Princeton GPU Hackathon 2021

EcoSLIM

Jun Zhang
U of Arizona

Hoang Tran
Princeton U

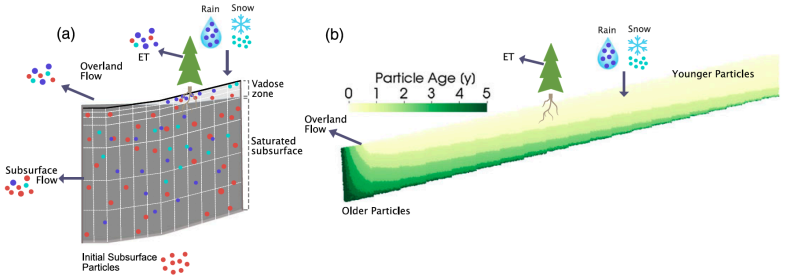
Chen Yang
Princeton U

Troy Comi
Princeton U

Carl Ponder
NVIDIA

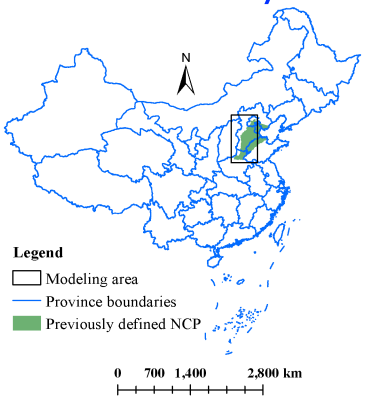
EcoSLIM

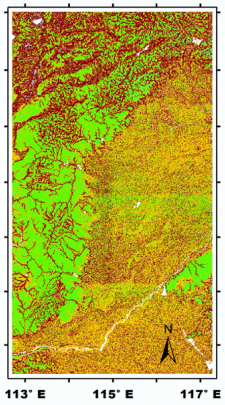
A Lagrangian particle tracking code calculating water ages and source water mixing



(a) Schematic of particle tracking method and (b) example of particle age distribution during a simulation

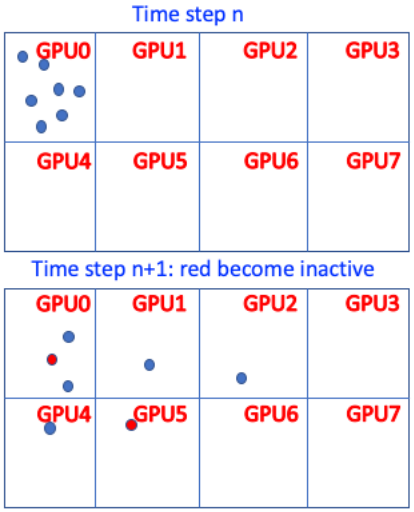
- What was your goal coming here?

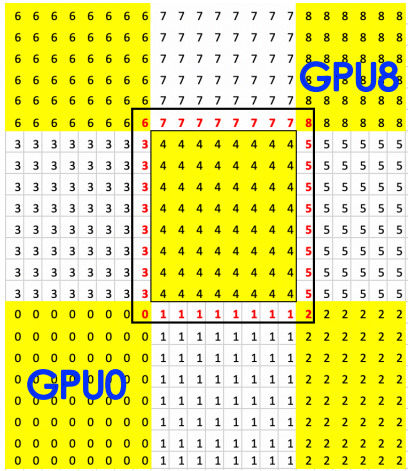




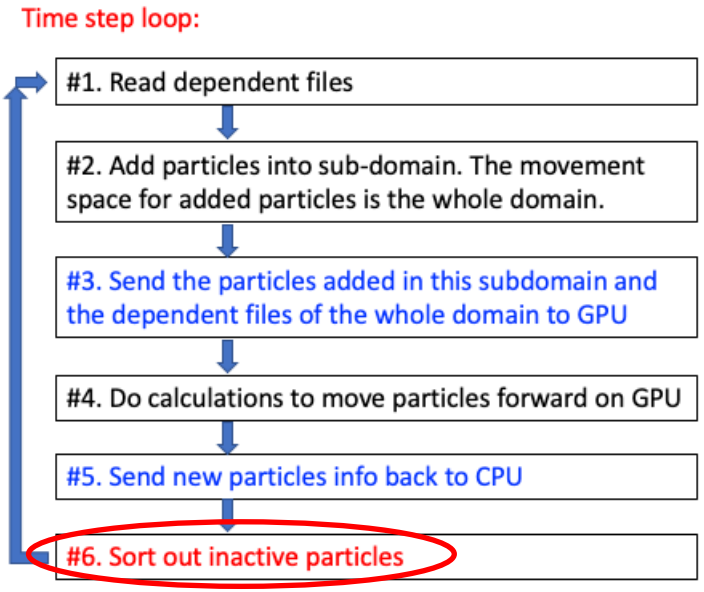
- Initial strategy? How did this strategy change?

Multi-GPU with MPI
Fortran/CUDA-Fortran





What we focus on?



Stream compaction based on prefix sum

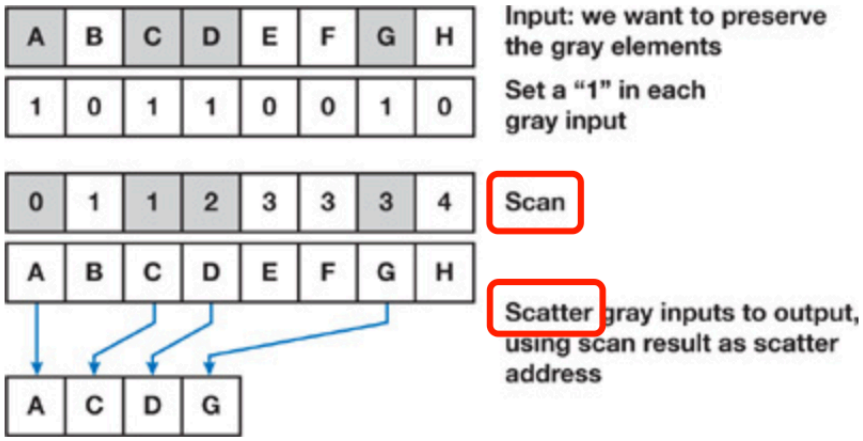
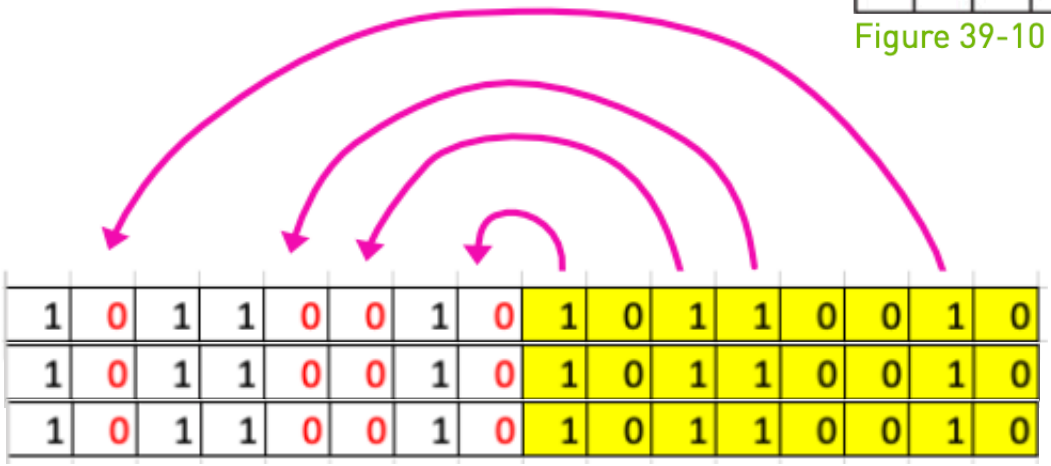


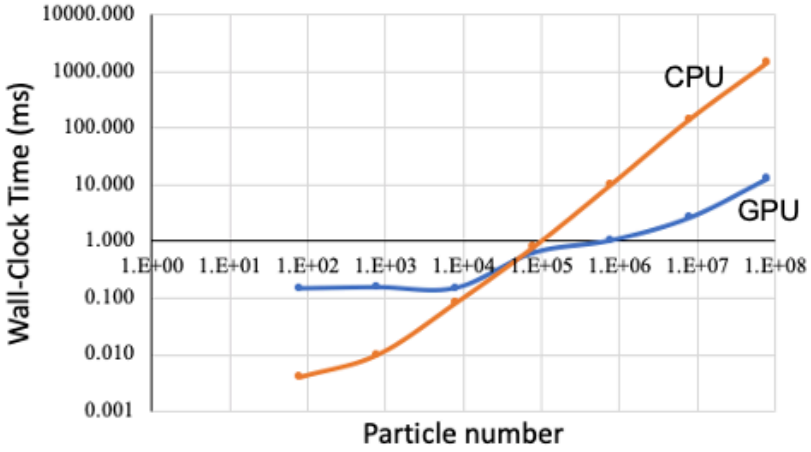
Figure 39-10 Scan and Scatter



New in-place compaction

1. 1D array to 2D array
2. In-place compaction

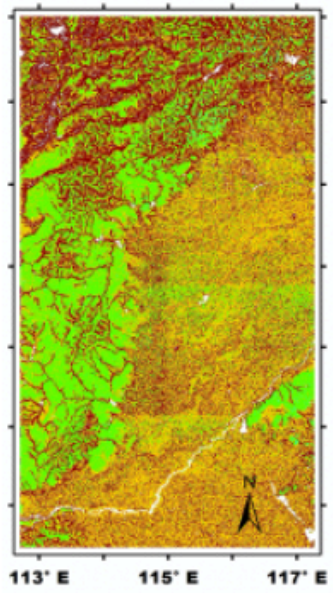
Results



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

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

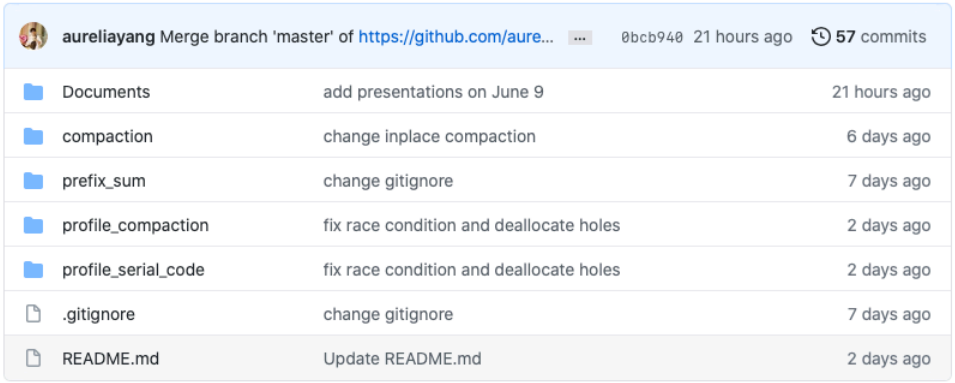
a. the parallel code used 12.473 ms, the serial code used 1405.444 ms, so the speedup is **112-fold**
b. considering transfer data, speedup is **159-fold**



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

Outcome on Github

https://github.com/aureliayang/EcoSLIM_Compaction



aureliayang Merge branch 'master' of https://github.com/aure... ... `0bcb940` 21 hours ago **57** commits

Documents	add presentations on June 9	21 hours ago
compaction	change inplace compaction	6 days ago
prefix_sum	change gitignore	7 days ago
profile_compaction	fix race condition and deallocate holes	2 days ago
profile_serial_code	fix race condition and deallocate holes	2 days ago
.gitignore	change gitignore	7 days ago
README.md	Update README.md	2 days ago

README.md

Improved Stream Compaction

1. **compaction/ecoslim_compaction.cuf** is a module that contians two device subroutines. Both are for 2D particle array. The first one needs a temporary array while the second one is in-place compaction (with race condition). It has passed test on Della-GPU at Princeton through a small array of size 8*4.
2. **prefix_sum/thrust_scan:** test for scan using a column in 2D array. It uses a wrapper to call scan in Thrust.
3. **prefix_sum/thrust_scan_MPI:** test for scan using MPI (tested on Tiger-GPU).
4. **Documents/:** presentation slides in the GPU Hackathon
5. **profile_serial_code/:** the origional serial code
6. **profile_compaction/:** the parallel in-place compaction code

The test/profiling results (slurm-xxx.out in each folder) using an array of size 80,000,000*10 for serial and parallel codes:
Environment (Princeton Della-GPU cluster): AMD EPYC 7H12 64-Core Processor, NVIDIA A100 GPU, nvhpc/21.1, cudatoolkit/11.1
a. the parallel code used 12.473 ms, the serial code used 1405.444 ms, so the speedup is **112-fold** (1405.444/12.473)
b. considering that it is not necessary to transfer data if using parallel code on GPU, it is **159-fold** ((1405.444+338.730+244.039)/12.473)

Development of stream compaction for EcoSLIM in 2021 GPU Hackathon at Princeton

Readme

Releases

No releases published
Create a new release

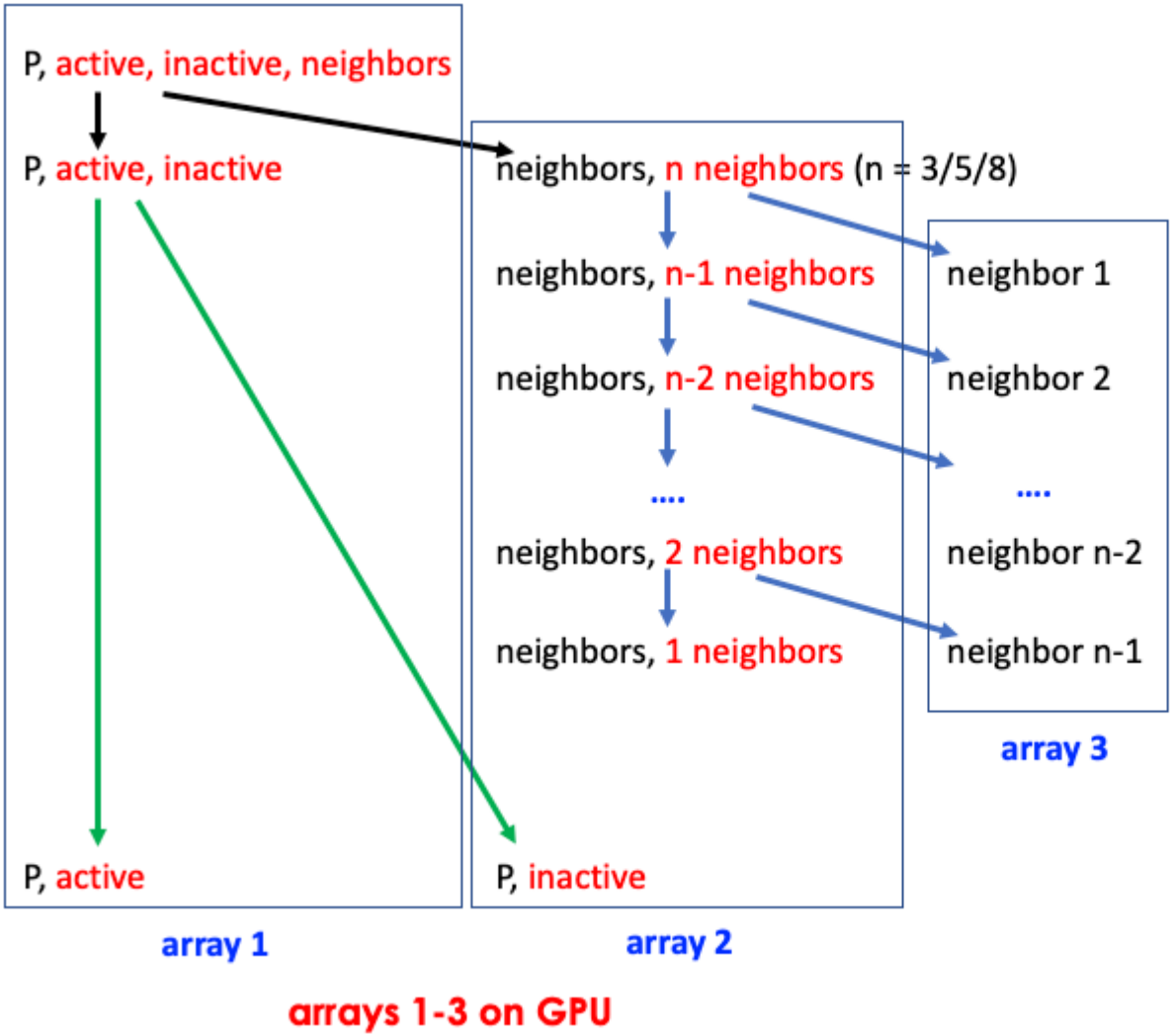
Packages

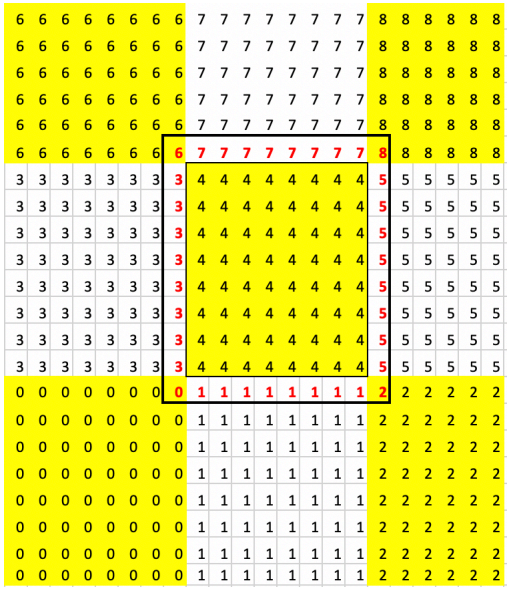
No packages published
Publish your first package

Languages

Cuda 65.1% **Makefile** 19.5%
Shell 15.4%

Applications of compaction





call scan
call compaction

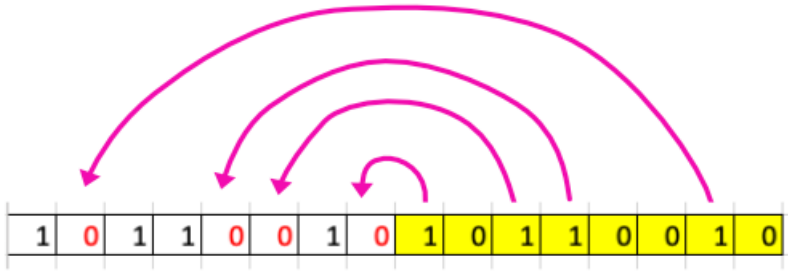
do n = 1, (n-1)neighbors
call scan
call compaction
end do

call scan
call compaction

What problems you encountered

- Race condition

1. **build holes**
2. **fill holes**



Holes: the locations of zeros

```
ii = (blockIdx%x - 1) * blockDim%x + threadIdx%x
if(ii <= numElements .and. idnint(d_isValid(ii)) == 0) then
    holes(ii - idnint(d_indices(ii))) = ii
end if

!ii  = numElements - ii + 1
ii2 = idnint(d_indices(numElements)) - idnint(d_indices(ii)) + 1

if(ii <= numElements .and. idnint(d_isValid(ii)) > 0 .and. &
   holes(ii2) <= idnint(d_indices(numElements))) then
    do j = 1, n_attri
       d_in(holes(ii2),j) = d_in(ii,j)
    enddo
end if
```

```
!----------------
istat = cudaEventRecord(startEvent, 0)

call thrustscan(gpuData(:,11),size(gpuData(:,11)),gpuData(:,12))
temp = gpuData(N,12); allocate(holes(N-idnint(temp))) !This time i

call prepare_holes<<<ceiling(dble(N)/tPB),tPB>>>(&
holes,gpuData(:,12),gpuData(:,11),N)

call compaction_inplace<<<ceiling(dble(N)/tPB),tPB>>>(&
holes,gpuData(:,12),gpuData(:,11),gpuData,N,M)
!in place
deallocate(holes)

istat = cudaEventRecord(stopEvent, 0)
istat = cudaEventSynchronize(stopEvent)
istat = cudaEventElapsedTime(time2, startEvent, stopEvent)
```

What did you learn?

NVIDA Nsight: profiling our stream compaction results

Time(%)	Total Time (ns)	Num Calls	Average	Minimum	Maximum	Name
64.2	2,428,692,058	300	8,095,640.2	5,039	50,054,166	cudaMemcpy
28.3	1,071,630,045	2	535,815,022.5	2,615	1,071,627,430	cudaEventCreate
5.0	190,215,074	1	190,215,074.0	190,215,074	190,215,074	cudaMallocHost
1.2	47,295,089	270	175,167.0	2,545	2,127,825	cudaFree
0.9	34,055,413	272	125,203.7	3,095	3,380,200	cudaMalloc

Time(%)	Total Time (ns)	Operations	Average	Minimum	Maximum	Operation
50.2	1,208,948,011	240	5,037,283.4	2,368	40,472,025	[CUDA memcpy HtoD]
49.8	1,197,145,410	60	19,952,423.5	4,543	40,216,057	[CUDA memcpy DtoH]

8×10^6

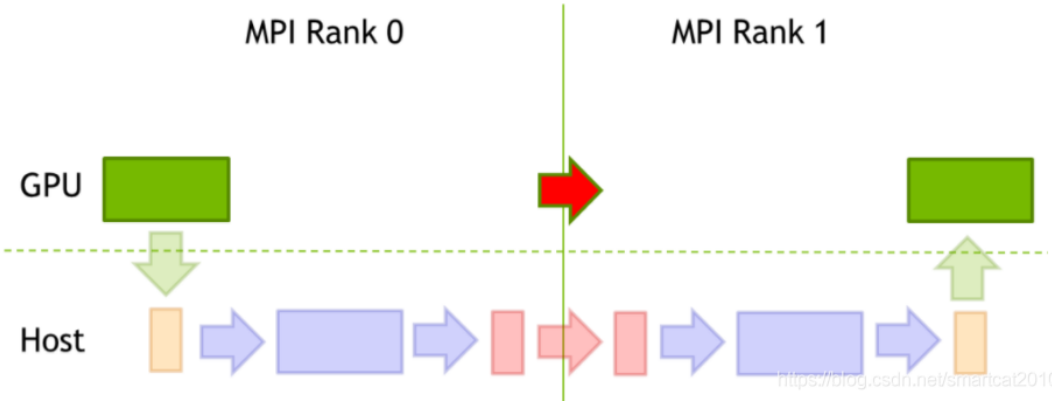
8×10^7

Time(%)	Total Time (ns)	Num Calls	Average	Minimum	Maximum	Name
86.6	17,649,944,930	300	58,833,149.8	4,569	365,509,714	cudaMemcpy
9.8	1,987,035,036	1	1,987,035,036.0	1,987,035,036	1,987,035,036	cudaMallocHost
1.7	346,084,633	2	173,042,316.5	2,825	346,081,808	cudaEventCreate
1.4	292,094,288	270	1,081,830.7	2,855	8,164,309	cudaFree

Time(%)	Total Time (ns)	Operations	Average	Minimum	Maximum	Operation
58.5	10,319,055,599	240	42,996,065.0	1,472	365,470,556	[CUDA memcpy HtoD]
41.5	7,323,201,328	60	122,053,355.5	2,752	245,834,493	[CUDA memcpy DtoH]

What did you learn?

CUDA-Aware MPI




```
! Initialize host and device buffers
allocate(h_buff(size))
allocate(d_buff(size))
! Implicity copy rank to device
d_rank = rank

! Preform allgather using device buffers
call MPI_ALLGATHER(d_rank, 1, MPI_INTEGER, d_buff, 1, &
MPI_INTEGER, MPI_COMM_WORLD, ierror);

! Check that buffer is correct
h_buff = d_buff(1:size)
```

What did you learn?

Generate random numbers using Device API of cuRAND

```
1  module mrand
2      use curand_device
3      integer, parameter :: n = 500
4      type(curandStateXORWOW),device,allocatable,dimension(:,:) :: h
5      contains
6          attributes(global) subroutine createRand()
7             integer(8) :: seed,seq,offset,i,j
8             !type(curandStateXORWOW),dimension(:,:) :: h
9             j = blockIdx%x; i = threadIdx%x
10            seed = 12345_8 + j*n*n + i*2
11            seq = 0_8
12            offset = 0_8
13            call curand_init(seed, seq, offset, h(i,j))
14         end subroutine createRa
15         attributes(global) subroutine randsub(a)
16         !type(curandStateXORWOW),dimension(:,:) :: h
17         real :: a(n,n,4)
18         integer(8) :: i,j,k
19         j = blockIdx%x; i = threadIdx%x
20         do k = 1, 4
21             a(i,j,k) = curand_uniform(h(i,j))
22         end do
23         end subroutine
24 end module
```

```
% nvfortran rand.cuf; a.out
   0.9999960        1.0117656E-06   0.4987832
   0.9999998        3.0419324E-07   0.5002192
   0.9999997        1.4506513E-06   0.5000274
```

What did you learn?

Refactor code with cmake

junzhang — jz1248@della-gpu:~/EcoSLIM/testing/accepted — ssh

1000	6.6480600000140555E-004	0.000000000000000
996	5.2757899999988922E-004	3.4099999979275708E-007
996	5.3334999999954391E-004	4.6200000092255777E-007
996	5.2854000000124302E-004	5.8200000196961810E-007
1026	5.4430000000138534E-004	7.2200000289512900E-007
	004	8.4200000216583248E-007
	004	8.9200000097378052E-007

Previous accepted results

junzhang — jz1248@della-gpu:~/EcoSLIM/testing/accepted/myaccepted

1000	9.3942099999999584E-004	0.000000000000000
996	7.6589599999898894E-004	3.8099999954965824E-007
996	7.7357000000155551E-004	4.9099999976931485E-007
996	7.6108700000077079E-004	6.3199999900120929E-007
1026	7.7404999999863833E-004	7.6199999909931648E-007
	004	8.8199999837001997E-007
	004	1.0219999975191740E-006

Results from cmake test

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
```

CMake

entr(1)

Run arbitrary commands when files change

tmux

Wishlist and next step

- **What do you wish existed to make your life easier?**
  - Tools
  - Language standards
  - Event
  - Systems
- **Go on the development of EcoSLIM to handle the continental US scale**

Acknowledgements

- Thanks to Julia and Gabe, and other organizers!
- Thanks to every awesome attendee!
- Thanks to Carl and Troy!
- Thanks to our members!