ECE 498AL

Programming Massively Parallel Processors

Lecture 1: Introduction
Course Goals

• Learn how to program massively parallel processors and achieve
  – high performance
  – functionality and maintainability
  – scalability across future generations

• Acquire technical knowledge required to achieve the above goals
  – principles and patterns of parallel programming
  – processor architecture features and constraints
  – programming API, tools and techniques
People

• Professors:
  Wen-mei Hwu
  215 CSL, w-hwu@uiuc.edu, 244-8270
  use ECE498AL to start your e-mail subject line
  Office hours: 2-3:30pm Wednesdays; or after class
  David Kirk
  Chief Scientist, NVIDIA and Professor of ECE

• Teaching Assistant:
  ece498alTA@gmail.com
  John Stratton (stratton@uiuc.edu)
  Office hours: TBA
Web Resources

• Web site: http://courses.ece.uiuc.edu/ece498/al
  – Handouts and lecture slides/recordings
  – Textbook, documentation, software resources
  – Note: While we’ll make an effort to post announcements on the web, we can’t guarantee it, and won’t make any allowances for people who miss things in class.

• Web board
  – Channel for electronic announcements
  – Forum for Q&A - the TAs and Professors read the board, and your classmates often have answers

• Compass - grades
Grading
This is a lab oriented course!

• Exam5: 20%
• Labs: 30%
  – Demo/knowledge: 25%
  – Functionality: 40%
  – Report: 35%
• Project: 50%
  – Design Document: 25%
  – Project Presentation: 25%
  – Demo/Final Report: 50%
Bonus Days

• Each of you get five bonus days
  – A bonus day is a no-questions-asked one-day extension that can be used on most assignments
  – You can’t turn in multiple versions of a team assignment on different days; all of you must combine individual bonus days into one team bonus day.
  – You can use multiple bonus days on the same thing
  – Weekends/holidays don’t count for the number of days of extension (Friday-Monday is one day extension)

• Intended to cover illnesses, interview visits, just needing more time, etc.
Using Bonus Days

• Web page has a bonus day form. Print it out, sign, and attach to the thing you’re turning in.
  – Everyone who’s using a bonus day on an team assignment needs to sign the form

• Penalty for being late beyond bonus days is 10% of the possible points/day, again counting only weekdays (Spring/Fall break counts as weekdays)

• Things you can’t use bonus days on:
  – Final project design documents, final project presentations, final project demo, exam
Academic Honesty

• You are allowed and encouraged to discuss assignments with other students in the class. Getting verbal advice/help from people who’ve already taken the course is also fine.

• Any reference to assignments from previous terms or web postings is unacceptable

• Any copying of non-trivial code is unacceptable
  – Non-trivial = more than a line or so
  – Includes reading someone else’s code and then going off to write your own.
Academic Honesty (cont.)

• Giving/receiving help on an exam is unacceptable

• Penalties for academic dishonesty:
  – Zero on the assignment for the first occasion
  – Automatic failure of the course for repeat offenses
Team Projects

• Work can be divided up between team members in any way that works for you
• However, each team member will demo the final checkpoint of each MP individually, and will get a separate demo grade
  – This will include questions on the entire design
  – Rationale: if you don’t know enough about the whole design to answer questions on it, you aren’t involved enough in the MP
Lab Equipment

• Your own PCs running G80 emulators
  – Better debugging environment
  – Sufficient for first couple of weeks

• NVIDIA G80/G280 boards
  – QP/AC x86/GPU cluster accounts
  – Much much faster but less debugging support
UIUC/NCSA QP Cluster

- 16 nodes
  - 4-GPU (G80, 2 Quadro), 1-FPGA Opteron node at NCSA
  - GPUs donated by NVIDIA
  - FPGA donated by Xilinx

- Coulomb Summation:
  - 1.16 TFLOPS/node
  - 176x speedup vs. Intel QX6700 CPU core w/ SSE

- A large user community
  - QP has completed ~27,000 jobs and ~14,000 job hours since it began operation in May 2008
  - Urbana semester course, summer school
  - Many research accounts, many new requests

http://www.ncsa.uiuc.edu/Projects/GPUcluster/

A partnership between NCSA and academic departments.
UIUC/NCSA AC Cluster

- 32 nodes
  - 4-GPU (GTX280, Tesla),
  - 1-FPGA Opteron node at NCSA
  - GPUs donated by NVIDIA
  - FPGA donated by Xilinx

- Coulomb Summation:
  - 1.78 TFLOPS/node
  - 271x speedup vs. Intel QX6700 CPU core w/ SSE

UIUC/NCSA QP Cluster

http://www.ncsa.uiuc.edu/Projects/GPUcluster/

A partnership between NCSA and academic departments.
Text/Notes

1. Draft textbook by Prof. Hwu and Prof. Kirk available at the website
4. Lecture notes and recordings will be posted at the class web site
Tentative Schedule/Make-up Classes

• Regular make-up classes
  – Wed, 5:10-6:30 during selected weeks, location TBD

• Week 1:
  – Tue, 1/20 : Lecture 1: Introduction
  – Thu, 1/22: Lecture 2 – GPU Computing and CUDA Intro
  – MP-0, installation, run hello world

• Week 2:
  – Tue, 1/27: Lecture 3 – GPU Computing and CUDA Intro
  – Thu, 1/29: Lecture 4 – CUDA threading model
  – MP-1, simple matrix multiplication and simple vector reduction

• Week 3:
  – Tue, 2/3: Lecture 5 - CUDA memory model
  – Thu, 2/5: Lecture 6 – CUDA memory model, tiling
  – MP-2, tiled matrix multiplication

• Week 4
  – Tue, 2/10: Lecture 7 – CUDA computing history, Hardware
  – Thu, 2/12: Lecture 8 – CUDA performance
  – MP-3, simple and tiled 2D convolution
ECE498AL Development History

- 1/06: Kirk gives guest lecture at Hwu’s class. Blahut challenges Kirk and Hwu.
- 11/06: NVIDIA announces G80, Hwu/students training at NVIDIA.
- 1/07: Kirk visits UIUC, picking UIUC over others.
- 2/07: Kirk and Hwu in panic mode cooking up course material.
- 3/07: NVIDIA releases CUDA, UIUC lecture and lab material went online, G80 cards.
- 2/07: NAMD and other apps group report results and post projects.
- 6/08: UIUC becomes world’s first CUDA Center of Excellence.

UIUC becomes world’s first CUDA Center of Excellence.
Why Massively Parallel Processor

- A quiet revolution and potential build-up
  - Calculation: 367 GFLOPS vs. 32 GFLOPS
  - Memory Bandwidth: 86.4 GB/s vs. 8.4 GB/s
  - Until last year, programmed through graphics API

- GPU in every PC and workstation – massive volume and potential impact

© David Kirk/NVIDIA and Wen-mei W. Hwu, 2007-2009
ECE 498AL, University of Illinois, Urbana-Champaign
GeForce 8800

16 highly threaded SM’s, >128 FPU’s, 367 GFLOPS, 768 MB DRAM, 86.4 GB/S Mem BW, 4GB/S BW to CPU
G80 Characteristics

• 367 GFLOPS peak performance (25-50 times of current high-end microprocessors)
• 265 GFLOPS sustained for apps such as VMD
• Massively parallel, 128 cores, 90W
• Massively threaded, sustains 1000s of threads per app
• 30-100 times speedup over high-end microprocessors on scientific and media applications: medical imaging, molecular dynamics

“I think they're right on the money, but the huge performance differential (currently 3 GPUs ~≈ 300 SGI Altix Itanium2s) will invite close scrutiny so I have to be careful what I say publically until I triple check those numbers.”

- John Stone, VMD group, Physics UIUC
Future Apps Reflect a Concurrent World

• Exciting applications in future mass computing market have been traditionally considered “supercomputing applications”
  – Molecular dynamics simulation, Video and audio coding and manipulation, 3D imaging and visualization, Consumer game physics, and virtual reality products
  – These “Super-apps” represent and model physical, concurrent world

• Various granularities of parallelism exist, but…
  – programming model must not hinder parallel implementation
  – data delivery needs careful management
Stretching Traditional Architectures

- Traditional parallel architectures cover some super-applications
  - DSP, GPU, network apps, Scientific
- The game is to grow mainstream architectures “out” or domain-specific architectures “in”
  - CUDA is latter
## Previous Projects

<table>
<thead>
<tr>
<th>Application</th>
<th>Description</th>
<th>Source</th>
<th>Kernel</th>
<th>% time</th>
</tr>
</thead>
<tbody>
<tr>
<td>H.264</td>
<td>SPEC ‘06 version, change in guess vector</td>
<td>34,811</td>
<td>194</td>
<td>35%</td>
</tr>
<tr>
<td>LBM</td>
<td>SPEC ‘06 version, change to single precision and print fewer reports</td>
<td>1,481</td>
<td>285</td>
<td>&gt;99%</td>
</tr>
<tr>
<td>RC5-72</td>
<td>Distributed.net RC5-72 challenge client code</td>
<td>1,979</td>
<td>218</td>
<td>&gt;99%</td>
</tr>
<tr>
<td>FEM</td>
<td>Finite element modeling, simulation of 3D graded materials</td>
<td>1,874</td>
<td>146</td>
<td>99%</td>
</tr>
<tr>
<td>RPES</td>
<td>Rye Polynomial Equation Solver, quantum chem, 2-electron repulsion</td>
<td>1,104</td>
<td>281</td>
<td>99%</td>
</tr>
<tr>
<td>PNS</td>
<td>Petri Net simulation of a distributed system</td>
<td>322</td>
<td>160</td>
<td>&gt;99%</td>
</tr>
<tr>
<td>SAXPY</td>
<td>Single-precision implementation of saxpy, used in Linpack’s Gaussian elim. routine</td>
<td>952</td>
<td>31</td>
<td>&gt;99%</td>
</tr>
<tr>
<td>TRACF</td>
<td>Two Point Angular Correlation Function</td>
<td>536</td>
<td>98</td>
<td>96%</td>
</tr>
<tr>
<td>FDTD</td>
<td>Finite-Difference Time Domain analysis of 2D electromagnetic wave propagation</td>
<td>1,365</td>
<td>93</td>
<td>16%</td>
</tr>
<tr>
<td>MRI-Q</td>
<td>Computing a matrix Q, a scanner’s configuration in MRI reconstruction</td>
<td>490</td>
<td>33</td>
<td>&gt;99%</td>
</tr>
</tbody>
</table>
Speedup of Applications

- GeForce 8800 GTX vs. 2.2GHz Opteron 248
- 10× speedup in a kernel is typical, as long as the kernel can occupy enough parallel threads
- 25× to 400× speedup if the function’s data requirements and control flow suit the GPU and the application is optimized
- “Need for Speed” Seminar Series organized by Patel and Hwu this semester.