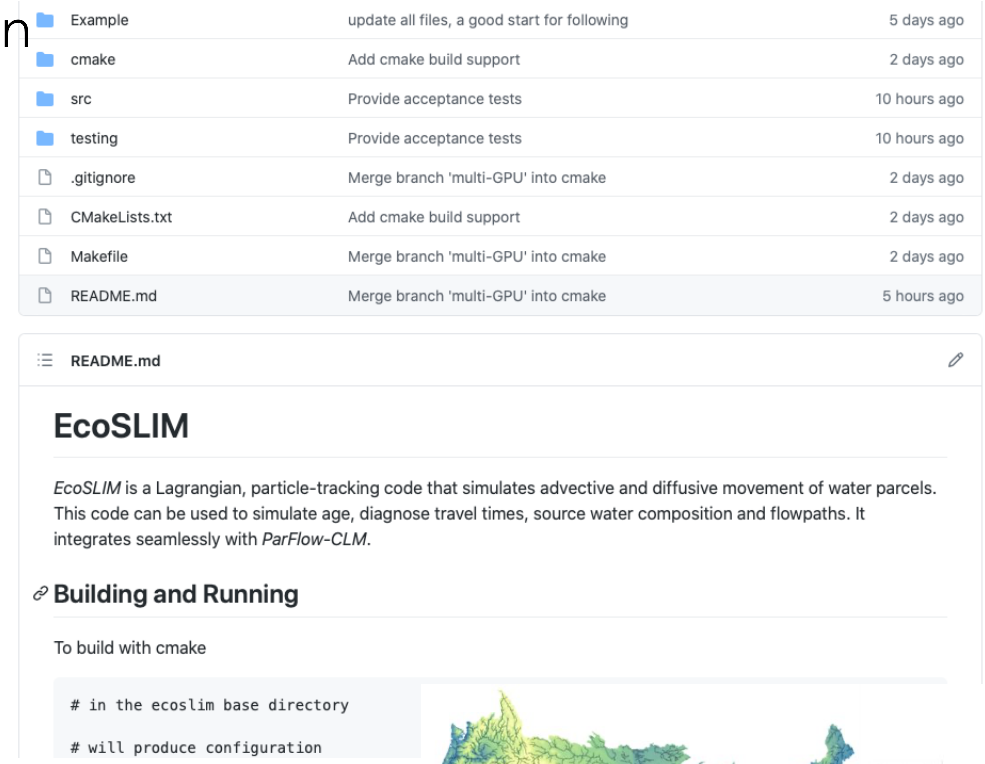
➢ Use Nsight to profile the code

➢ Work with Troy for refactoring code

➢ Present the new structure of the whole code



In-place compaction



Example	update all files, a good start for following	5 days ago
cmake	Add cmake build support	2 days ago
src	Provide acceptance tests	10 hours ago
testing	Provide acceptance tests	10 hours ago
.gitignore	Merge branch 'multi-GPU' into cmake	2 days ago
CMakeLists.txt	Add cmake build support	2 days ago
Makefile	Merge branch 'multi-GPU' into cmake	2 days ago
README.md	Merge branch 'multi-GPU' into cmake	5 hours ago

README.md

EcoSLIM

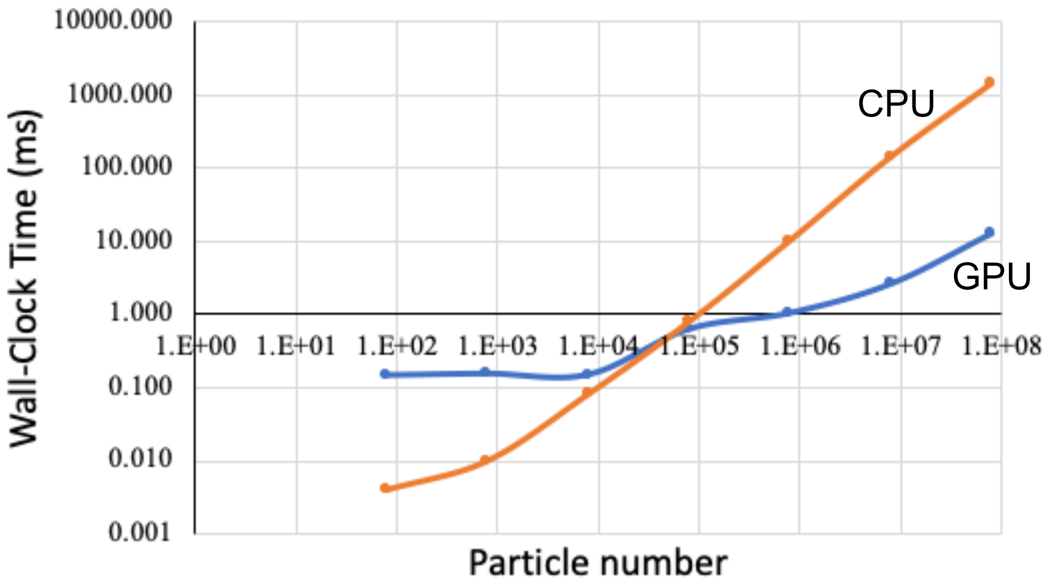
EcoSLIM is a Lagrangian, particle-tracking code that simulates advective and diffusive movement of water parcels. This code can be used to simulate age, diagnose travel times, source water composition and flowpaths. It integrates seamlessly with *ParFlow-CLM*.

Building and Running

To build with cmake

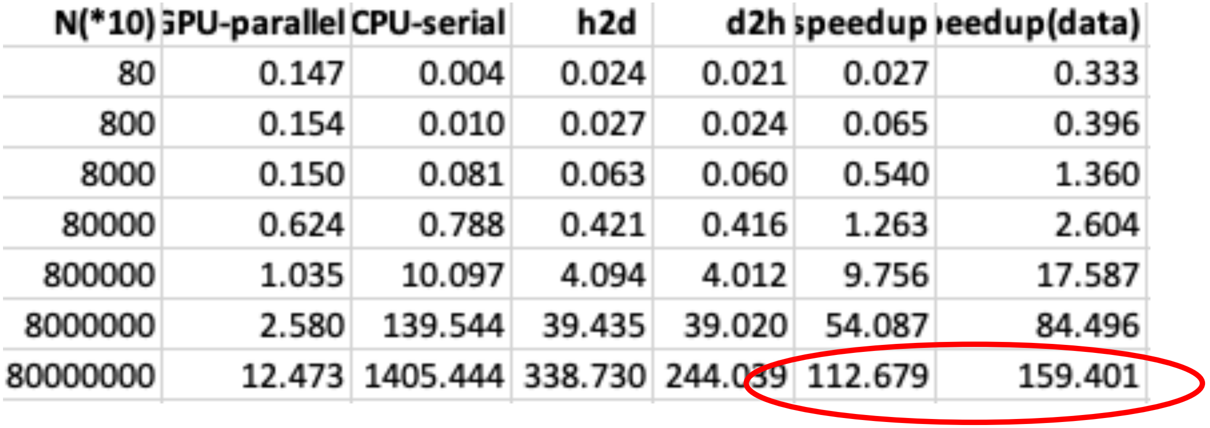
```
# in the ecoslim base directory

# will produce configuration
```

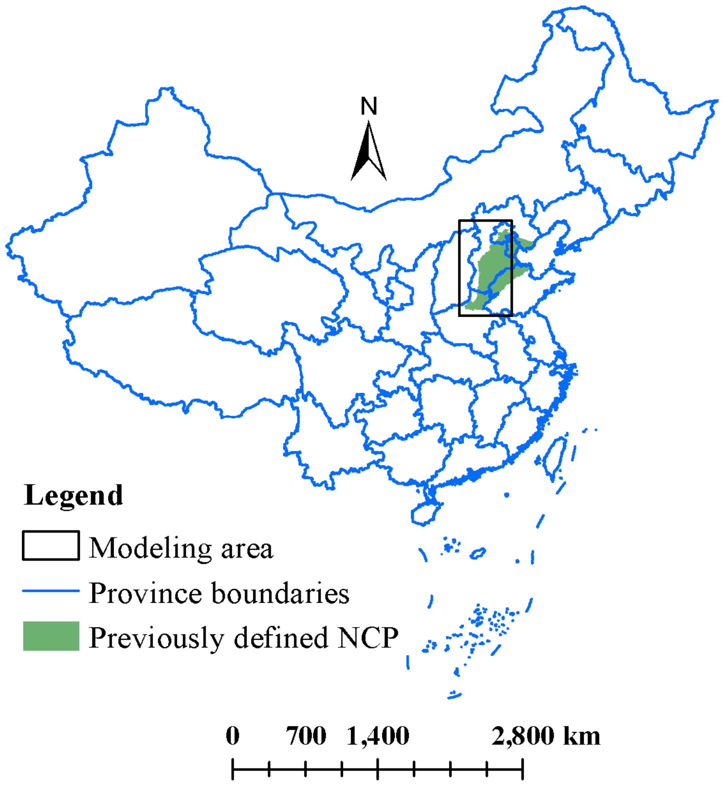


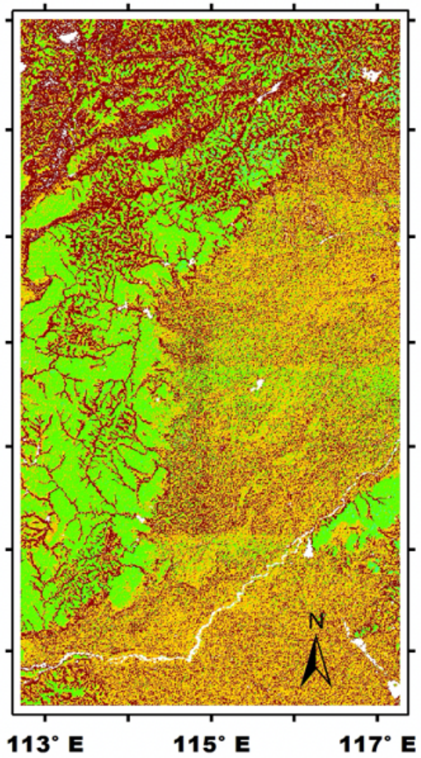
The test results using an array of size 80,000,000*10 for serial and parallel codes:

a. the parallel code used 12.473 ms, the serial code used 1405.444 ms, so the speedup is **112-fold**
b. considering transfer data, speedup is **159-fold**



N(*10)	GPU-parallel	CPU-serial	h2d	d2h	speedup	speedup(data)
80	0.147	0.004	0.024	0.021	0.027	0.333
800	0.154	0.010	0.027	0.024	0.065	0.396
8000	0.150	0.081	0.063	0.060	0.540	1.360
80000	0.624	0.788	0.421	0.416	1.263	2.604
800000	1.035	10.097	4.094	4.012	9.756	17.587
8000000	2.580	139.544	39.435	39.020	54.087	84.496
80000000	12.473	1405.444	338.730	244.039	112.679	159.401





Long term simulation of 40 years with hourly time step
1s*8760*40/3600/24 = 4 days wall-clock time saved

Problems and Solutions

- What problems are you currently facing?
- Have you resolved any problems (or found bugs) that others might find useful?