LLVM Clang + OpenMP GPU Offloading Testbed: FastCaloSim

HEP-CCE All Hands Meeting
11-13 Oct '22

FastCaloSim: OpenMP GPU Offloading

- Fast simulation of ATLAS calorimeter system
- Originally C++, ported to CUDA, Kokkos, HIP, SYCL

- For the OpenMP port:
  - Random Number Generation
    - Seed on GPU (cuRAND/ rocRAND)
    - Seed on CPU, copy to GPU
  - Load Geometry

Memory: alloc, copy, free

```
cudaMalloc (**devicePointer, size)

devicePointer= omp_target_alloc(size, deviceID)
```

```
cudaMemcpy (dest, src, count, cudaMemcpyHostToDevice)

omp_target_memcpy (dest, src, count,
                   dest_offset, src_offset,
                   dst_dev_id, src_dev_id)
```

```
cudaFree (devicePointer)

omp_target_free(devicePointer, deviceID)
```

  - Simulate Hits
    - 3 kernels (clean, hits, count)
    - Parallelizable *for* loops, thread local flops
    - Atomic operations

```
#pragma omp target is_device_ptr ( devicePointer ) map ( )
#pragma omp teams distribute parallel for
for (  ;   ;  ) {
   ...
   #pragma omp atomic
   ...
}
```

FastCaloSim (RNs on GPU)

GPU NVIDIA A6000
Clang 15.0.0, 10k events

```
memory allocation ();
copy H to D // random numbers, geometry

for ( ; ; ) { //events
   ...
   for ( ; ; ) { //particles
      ...
      Args args; // set arguments
      ...
      simulate_clean ( args ); // for ncells ~187k
      simulate_A ( args );     // for nhits ~ 5-6k
      simulate_ct ( args );    // for ncells
   }
   copy D to H //cells energy, hit counts
}
```

Kernel	OpenMP	CUDA
simulate_clean	0.133 s	0.053 s
simulate_A	0.361 s	0.359 s
simulate_ct	0.144 s	0.058 s

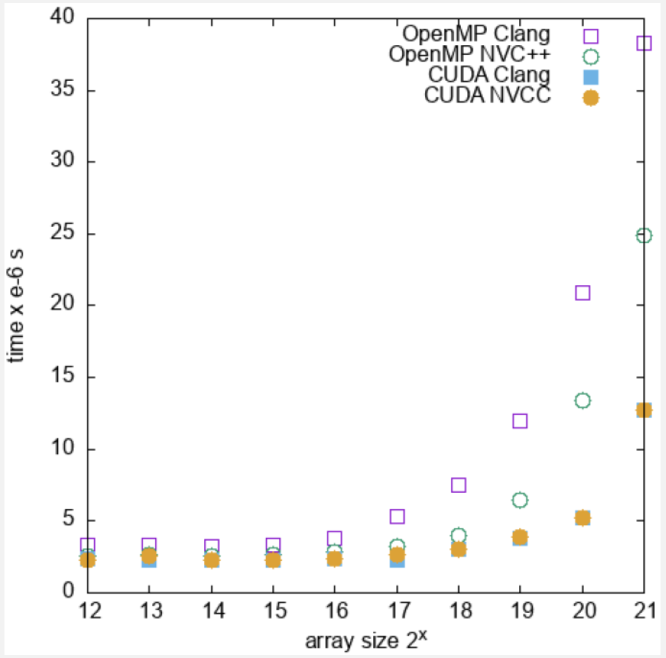
API	OpenMP	CUDA
copy H to D	0.021 s	0.002 s
copy D to H	0.044 s	0.044 s

Kernel Performance

Device Array Initialization (simulate_clean)

```
real *device_array = (real *) omp_target_alloc( N * sizeof( real ),
m_default_device);

#pragma omp target is_device_ptr ( device_array )
#pragma omp teams distribute parallel for num_teams(nblocks)
num_threads(blocksize)
 for(i = 0; i < N; i++) {
   device_array[i] = 0.;
 }
```
*Communicated to LLVM developers



Clang 15, array 2^18

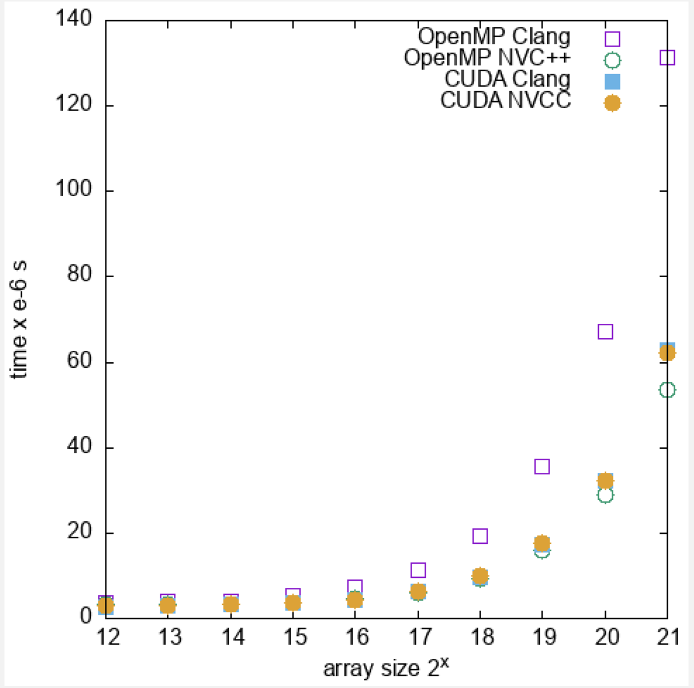
```
 Time(%)  Total Time (ns)     Name

CUDA Kernel Statistics:
 -------  --------------  ---------
 100.0      3,328      set_to_zero_cu
 -------  --------------  ---------
OpenMP Kernel Statistics:
 -------  --------------  ---------
 100.0      7,584      omp_offloading_setzero
```

Here, 2.2x slower, FCS 2.5x slower

Atomic capture (simulate_ct)

```
#pragma omp target is_device_ptr ( devc_count, devc_array,
devc_array_pos ) device(m_default_device)
 #pragma omp teams distribute parallel for num_teams(256)
 for ( int i = 0; i < N; i++ ) {
    if ( devc_array[i] > 0. ) {
      #pragma omp atomic capture
      temp = devc_count[0]++;
      devc_array_pos[temp] = devc_array[i];
    }
 }
```
*To be communicated to LLVM developers



```
CUDA Kernel Statistics:
 -------  --------------  ---------
100.0      9,824      collect_pos
 -------  --------------  ---------
OpenMP Kernel Statistics:
 -------  --------------  ---------
100.0      19,361      omp_offloading_collect
```

Here, 2x slower, FCS 2.5x slower

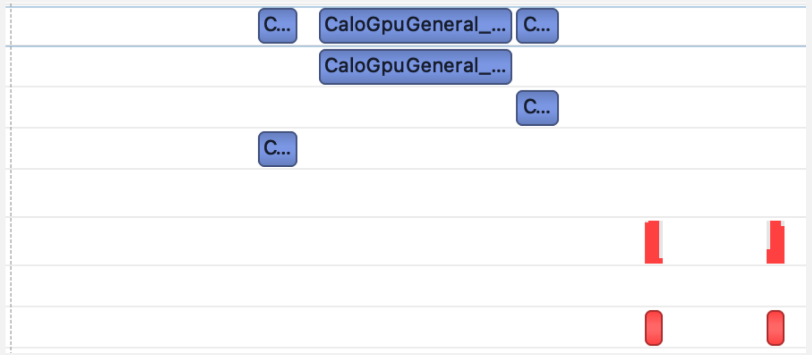
Data Movement

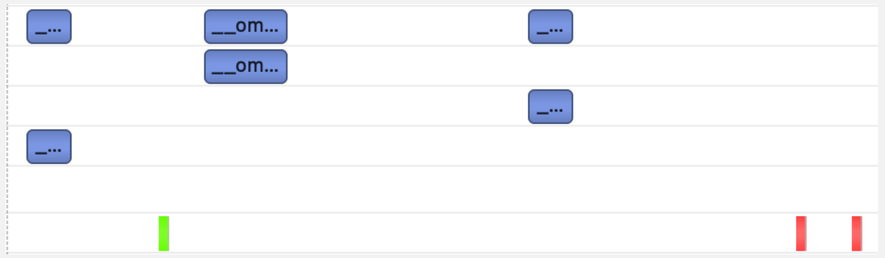
CUDA Statistics

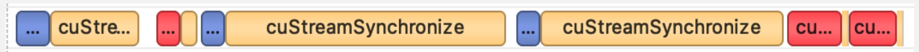
```
Time(%)  Time (s)  Count  Total (MB)    Operation
-------  -------------- ----- ------  ----------  ----------  ---------
  95.3     0.044   39,950   20.785    [memcpy DtoH]
   4.7     0.002     475    23.153   [memcpy HtoD]
```

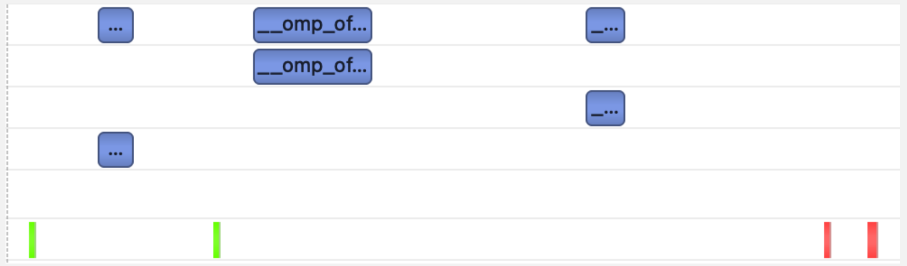
OpenMP Statistics

```
Time(%)  Time (s)  Count  Total (MB)    Operation
-------  -------------- ------ ------ ---------- ----------   -----------
  67.4     0.044     39,954  20.785    [memcpy DtoH]
  32.6     0.021     20,451  29.864    [memcpy HtoD]
```









- 6.71 MB of args mapped implicitly
- Once for each simulate_A call
- Is Nsight Systems missing it for CUDA?

What did we learn?

- Important to match block and grid to CUDA, default values not good enough
- Use declare mapper for structs: individual members copied one at a time
- Nsight Systems creates a larger overhead for profiling OpenMP target offloads
- Nsight Systems does not count function arguments as data copy

Challenges

- Total run time
  - CUDA: 13.51 s
  - OpenMP: 14.95 s
- cuStreamSync ? (illegal mem access with **nowait** clause)
- Matching performance metrics with CUDA

- Future work

- Compilers: nvc++ / amdclang++ / g++ / icpc
- Test on other Nvidia and AMD/Intel GPU

Thank you!