L18: Global Synchronization and Sorting

Administrative

• Grading
  - Should have exams. Nice job!
  - Design review written feedback later today

• TODAY: Cross-cutting systems seminar,
  Monday, April 20, 12:15-1:30PM, LCR: “Technology Drivers for Multiprocessor Architectures,” Rajeev Balaabramoniam, Mary Hall, Ganesh Gopalakrishnan, John Regehr

• Deadline Extended to May 4: Symposium on Application Accelerators in High Performance Computing
  http://www.saahpc.org/

• Final Reports on projects
  - Poster session April 29 with dry run April 27
  - Also, submit written document and software by May 6
  - Invite your friends! I’ll invite faculty, NVIDIA, graduate students, application owners, ...

Final Project Presentation

• Dry run on April 27
  - Easels, tape and poster board provided
  - Tape a set of Powerpoint slides to a standard 2’x3’ poster, or bring your own poster.

• Final Report on Projects due May 6
  - Submit code
  - And written document, roughly 10 pages, based on earlier submission.
  - In addition to original proposal, include
    - Project Plan and How Decomposed (from DR)
    - Description of CUDA implementation
    - Performance Measurement
    - Related Work (from DR)

Sources for Today’s Lecture

• Global barrier, Vasily Volkov code
• Suggested locking code
• Bitonic sort (CUDA zone)
  http://www.ce.chalmers.se/~uffe/hybridsortElsevier.pdf
Global Barrier - What does it do?

```c
__global__ void barrier( volatile int *slave2master, volatile int *master2slave, int niterations ) {
    for( int id = 1; id <= niterations; id++ ) {
        if( blockIdx.x == 0 ) {
            // master thread block: wait until all slaves signal, then signal
            if( threadIdx.x != 0 )
                while( slave2master[threadIdx.x] != id );
            __syncthreads( );
            master2slave[threadIdx.x] = id;
        } else {
            // slave thread block: signal to the master, wait for reply
            if( threadIdx.x == 0 )
                slave2master[blockIdx.x] = id;
            while( master2slave[blockIdx.x] != id );
            __syncthreads();
        }
    }
}
```

Questions

- Safe? Why?
- Multiple iterations?

Simple Lock Using Atomic Updates

Can you use atomic updates to create a lock variable?

Consider primitives:

```c
int lockVar;
atomicAdd(&lockVar, 1);
atomicAdd(&lockVar, -1);
```
Features of Implementation

• All data fits in shared memory (sorted from there)
• Time complexity: $O(n\log n^3)$

New Sorting Algorithm (Sintorn and Assarsson)

• Each pass:
  - Merge 2L sorted lists into L sorted lists
  - When L is less than the number of 2*SMs, switch to

• Three parts:
  - Histogramming: to split input list into L independent sublists for Pivot Points
  - Bucketsort: to split into lists than can be sorted using next step
  - Vector-Mergesort:
    - Elements are grouped into 4-float vectors and a kernel sorts each vector internally
    - Repeat until sublist is sorted

• Results:
  - 20% improvement over radix sort, best GPU algorithm
  - 6-14 times faster than quicksort on CPU