L12: Application Case Study I: Material Point Method

Administrative

- Proposal due next Thursday, March 8 at 5PM
- STRSM due Thursday, March 22 at 5PM
- Midterm coming
  - In class March 28, open notes
  - Will post prior exams
- Design Review
  - Intermediate assessment of progress on project, oral and short
  - Tentatively April 2 and 4
- Final projects
  - Poster session, Monday, April 23 (dry run April 18)
  - Final report, May 2

Project Proposal (due 3/8)

- Team of 2-3 people
  - Please let me know if you need a partner
- Proposal Logistics:
  - –Significant implementation, worth 50% of grade
  - –Each person turns in the proposal (should be same as other team members)
- Proposal:
  - –3-4 page document (11pt, single-spaced)
  - –Submit with handin program:
    "handin CS6235 prop <pdf-file>"

Project Parts (Total = 50%)

- Proposal (5%)
  - Short written document, next few slides
- Design Review (10%)
  - Oral, in-class presentation 3 weeks before end
- Presentation and Poster (15%)
  - Poster session last week of class, dry run week before
- Final Report (20%)
  - Due during finals – no final for this class

Content of Proposal

I. Team members: Name and a sentence on expertise for each member

II. Problem description
- What is the computation and why is it important?
- Abstraction of computation: equations, graphic or pseudo-code, no more than 1 page

III. Suitability for GPU acceleration
- Amdahl's Law: describe the inherent parallelism. Argue that it is close to 100% of computation. Use measurements from CPU execution of computation if possible.
- Synchronization and Communication: Discuss what data structures may need to be protected by synchronization, or communication through host.
- Copy Overhead: Discuss the data footprint and anticipated cost of copying to/from host memory.

IV. Intellectual Challenges
- Generally, what makes this computation worthy of a project?
- Point to any difficulties you anticipate at present in achieving high speedup

Projects from 2010

1. Green Coordinates for 3D Mesh Deformation
   Timothy George, Andrei Ostanin and Gene Peterson
2. Symmetric Singular Value Decomposition on GPUs using CUDA
   Gagandeep Singh and Vishay Vanjani
3. GPU Implementation of the Immersed Boundary Method
   Dan Maljovec and Varun Shankar
4. GPU Accelerated Particle System Representation for Triangulated Surface Meshes
   Manasi Datar and Brad Petersen
5. Coulombs Law on CUDA
   Torrey Atcitty and Joe Mayo
6. Bidomain Reaction-Diffusion Model
   Jason Briggs and Ayla Khan
7. Graph Coloring using CUDA
   Andre Vincent Pascal Grosset, Shusen Liu and Peihong Zhu
8. Parallelization API Performance Across Heterogeneous Hardware Platforms in Commercial Software Systems
   Toren Monson and Matt Stoker
9. EigenCFA: A CFA for the Lambda-Calculus on a GPU
   Tarun Prabhu and Shreyas Ramalingam
10. Anti-Chess
    Shayan Chandrashekar, Shreyas Subrmanya, Bharath Venkataramani

Projects from 2011

1. Counter Aliasing on CUDA, Dan Parker, Jordan Squire
2. Point Based Animation of Elastic Objects, Ashwin Kumar K and Ashok J
3. Data Fitting for Shape Analysis using CUDA, Qin Liu, Xiaoyue Huang
4. Model-Based Reconstruction of Undersampled DCE-MRI Tumor Data, Ben Felsted, Simon Williams, Cheng Ye
5. Component Streaming on Nvidia GPUs, Sujin Philip, Vince Schuster
6. Compensated Parallel Summation using Kahan's Algorithm, Devin Robison, Yang Gao
7. Implementation of Smoothness-Increasing Accuracy-Conserving Filters for DIscontinuous Galerkin Methods on the GPU, James King, Bharathan Rajaram, Supraja Jayakumar
8. Material Composites Optimization on GPU, Jonathan Bronson, Sheeraj Jadhav, Jihwan Kim
9. Grid-Based Fluid Simulation, Kyle Madsen, Ryan McAlister
10. Graph Drawing with CUDA to Solve the Placement Problem, Shomit Das, Anshul Joshi, Marty Lewis
11. Augmenting Operating Systems with the GPU: The Case of a GPU- augmented encrypted Filesystem, Weibin Sun, Xing Lin
12. Greater Than-Strong Conditional Oblivious Transfer Protocol using Graphics Processing Units, Prarthana Lakshmane Gowda, Nikhil Mishrikoti, Axel Rivera
13. Accelerating Dynamic Binary Translation with GPUs, Chung Hwan Kim, Srikanth Manikarnike, Vaibhav Sharma
14. Online Adaptive Code Generation and Tuning of CUDA Code, Suchit Maindola, Saurav Muralidharan
15. GPU-Accelerated Set-Based Analysis for Scheme, Youngrok Bahn, Seungkeol Choe
16. Containment Analysis on GPU, Anand Venkat, Preethi Kotari, Jacob Johns

Triangular Solve (STRSM)

```
for (j = 0; j < n; j++)
   for (k = 0; k < n; k++)
      if (B[j*n+k] != 0.0f) {
         for (i = k+1; i < n; i++)
            B[j*n+i] -= A[k * n + i] * B[j * n + k];
      }
```

Equivalent to:

```
cublasStrsm('l' /* left operator */, 'l' /* lower triangular */,
            'N' /* not transposed */, 'u' /* unit triangular */,
            N, N, alpha, d_A, N, d_B, N);
```

See: http://www.netlib.org/blas/strsm.f

Approaching Projects/STRSM/Case Studies

1. Parallelism?
   - How do dependences constrain partitioning strategies?
2. Analyze data accesses for different partitioning strategies
   - Start with global memory: coalesced?
   - Consider reuse: within a thread? Within a block? Across blocks?
3. Data Placement (adjust partitioning strategy?)
   - Registers, shared memory, constant memory, texture memory or just leave in global memory
4. Tuning
   - Unrolling, fine-tune partitioning, floating point, control flow, …

Step 1. Simple Partition for STRSM0

```
__global__ void strsm1( int n, float *A, float *B )
{
  int bx = blockIdx.x;
  int tx = threadIdx.x;
  int j = bx*THREADSPERBLOCK + tx; // 1 thread per column, columns work independently
  int JN = j * n;
  int i, k;

  for (k = 0; k < n; ++k) {    // ROW
    int KN = k * n;
    for (i = k+1; i < n; ++i) {  // ALSO row
      // B[i][j] -= A[i][k] * B[k][j] element depends on elts in ROWS above it in same col
      B[ JN + i ] -= A[ KN + i ] * B[ JN + k ];
    }
  }
}
```


Outline

- A few application case studies before break
- MRI and Molecular Visualization in Kirk and Hwu
- Material Point Method
  - Class project in 2009
  - Read "GPU Acceleration of the Generalized Interpolation Material Point Method," Wei-Fan Chiang, Michael DeLisi, Todd Hummel, Tyler Prete, Kevin Tew, Mary Hall, Phil Wallstedt, and James Guilkey, Symposium on Application Accelerators for High Performance Computing, July 2009.
  - Slides from SAAHPC 2009
  - Deadline for this year is April 26