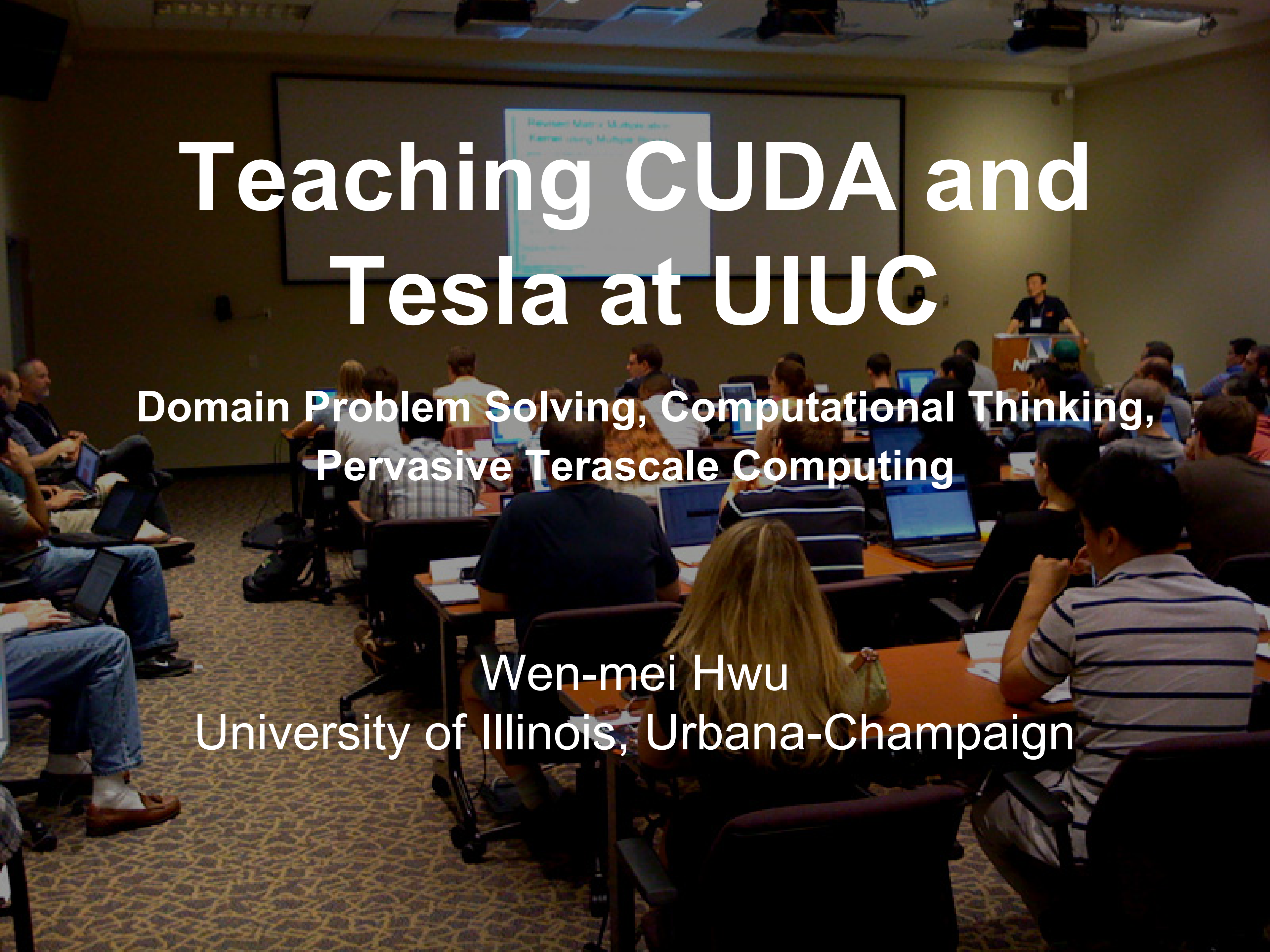


Teaching CUDA and Tesla at UIUC

Domain Problem Solving, Computational Thinking, Pervasive Terascale Computing

Wen-mei Hwu
University of Illinois, Urbana-Champaign

Our Objective

- To enable scientists and engineers to take advantage of pervasive, inexpensive, massively parallel computing devices to achieve breakthroughs in their disciplines.

We need to reach out to all scientists and engineers, not just computer scientists and engineers.

Course Versions Offered

- One-semester, graduate/senior version
  - twice, full registration, total 100+ students, hands-on lab
  - courses.ece.uiuc.edu/ece498/al, 100+ downloads per day
- Four day, graduate version
  - Summer school, full registration, 180 applicants, 44 accepted, 50+ remote participants, hands-on lab
  - www.greatlakesconsortium.org/events/GPUMulticore/agenda.html
- Three day, graduate version
  - Taiwan, full registration, 60+ students, hands-on lab
- Two day, graduate version
  - China, full registration, 50+ students, hands-on lab

Key Ingredients of a Parallel Programming Course

- An attractive parallel computing platform
- Balanced lectures and programming assignments
- Rewarding project experience.

Attractive Parallel Platform

- Wide availability
  - Students can use it for their own research work after the course.
  - With an average price as low as $100 and more than 70M units already in laptops, desktops, and servers, CUDA meets this criterion.
- Reasonable learning curve
  - An average student can write simple programs in a few hours after one lecture and a highly efficient program in one month
  - Experience confirms that CUDA meets this criterion.
- Rewarding experience
  - More than 10 times speedup as compared to a serial version
  - Results show that CUDA often greatly exceeds this criterion.

UIUC/NCSA QP Cluster

- 32 nodes
- 4-GPU (GT200)
- Coulomb Summation:
  - 1.78 TFLOPS/node
  - 271x speedup vs. Intel QX6700 CPU core w/ SSE

A

SE

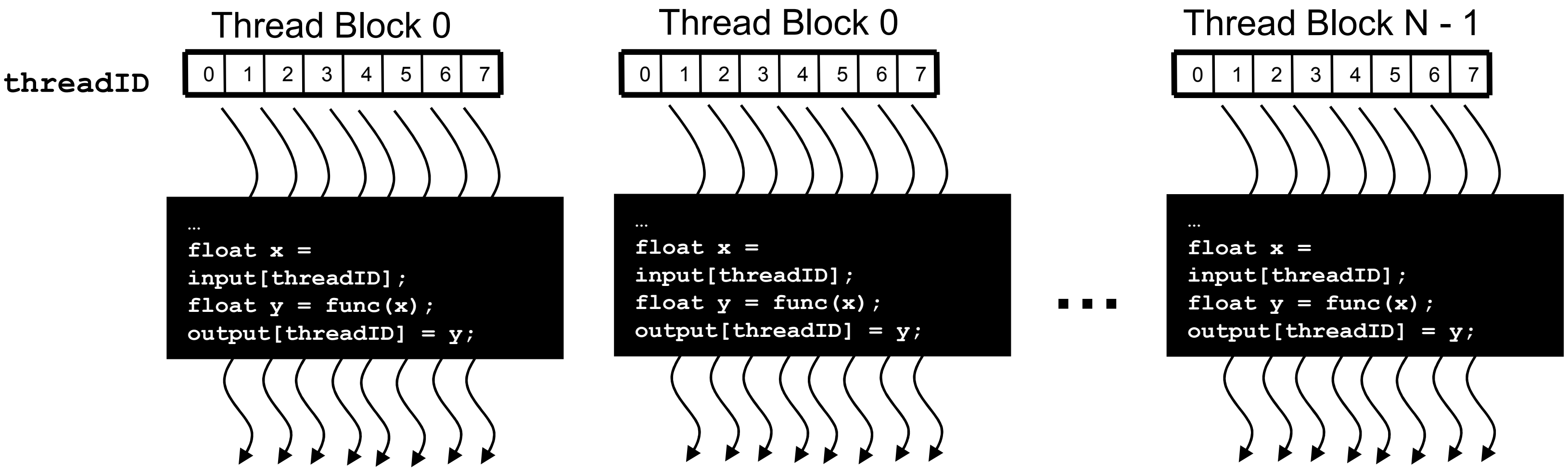
- Used extensively
  - Taiwan 3-day course
  - Urbana semester, summer
  - Many research accounts

UIUC/NCSA QP Cluster

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

Balanced Lectures and Programming Assignments

- One lecture teaches basic CUDA programming model
  - – CUDA host/device, CUDA threads, CUDA memories, CUDA extensions to the C language, and CUDA programming tools.
  - – Students write a parallel matrix multiplication code in three hours.



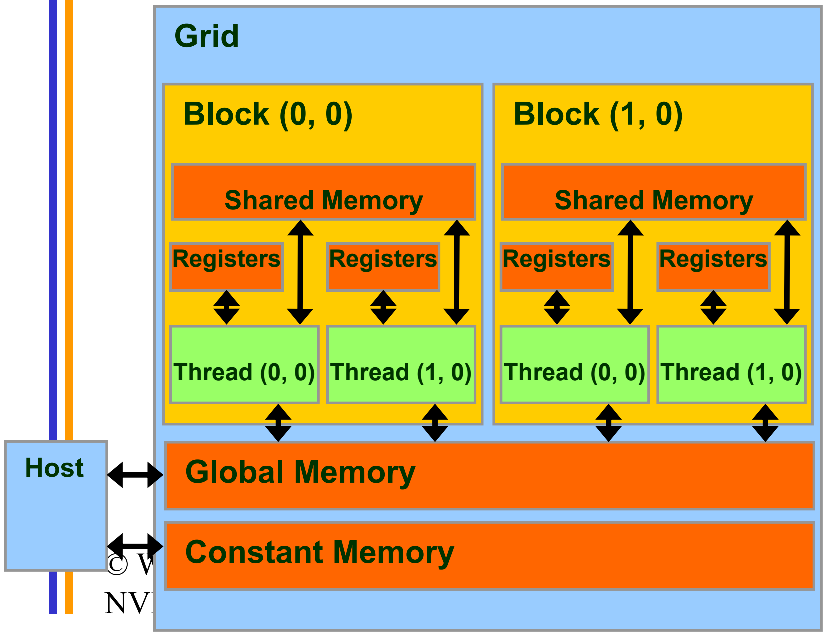
```
…
float x =
input[threadID];
float y = func(x);
output[threadID] = y;
```

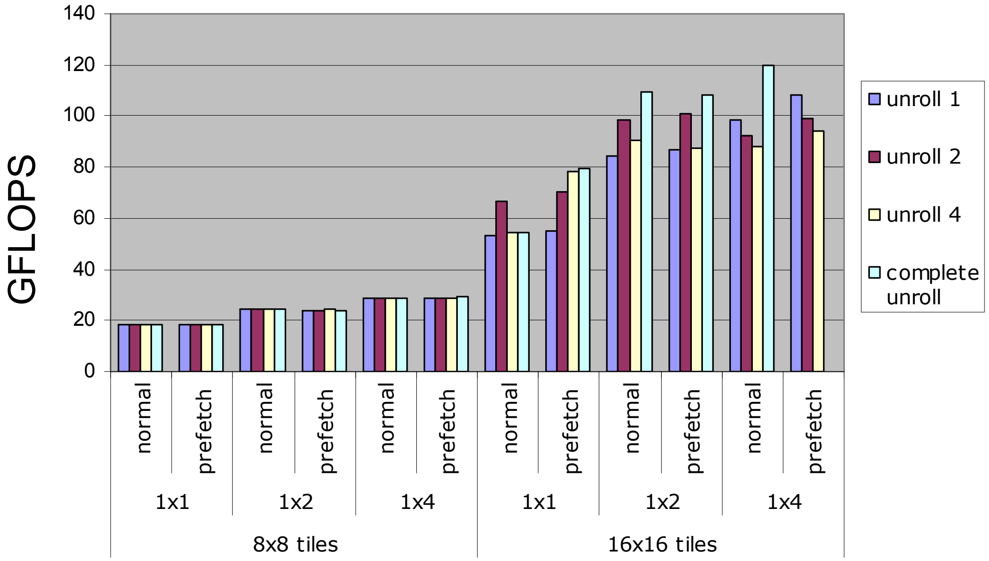
```
…
float x =
input[threadID];
float y = func(x);
output[threadID] = y;
```

```
…
float x =
input[threadID];
float y = func(x);
output[threadID] = y;
```

Balanced Lectures and Programming Assignments (cont.)

- 10 lectures on *conceptual* understanding of the CUDA memory model, the CUDA threading model, the GPU hardware performance features, and common data-parallel programming patterns.
  - Matrix multiplication codes goes from about 10 GFLOPS to about 120 GFLOPS through this period.
  - Programming assignments on convolution, vector reduction, and prefix scan through this period.

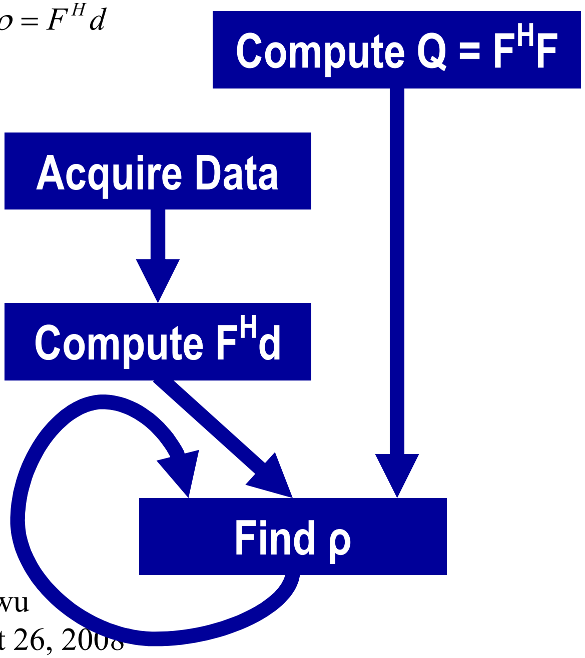




Balanced Lectures and Programming Assignments (cont.)

- Remaining lectures cover computational thinking, a broader range of parallel execution models, parallel programming principles, and case studies.

$F^H F\rho = F^H d$



Compute Q = $F^H F$

Acquire Data

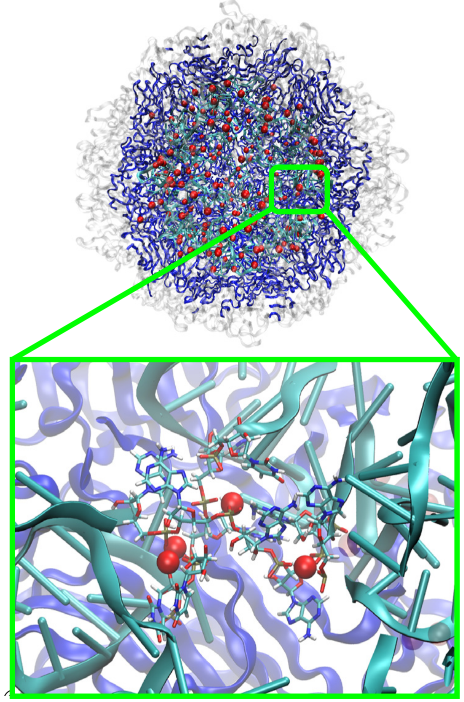
Compute $F^H d$

Find ρ

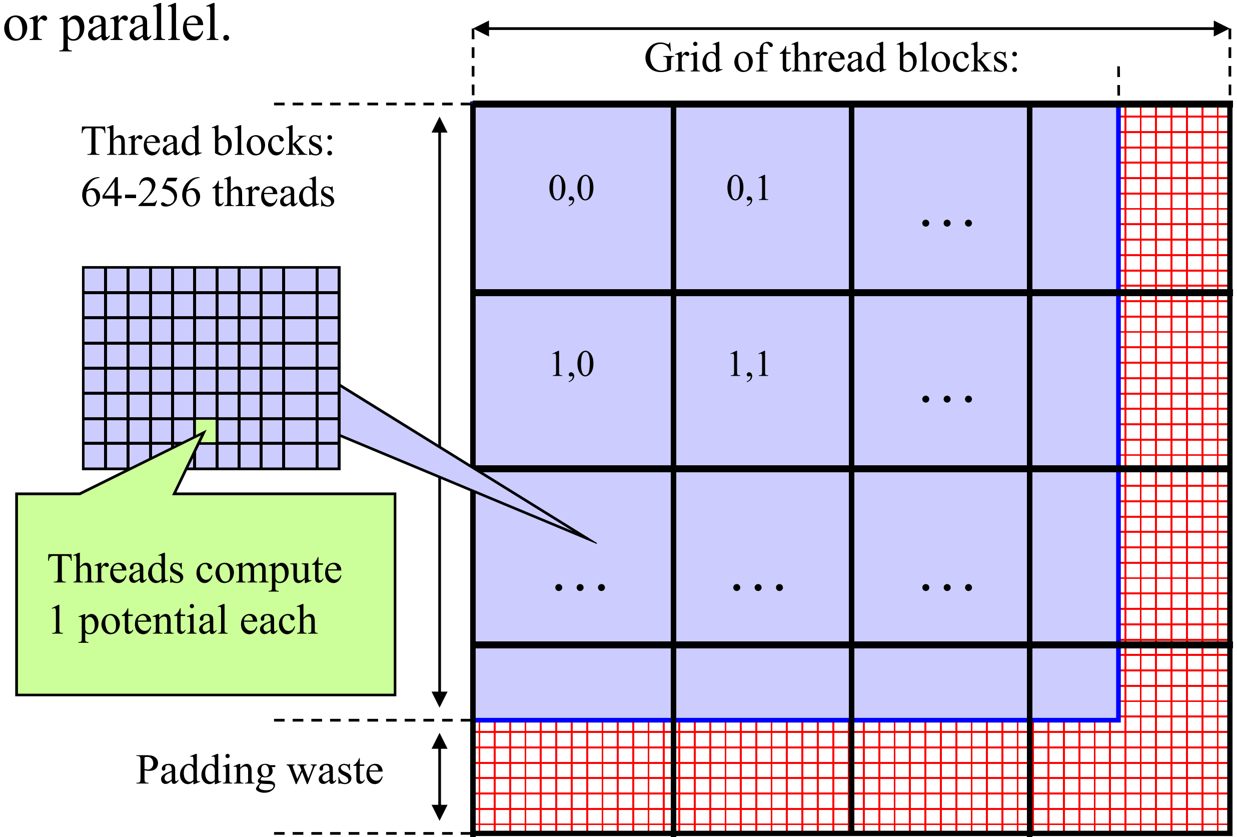
	OpenMP	MPI	CUDA
SPMD	☺ ☺ ☺	☺ ☺ ☺ ☺	☺ ☺ ☺ ☺ ☺
Loop Parallel	☺ ☺ ☺ ☺	☺	
Master/ Slave	☺ ☺	☺ ☺ ☺	
Fork/Join	☺ ☺ ☺		

Computational Thinking

- The most important skill in parallel programming
  - Organizing the computation tasks so that one can more easily identify high-level concurrent opportunities.
  - Mastering the translational process from scientific problems to computational tasks, important in producing quality application software, serial or parallel.



Satellite Tobacco Mosaic Virus (STMV)
Ion Placement



Grid of thread blocks:

Thread blocks:
64-256 threads

Threads compute
1 potential each

Padding waste

Rewarding Project Experience

- Several application development teams mentor projects.
  - 1-page project sheets recruit students to work on their applications.
  - Students can also propose their own applications and mentors.
- Six lecture slots are dedicated to project workshops
  - students present their project proposals to and receive feedback from the co-instructors, the TAs, the project mentors, and their fellow students.
- Six more lecture slots are dedicated to a class symposium, at which students present their final project results to an even wider audience.
  - In spring semester 2007, 52 students formed sixteen project teams of three to four students each, and all the students completed their projects.
  - The applications achieved up to 457x speedup over original application code in a high-end PC.
  - Several projects have evolved into thesis topics and student publications, and one even contributed to a successful NIH proposal.

Final Project Mentors

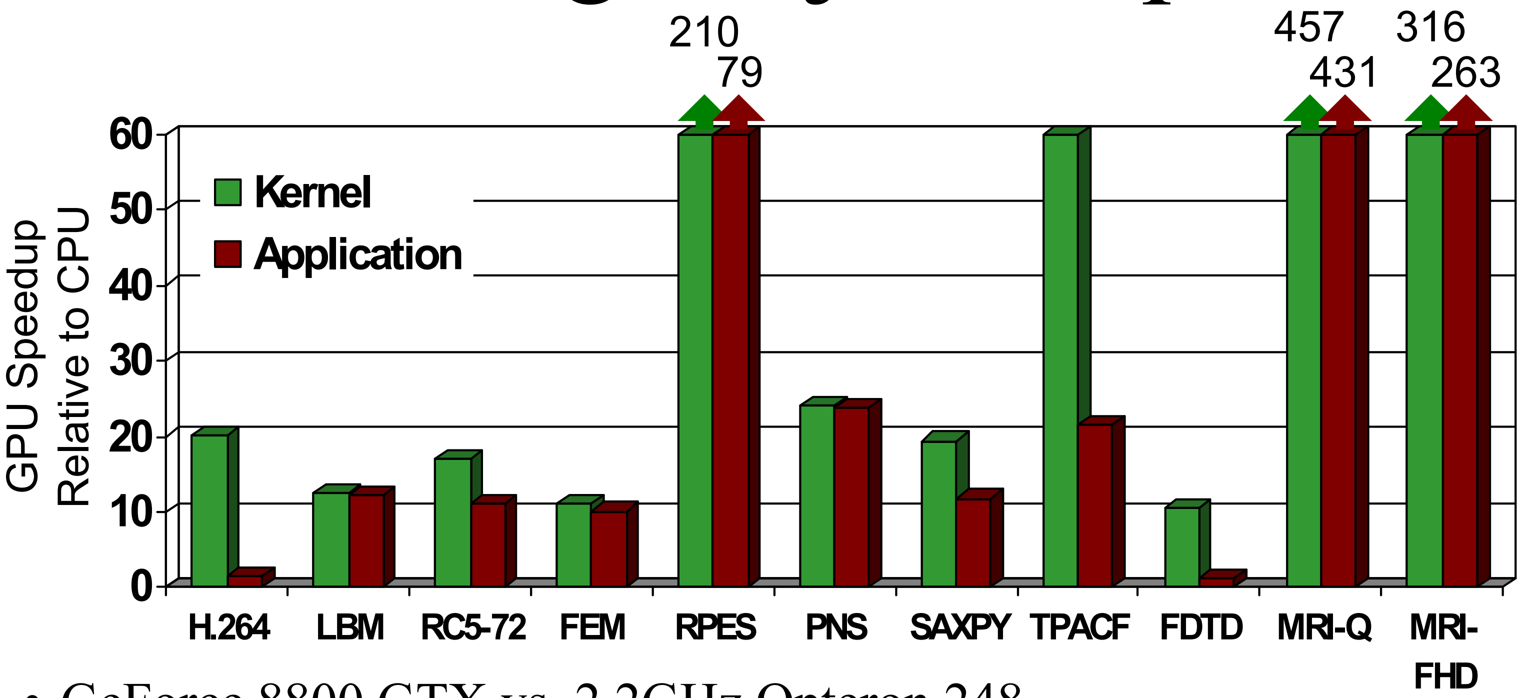
- Instructors recruit several major computational science research groups to serve as mentors.

- Mentors contribute a one-to-two-page project specification:
  - **Introduction:** significance of the application.
  - **Description:** what the application really accomplishes.
  - **Objective:** what the mentor wnat the student teams to accomplish
  - **Background:** Technical skills (types of Math, Physics, Chemistry courses) required to understand and work on the application.
  - **Resources:** web and traditional resources for technical background and building blocks, along with specific URLs or ftp paths.
  - **Contact Information:** Name and contact information for the person who will be mentoring the teams working on the application.

Project Workshop

- Students are not graded during the workshops, designed to stimulate meaning dialog with the instructor(s), TAs, and fellow students.
- Instructor(s) and TAs attend all the presentations and to give useful feedback:
  - Are the projects too big or too small for the amount of time available?
  - Is there existing work in the field that the project can benefit from?
  - Are the computations being targeted appropriate for the CUDA model?

Rewarding Project Experience



- 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 application’s data requirements, operation types and control flow suit the GPU
- Keep in mind that the speedup also reflects how suitable the CPU is for executing the kernel

Web Resources

- http://courses.ece.uiuc.edu/ece498AL
  - Handouts and Lecture Slides
  - Lecture voice recordings
  - Links to application groups supporting final projects
  - Hardware and software documentation
  - 100's of visits and audio streaming per day

Graduate Summer School Version

- Aimed at science and engineering graduate students
  - Full registration
  - 180 applicants from 3 continents
  - 44 accepted from 25 universities, 3 continents
  - 50+ remote participants
  - 9 lectures, hands-on lab, but no time for final project
  - Wen-mei Hwu and David Kirk Co-instructor
  - www.greatlakesconsortium.org/events/GPUMulticore

- Emphasize research applications
  - Multi-disciplinary panel, 3 visionary keynotes
  - Project idea sharing, evening discussion sessions
  - Optional follow-up research project and

- Sponsored by UIUC, NCSA, Microsoft, NVIDIA, VSCSE

Three keynotes

NVIDIA, Microsoft Research, UIC Medical Center

Multi-disciplinary Panel

Cosmology, CFD, Molecular Dynamics, Microsoft e-science

Pleasant Side-Effects

- "TAs have top priority for getting their bug reports through, same as the largest NVIDIA partners"
  - Great professional experience for TA's
- Many are following the course
  - E-mail questions and calls from major microprocessor companies
- Many universities (~50) are offering courses based on the NVIDIA/UIUC material

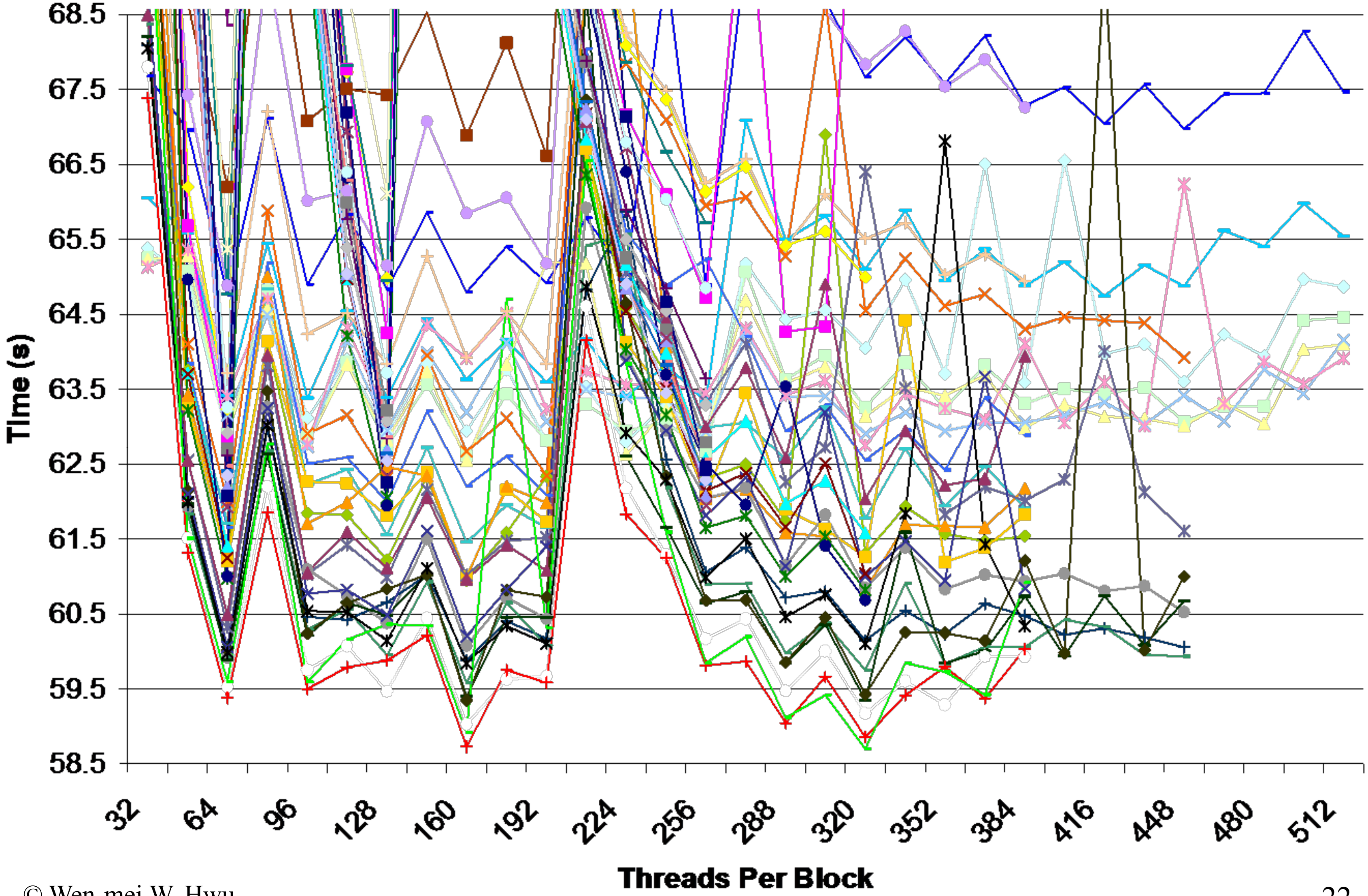
Pleasant Side-Effects (cont.)

- Research Advancement
  - QP Cluster supports many research projects
  - Supported a successful NIH Resoucre site visit
  - New NSF and NIH center proposals
  - A Vibrant IACAT research community
- NVIDIA and UIUC collaborated on HotChips presentation in August 2007
  - Previous Intel/UIUC collborative HotChips presentation 2001 on Itanium

Research Needs Identified

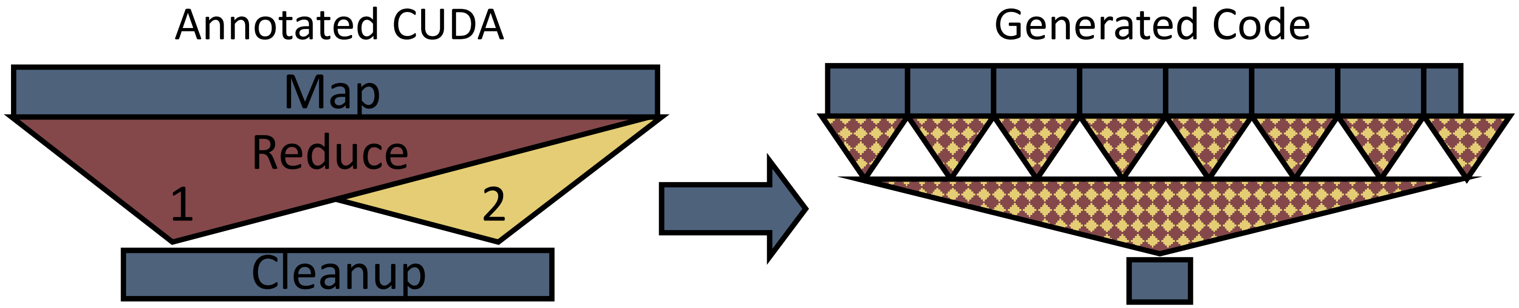
- Simple parallelism
  - Simple, logical forms of parallelism for application programmers
- Power tools
  - Leverage and strengthen app development frameworks
  - Empower tools with specification, analysis and domain information

Reduced Tuning Efforts



High-level Frameworks

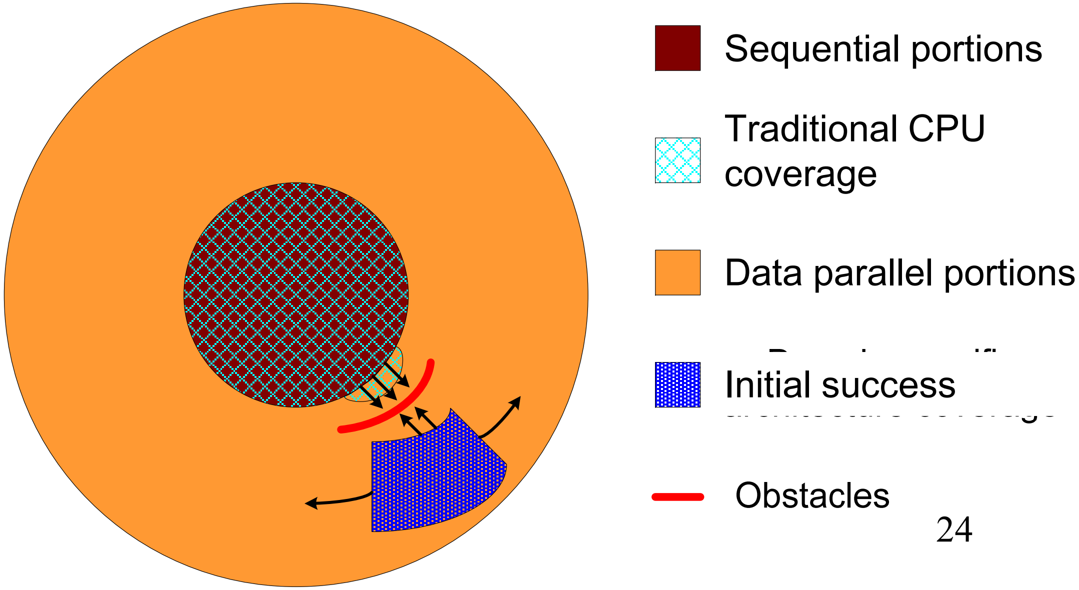
- Programming many-core GPUs requires restructuring computations to get around its coordination limitations
  - Global communication is very complicated
- Approach: put this complication in a code generation framework
  - Coordination is made explicit by expressing computation as MapReduce
  - User specifies set of reduction functions, map & cleanup functions
  - Framework generates efficient multistage reductions implemented in CUDA kernels

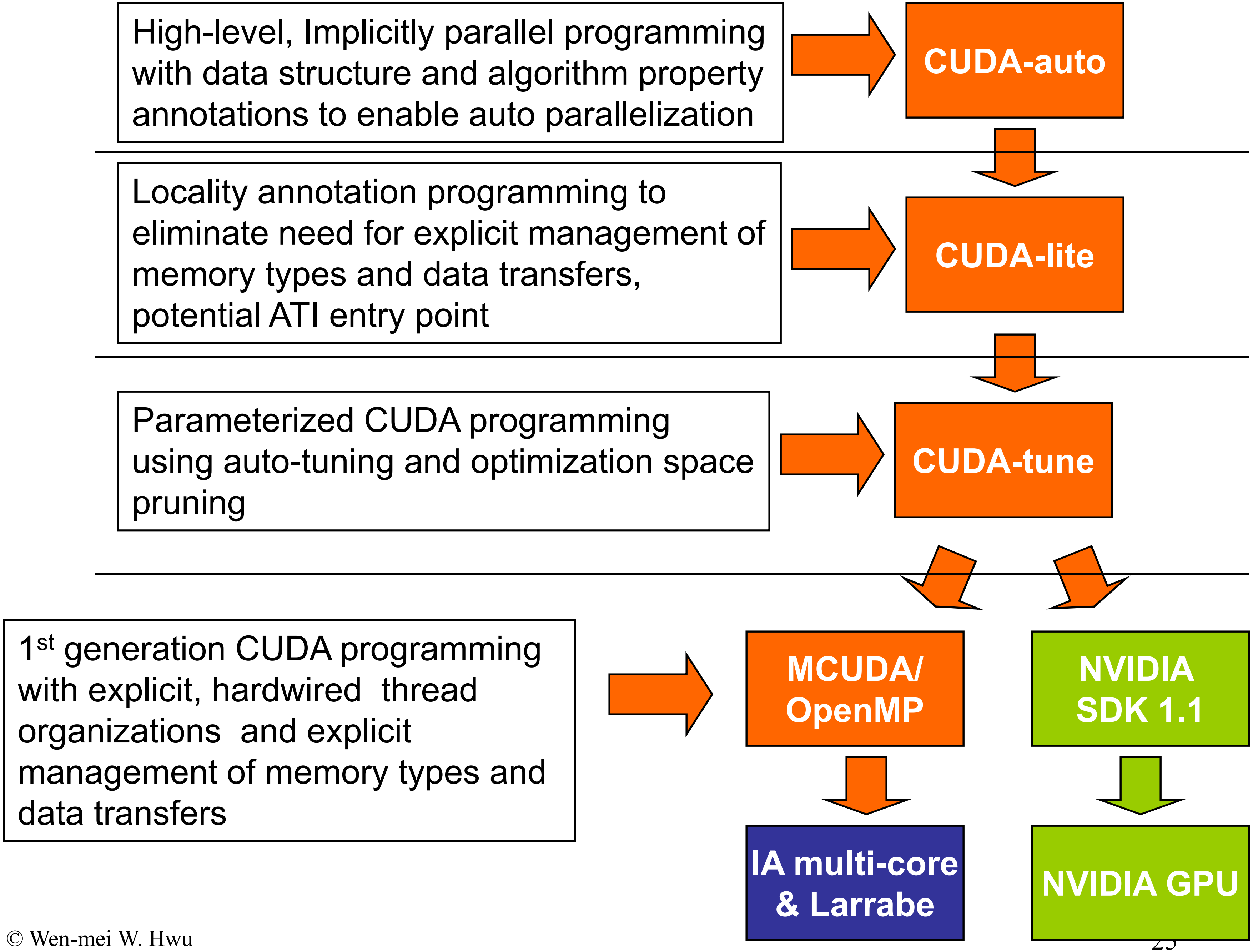


Collaboration with Keutzer, UCB

Reaching deeper into apps.

- Many data parallel apps have a small number of simple, dominating components
  - – Low hanging fruit for parallel computing (meat)
- Small computation components often dominate after initial low hanging fruits are picked
  - – Some are much more difficult to parallelize (pit)





High-level, Implicitly parallel programming with data structure and algorithm property annotations to enable auto parallelization

CUDA-auto

Locality annotation programming to eliminate need for explicit management of memory types and data transfers, potential ATI entry point

CUDA-lite

Parameterized CUDA programming using auto-tuning and optimization space pruning

CUDA-tune

1st generation CUDA programming with explicit, hardwired thread organizations and explicit management of memory types and data transfers

MCUDA/ OpenMP

NVIDIA SDK 1.1

IA multi-core & Larrabe

NVIDIA GPU

Yes, we are writing a textbook

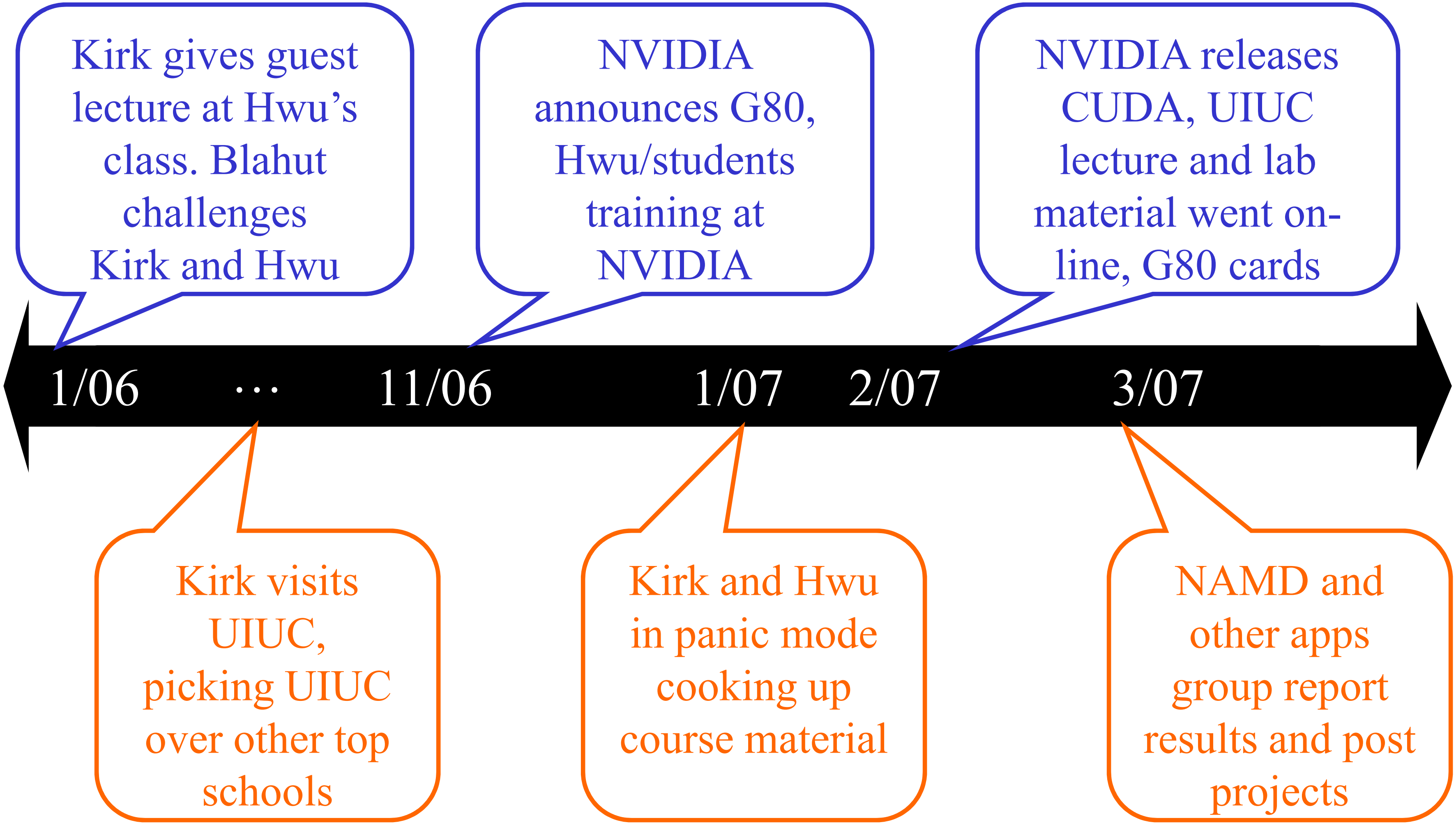
Programming Massively Parallel Processors Using CUDA

- The first four chapters in preliminary draft form now available
- We very much welcome your feedback!

Yes, more versions are coming.

- Computational education for scientists and engineers should start by junior year.
  - Computational thinking should be as pervasive as calculus
  - We are working on a version of the course suitable for sophomores.
- We are working towards offering the course to 10,000 science and engineering students/year
  - Electronic delivery of expert lectures on computational thinking and programming techniques
  - Local, customized instructions on domain problem solving
  - On-line global community forums for students and instructors.
  - On-line, continuous student/teacher feedback and discussion forums for continuous improvement.

Early Development History



1/06 … 11/06 1/07 2/07 3/07

Kirk gives guest lecture at Hwu's class. Blahut challenges Kirk and Hwu

NVIDIA announces G80, Hwu/students training at NVIDIA

NVIDIA releases CUDA, UIUC lecture and lab material went on-line, G80 cards

Kirk visits UIUC, picking UIUC over other top schools

Kirk and Hwu in panic mode cooking up course material

NAMD and other apps group report results and post projects

Acknowledgement

- NVIDIA contributors:
  - David Kirk – co-instructor and co-author
  - Kitchen crew, etc
  - Michael Shebanow, John Nicols, Dan Vivoli – guest lectures
- UIUC contributors:
  - John Stone, Sam Stone – application porting, tools feedback, performance insights, guest lectures
  - Kuangwei Hwang, John Stratton, Xiao-Long Wu – lab set up, lab assignments, lab solutions, web site, recordings
  - Robert Brunner, Klaus Schulten, Todd Martinez, Justin Haldar, Jianming Jing, and many more. - final project mentoring

Thank you! Any Questions?