18-742 Fall 2012
Parallel Computer Architecture
Lecture 1: Introduction

Prof. Onur Mutlu
Carnegie Mellon University
9/5/2012

Agenda

- Course logistics, info, requirements
  - Who are we?
  - Who should take this course?
  - What will you learn?
  - What will I expect?
  - Policies
- Homework and reading for next time
- Some fundamental concepts

Quiz 0 (Student Info Sheet)

- Due Sep 7 (this Friday)
- Our way of getting to know about you fast
- All grading predicated on passing Quiz 0
  - But, you are not in this room for grades anyway

Course Info: Who Are We?

- Instructor: Prof. Onur Mutlu
  - onur@cmu.edu
  - Office: Hamerschlag Hall A305
  - Office Hours: W 2:30-3:30pm (or by appointment)
  - http://www.ece.cmu.edu/~omutlu
  - PhD from UT-Austin, worked at Microsoft Research, Intel, AMD
  - Research interests:
    - Computer architecture, hardware/software interaction
    - Many-core systems
    - Memory and storage systems
    - Improving programmer productivity
    - Interconnection networks
    - Hardware/software interaction and co-design (PL, OS, Architecture)
    - Fault tolerance
    - Hardware security
    - Algorithms and architectures for genomics and embedded systems

Course Info: Who Are We?

- Teaching Assistant: HanBin Yoon
  - hanbinyoon@cmu.edu
  - Office: CIC 4th Floor
  - Office hours: MF 3:15-4:15pm (or by appointment)

What This Course is About

- Goal 1:
  - Build a strong understanding of the fundamentals of the architecture of parallel computers and the tradeoffs made in their design.
  - Examine how architectures are designed to exploit and extract different types of parallelism. The focus will be on *fundamentals*, *tradeoffs in parallel architecture design*, and *cutting-edge research*.
- Goal 2:
  - Do research in parallel computer architecture. You will conduct 1) a literature survey of very recent papers on a topic and 2) an open-ended research project to advance the state of the art.
  - Get familiar with and critically analyze research papers. Deliver technical talks of both your survey and project findings to the entire class.

Who Should Take This Course?

- This course is entirely optional: advanced graduate course
- You should be self-motivated and enthusiastic about doing research in computer architecture

- Must have *done well* in Graduate Comp Arch (740 or 18-741)
  - B or above
  - If not, you have to convince me you know the required basics

- Must be enthusiastic enough to
  - Work very hard
  - Read and critically analyze a lot of research papers
  - Pace yourself without deadlines; be strongly self motivated
  - Discover on your own (research project)
  - Make a difference (advance the state of the art)

Where to Get Up-to-date Course Info?

- Website: http://www.ece.cmu.edu/~ece742
  - Syllabus and logistic information
  - Lecture notes
  - Readings
  - Project info
  - Review site
- Blackboard: Linked from website
  - For you to upload your work
- Your email
- Me and the TAs

Lectures and Course Schedule

- Reserved Lecture Times:
  - MWF 4:30-6:20pm
  - Doherty Hall 1112
  - I intend to lecture 2/3 days on average. Days and load will be determined dynamically, usually in the previous week.
  - No lecture → discussions on literature surveys, papers, and projects

- Tentative schedule in your syllabus
  - But don't believe all of it
  - Systems that perform best are usually dynamically scheduled.
    - Static vs. Dynamic Scheduling
    - Why do you *really* need dynamic scheduling?

Static versus Dynamic Scheduling

- Static: Done at compile time or parallel task creation time
  - Schedule does not change based on runtime information
- Dynamic: Done at run time (e.g., after tasks are created)
  - Schedule changes based on runtime information
- Example: Instruction scheduling
  - Why would you like to do dynamic scheduling?
  - What pieces of information are not available to the static scheduler?

Parallel Task Assignment: Tradeoffs

- Problem: N tasks, P processors, N>P. Do we assign tasks to processors statically (fixed) or dynamically (adaptive)?

- Static assignment
  + Simpler: No movement of tasks.
  - Inefficient: Underutilizes resources when load is not balanced

    When can load not be balanced?

- Dynamic assignment
  + Efficient: Better utilizes processors when load is not balanced
  - More complex: Need to move tasks to balance processor load
  - Higher overhead: Task movement takes time, can disrupt locality

Parallel Task Assignment: Example

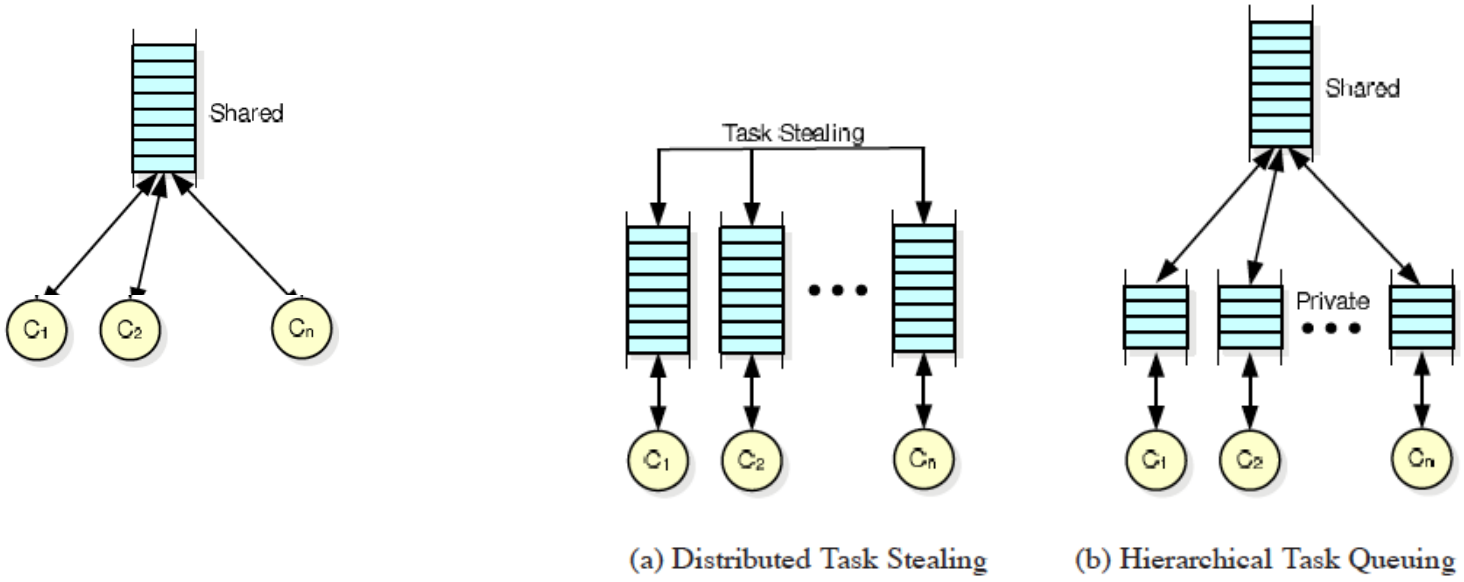
- Compute histogram of a large set of values
- Parallelization:
  - Divide the values across T tasks
  - Each task computes a local histogram for its value set
  - Local histograms merged with global histograms in the end

```
GetPageHistogram(Page *P)

    For each thread: {

        /* Parallel part of the function */
        UpdateLocalHistogram(Fraction of Page)

        /* Serial part of the function */
        Critical Section:
            Add local histogram to global histogram

        Barrier
    }

    Return global histogram
```

Parallel Task Assignment: Example (II)

- How to schedule tasks updating local histograms?
  - Static: Assign equal number of tasks to each processor
  - Dynamic: Assign tasks to a processor that is available
  - When does static work as well as dynamic?

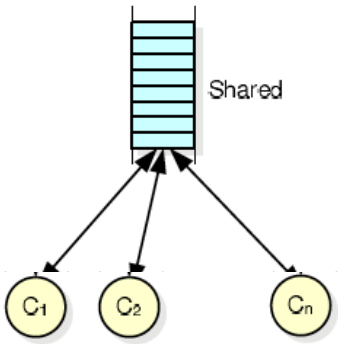
- Implementation of Dynamic Assignment with Task Queues

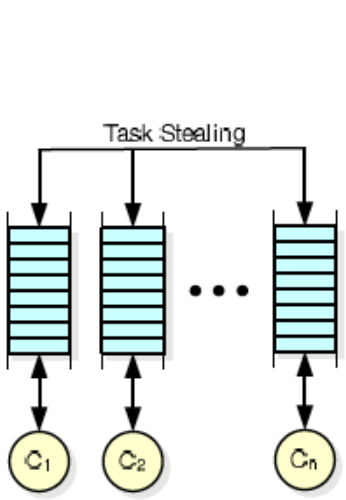


(a) Distributed Task Stealing (b) Hierarchical Task Queuing

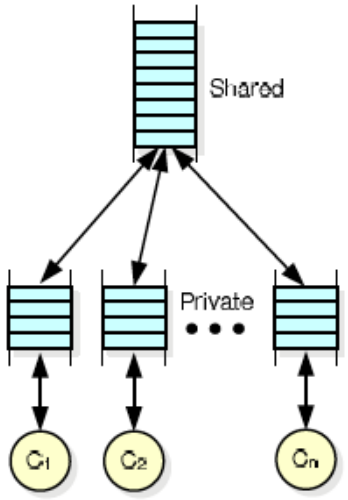
Software Task Queues

- What are the advantages and disadvantages of each?
  - Centralized
  - Distributed
  - Hierarchical





(a) Distributed Task Stealing



(b) Hierarchical Task Queuing

Task Stealing

- Idea: When a processor's task queue is empty it steals a task from another processor's task queue
  - Whom to steal from? (Randomized stealing works well)
  - How many tasks to steal?

+ Dynamic balancing of computation load

- Additional communication/synchronization overhead between processors
- Need to stop stealing if no tasks to steal

Parallel Task Assignment: Tradeoffs

- Who does the assignment? Hardware versus software?

- Software
  - + Better scope
  - - More time overhead
  - - Slow to adapt to dynamic events (e.g., a processor becoming idle)

- Hardware
  - + Low time overhead
  - + Can adjust to dynamic events faster
  - - Requires hardware changes (area and possibly energy overhead)

How Can the Hardware Help?

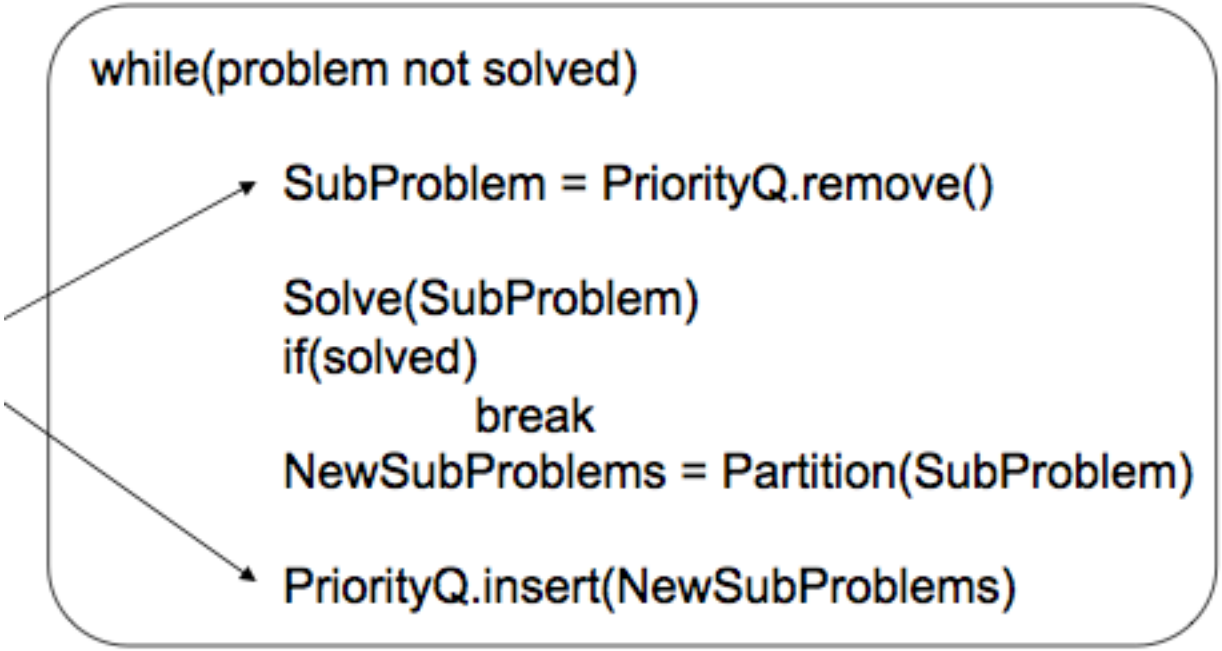
- Managing task queues in software has overhead
  - Especially high when task sizes are small

- An idea: Hardware Task Queues
  - Each processor has a dedicated task queue
  - Software fills the task queues (on demand)
  - Hardware manages movement of tasks from queue to queue
  - There can be a global task queue as well → hierarchical tasking in hardware

  - Kumar et al., "Carbon: Architectural Support for Fine-Grained Parallelism on Chip Multiprocessors," ISCA 2007.
    - Optional reading

Dynamic Task Generation

- Does static task assignment work in this case?
- Problem: Searching the exit of a maze



What Will You Learn?

- Parallel computer designs
  - State-of-the-art as well as research proposals
  - Tradeoffs and how to make them
  - Emphasis on cutting-edge research
- Hands-on research in a parallel computer architecture topic
  - Semester-long project
  - How to design better architectures (not an intro course)
- How to dig out information
  - No textbook really required
  - But, see the syllabus anyway

What Do I Expect From You?

- Learn the material
- And, research it → find the original source of ideas
- Do the work & work hard
- **Ask questions**, **take notes, participate in discussion**
- Come to class on time

- Start early and focus on the research project
- If you want feedback, come to office hours

- This class will definitely be tough
  - But you will have a lot of fun learning and creating

How Will You Be Evaluated?

- Research Project + Presentation + Poster: 40%
- Literature Survey + Presentation: 25%
- Exam: 20%
- Reviews, Class Participation, Quizzes, Assignments: 15%
- Our evaluation of your performance: 5%

- Grading will be back-end heavy. Most of your grade will be determined after late
  - How you prepare and manage your time will determine it

Policies

- No late assignments accepted
- Everything must be your own work (unless otherwise specified)
- Projects and Literature Survey in groups of 2 (maybe 3)
- Cheating → Failing grade
  - No exceptions

Assignments for Next Week

1. Review two papers from ISCA 2012 – due September 11, 11:59pm.

2. Attend NVIDIA talk on September 10 – write an online review of the talk; due September 11, 11:59pm.

3. Think hard about
   - Literature survey topics
   - Research project topics

4. Examine survey and project topics from Spring 2011

5. Find your literature survey and project partner

Paper Review Assignment for Next Week

- Due: Tuesday September 11, 11:59pm
- Required – Enter reviews in the online system
- Pick **two** papers from ISCA 2012 proceedings
  - At least one of them should be new to you
  - Cannot be a paper you have co-authored
  - http://isca2012.ittc.ku.edu/index.php?option=com_content&view=article&id=53&Itemid=57
- Read them thoroughly
- Write a "critical" review for each paper online
- Bonus: pick three papers instead of two

Review Assignment: NVIDIA Talk

- NVIDIA Tech Talk by ECE Alumnus, Philip Cuadra
- Monday, September 10, 2012 7-9 pm, HH-1107
- Refreshments will be provided.

- Inside the Kepler GPU Architecture and Dynamic Parallelism
- This talk will dive into the features of the compute architecture for "Kepler" – NVIDIA's new 7-billion transistor GPU. From the reorganized processing cores with new instructions and processing capabilities, to an improved memory system with faster atomic processing and low-overhead ECC, we will explore how the Kepler GPU achieves world leading performance and efficiency, and how it enables wholly new types of parallel problems to be solved. The software improvements to expose the new dynamic parallelism features of the architecture will also be discussed.

- Your job: Attend the talk and write a review of the talk.

How to Do the Paper/Talk Reviews

- Brief summary
  - What is the problem the paper is trying to solve?
  - What are the key ideas of the paper? Key insights?
  - What is the key contribution to literature at the time it was written?
  - What are the most important things you take out from it?
- Strengths (most important ones)
  - Does the paper solve the problem well?
- Weaknesses (most important ones)
  - This is where you should think critically. Every paper/idea has a weakness. This does not mean the paper is necessarily bad. It means there is room for improvement and future research can accomplish this.
- Can you do (much) better? Present your thoughts/ideas.
- What have you learned/enjoyed most in the paper? Why?

- Review should be short and concise (~half a page or shorter)

Advice on Paper/Talk Reviews

- When doing the reviews, be very critical
- Always think about better ways of solving the problem or related problems
- Do background reading
  - Reviewing a paper/talk is the best way of learning about a research problem/topic
- Think about forming a literature survey topic or a research proposal

Literature Survey

- More information to come
- Read a lot of papers; find focused problem areas to survey papers on

Research Project (I)

- Your chance to explore in depth a computer architecture topic that excites/interests you
- Your chance to publish your innovation in a top computer architecture/systems conference.

- Start thinking about your project topic from now!
- Interact with me, Han, and other students in SAFARI group

- Groups of 2 (in some cases, could be 3)
- Proposal due: Sep 21

- See old website for project handout and topics
- Discover topics on your own by reading heavily

Research Project (II)

- Goal:
  - Develop new insight
  - Approach 1:
    - Develop novel ideas to solve an important problem
    - Rigorously evaluate the benefits and limitations of the ideas
  - Approach 2:
    - Derive insight from rigorous analysis and understanding of previously proposed ideas
    - Propose potential new solutions based on the new insight
- The problem and ideas need to be concrete
- You should be doing problem-oriented research

Research Proposal Outline

- The Problem: What is the problem you are trying to solve
  - Define clearly.
- Novelty: Why has previous research not solved this problem? What are its shortcomings?
  - Describe/cite all relevant works you know of and describe why these works are inadequate to solve the problem.
- Idea: What is your initial idea/insight? What new solution are you proposing to the problem? Why does it make sense? How does/could it solve the problem better?
- Hypothesis: What is the main hypothesis you will test?
- Methodology: How will you test the hypothesis/ideas? Describe what simulator or model you will use and what initial experiments you will do.
- Plan: Describe the steps you will take. What will you accomplish by Milestone 1, 2, 3, and Final Report? Give 75%, 100%, 125% and moonshot goals.

All research projects can be and should be described in this fashion.

Heilmeier's Catechism (version 1)

- What are you trying to do? Articulate your objectives using absolutely no jargon.
- How is it done today, and what are the limits of current practice?
- What's new in your approach and why do you think it will be successful?
- Who cares?
- If you're successful, what difference will it make?
- What are the risks and the payoffs?
- How much will it cost?
- How long will it take?
- What are the midterm and final "exams" to check for success?

Heilmeier's Catechism (version 2)

- What is the problem?
- Why is it hard?
- How is it solved today?
- What is the new technical idea?
- Why can we succeed now?
- What is the impact if successful?

- http://en.wikipedia.org/wiki/George_H._Heilmeier

Supplementary Readings on Research, Writing, Reviews

- Hamming, "You and Your Research," Bell Communications Research Colloquium Seminar, 7 March 1986.
  - http://www.cs.virginia.edu/~robins/YouAndYourResearch.html
- Levin and Redell, "How (and how not) to write a good systems paper," OSR 1983.
- Smith, "The Task of the Referee," IEEE Computer 1990.
  - Read this to get an idea of the publication process
- SP Jones, "How to Write a Great Research Paper"
- Fong, "How to Write a CS Research Paper: A Bibliography"

Class Schedule for This and Next Week

- Friday, Sep 7:
  - Short lecture; could be cancelled
  - Turn in your Quiz 0 (Student Information Form) to myself or Han – attach your photo

- Monday, Sep 10: Full lecture (2 hours)
- Wednesday, Sep 12: Full lecture (2 hours)
- Friday, Sep 14: Full lecture (2 hours)

Next Lecture(s)

- Basics of Parallel Processing

- Outline of required readings (no reviews required yet):
  - Hill, Jouppi, Sohi, "Multiprocessors and Multicomputers," pp. 551-560 in Readings in Computer Architecture.
  - Hill, Jouppi, Sohi, "Dataflow and Multithreading," pp. 309-314 in Readings in Computer Architecture.

  - Suleman et al., "Accelerating Critical Section Execution with Asymmetric Multi-Core Architectures," ASPLOS 2009.
  - Culler & Singh, Chapter 1