CSCI-GA.3033-004

Graphics Processing Units (GPUs): Architecture and Programming

Computational Thinking

Mohamed Zahran (aka Z)
mzahran@cs.nyu.edu
http://www.mzahran.com
Computational Thinking 101

Computational Thinking is arguably the most important aspect of parallel Application development!
J. Wing Communications of the ACM, 49(3), 2006

What is it?
Decomposing a domain problem into well-defined, coordinated work units that can Each be realized with different numerical methods and well-known algorithms.
Why Do We Need Parallel Computing in the First place?

To solve a given problem in less time

To solve bigger problems

To achieve better solutions for a given problem and a given amount of time

Increased Speed!
Applications that are good candidates for parallel computing:

- Involve large problem sizes
- Involve high modeling complexity

Formulating the problem is crucial!!

The problem must be formulated in such a way that it can be decomposed into subproblems that can be executed at the same time.
The Process of Parallel Programming

- Problem decomposition
- Algorithm selection
- Implementation in a language
- Performance tuning

This is what we have been doing till now!
Problem Decomposition

Identify the work to be performed by each unit of parallel execution → thread in CUDA
Problem Decomposition:
Thread Arrangement

Example: Electrostatic Map Problem

Atom-centric: each thread responsible for calculating the effect of one atom on all grid points → Scatter

Grid-centric: each thread calculates the effect of all atoms on a grid point → Gather
Which is better?

- Gather is desirable
  - Threads can accumulate their results in their private registers.
  - Multiple threads share input atom values.
Problem Decomposition

• Picking the best thread arrangement requires the understanding of the underlying hardware.

• A real application consists of several modules that work together
  – Amount of work per module can vary dramatically
  – You need to decide if a module is worth implementing in CUDA

• Amdahl’s law
Algorithm Selection

• An algorithm must exhibit three essential properties:
  – definiteness = no ambiguity
  – effective computability = each step can be carried by a computer
  – finiteness = guaranteed to terminate

• When comparing several algorithms, take the following factors into account:
  – Steps of computation
  – Degree of parallel execution
  – Numerical stability
  – Memory bandwidth
Skills needed to go from: Parallel Programmer to: Computational Thinker

- Computer Architecture
- Programming models and compilers
- Algorithm techniques: (e.g. tiling)
- Domain knowledge
Conclusions

• Computational thinking is an art but a very crucial one.

• Jumping from problem definition to coding right away is the worst thing you can do!