Colorado State University

CS/ECE560: Foundations of Fine Grain Parallelism Introduction

Sanjay Rajopadhye
Colorado State University

Outline

- Class objectives, goals, introduction
- CUDA performance tuning (wrap up)
- Equational Programming (intro)

Parallelism is (now) ubiquitous

- Parallel Programming is hard
  - "End of the free lunch" [Sut05]
  - Arrival of "manycores" signals the end of "La-Z-Boy Programming" [Pat06]

  Becoming a parallel programming expert will get you a good job

  But your skills may become obsolete – new machines, new languages, …

  Parallelism must return to La-Z-Boy programming

[Sut05] Herb Sutter. "The Free Lunch Is Over: A Fundamental Turn Toward Concurrency," in *Software. Dr. Dobb's Journal,* vol. 30, no. 3, 2005.

[Pat06] David Patterson, in keynote talk at the International Workshop on Languages and Compilers For Parallel Computers *LCPC 2006, New Orleans, LA.*

Class Objectives

- Short term

  Become macho GPU programmer: write "heroically tuned" codes.

- Medium term

  Do it systematically: tuning for GTX 280 vs tuning for GTX 465: learn principles, not skills

- Long term

  Do it automatically: Learn the foundations of automatic compilation. Focus on a "regular subset" of programs

- Polyhedral Equational Model

Class Approach & Outline

- Big picture
  - Polyhedral Equations as programs: I'm loath to write C, despite the slogan "C no evil"
  - Equations vs (conventional) loop programs
- Equations-to-code (compiling equations)
  - Schedule
  - (processor) allocation
  - (memory) allocation
- But what about parallelism?

Class Approach & Outline …

10 assignments (basic + advanced) + term project

- CUDA performance tuning (2)
- Equational programming: Alpha/AlphaZ (1)
- Mathematical foundations: polyhedra, affine functions, and operations (2)
- Alpha analysis/transformation (1)
- Analysis: scheduling & allocation (2)
- Code generation/tiling (2)

Class grade breakdown

- Assignments (30%)
- Midterm (take home) (30%)
- Final project (30% = 2+3+5+15+5)
  - Proposal
  - Advancement report
  - Final report
  - Quality of work
  - Final poster
- Participation/Discussion/Quizzes (10%)

The Polyhedral Model

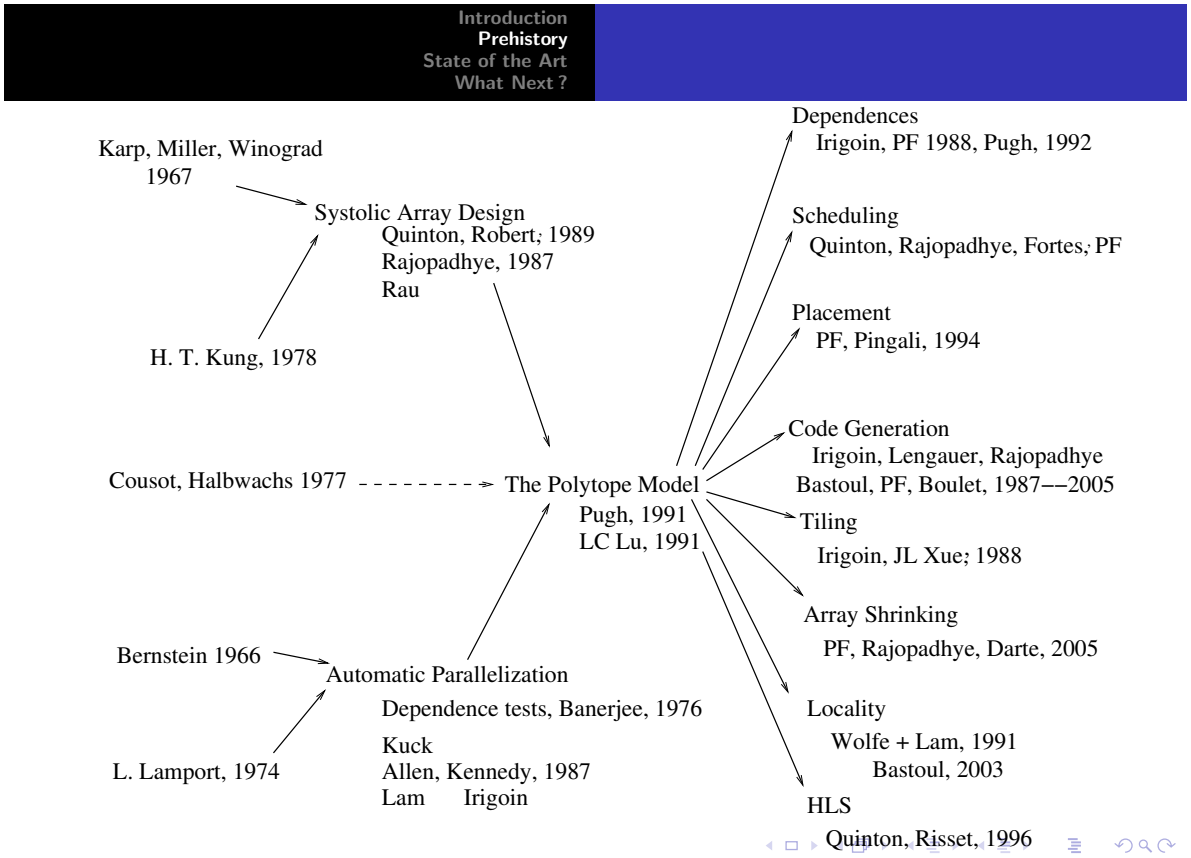
- What are polyhedra?
- Why are they useful/important
- What is the polyhedral model?

Model

- What is a model?
  - A mathematical/computational/mechanical/ … abstraction of some other (physical) entity
  - Objects in the model must "emulate" the "natural operations" of the modeled entities – semantics

(his) story

From Feautrier's keynote at LCPC 2009



"Real" vs "Abstract"

- Physical entity: programs/computations
- The Polyhedral Model is a "very high level" intermediate representation (IR) of "regular computations"
- Polyhedral equational model: real=abstract
- Amenable to:
  - Mathematical static analysis
  - Transformation within model: closure
  - Transformation outside model: (tiled) code generation

Outline

- ~~Class objectives, goals, introduction~~
- CUDA performance tuning (wrap up)
- Equational Programming (intro)?

Tuning (GTX 280 class GPU)

- Many resources on the web (NVIDIA webinars)
- Coalescing (HW1a)
  - Challenge question: Achieve maximum bandwidth, with fewest threads-per-block
  - For a "strided-by-block" access pattern.
- Arithmetic peak: warps and "virtualization"
- Bank conflicts in shared memory

Look ahead at HW2

- MAXPYrep:
  - Repeatedly execute `Y=A*X+Y`
  - Where A, X and Y are matrices
  - Matrices are small enough to fit in shared memory (ignore global memory access coalescing)
  - Goal: achieve machine peak
- Port all previous performance to GTX 480
  - And beyond …
  - Teach me

CUDA Tuning resources

- Oxford CUDA conf (CUDA webinar online)
- "Identifying Performance Limiters," Micikevicius NVIDIA/UCF (CUDA webinar)
- "Roofline for Fast Math" Sam Williams, LBL

Equational Programming

- Wiki page for Pascal's Triangle

  http://en.wikipedia.org/wiki/Pascal's_triangle

- … and also a non-standard way to compute Fibonacci numbers