Midterm Review

• Last Time
  » Web Indexing and Matrix Inversion
  » How to Make Page Rank go fast…

• Today
  » Midterm Review

• Reminders/Announcements
  » Midterm Thursday, April 28 (In class)
  » Please come on time; We’ll start on time

Coverage

• Lectures: Overheads, Discussions
• Readings: Textbooks, a few other articles
• Homeworks: Parallelism, Synchronization, Java, Performance
• Sections: Deeper Understanding
Readings

- Lea, Chapter 4.4
- One or two of the following: Javasoft Thread Tutorial OR Arnold/Gosling, The Java Programming Language, Chapter 9 OR 1996 Thread Programming Articles, Part 1 and Part 2
- Lea, Chapter 2, 1.2, 4.2
- => Not the optional readings occasionally mentioned in class

What is Parallelism?

- Importance of parallelism
  » Foundation of Performance
  » Has been in the past (pipelining, superscalar)
  » Continue in the future (multithreading, multi-core, multi-node)
- Technology
  » Hardware Drivers for Increased Parallelism
  » Smaller Transistors, slow increase in clock rates
  » Speed of light limitations
  » Difficulty of more “implicit parallelism”
  » Multi-core is coming; all machines are parallel
  » 10,000 – 100,000 node machines today, Millions in the future!
What is Parallelism? (cont.)

- Applications
  - Many large-scale applications have exploding needs for computation
  - Major sources
    - Deeper and more complex analysis: Example: detailed Modeling, Image processing, rendering
    - Massive Data: point of sale, video observation, sensor networks
    - Massive Activity: inventory and sale, web activities (massive human), computer-computer (frequent flyer!)

- Scale up/scale forward
  - Scale up for large-scale performance
  - Scale forward for continued rapid increases (as expected due to Moore's law)

Parallelism in Java

- Threads
  - Basis for Parallelism in Java – each an independently executable locus of control
  - Derivation from Thread Class, Define Runnable Interface
  - Relating Parallelism to Sequential Classes and Programs

- Synchronization
  - Synchronized Blocks and Methods
  - Threads and Locks, “Don’t break sequential code”
  - Making Data Structures “concurrency safe” and tension with flexible parallelization (a la Jacobi iteration)
  - Deadlocks: Set of threads waiting for locks which will never be satisfied
  - Livelock: Set of threads chasing each other, never will stop
Parallelism in Java (cont.)

• Thread Coordination and Scheduling
  » Wait(), Yield(), Notify(), NotifyAll()
  » Why you would want to use them
  » How to use them
  » Pitfalls in Using them (correctness)
  » Costs of Using them

• Advantages of Java
  » Integrated Model of Threading, Parallelism, Synchronization
  » Nice, clean, portable interfaces to threading, locks, parallelism, etc. (this CAN be ugly)
  » Reasonable type integration; no need for lots of casts, etc.

Architecture and Hardware

• Implicit vs. Explicit parallelism
  » Pipelining and Superscalar
  » Multithreading (including SMT), Multiple Processors, and Multiple-Nodes

• Multiprocessors/threads (Shared Memory)
  » Parallelism against a shared memory (like Opterons)
  » Locks and Synchronization
  » Caches and Locality; relative performance
  » Limited Scalability
Architecture and Hardware (cont.)

• Multi-Node/Cluster (Distributed Memory)
  » Local SMP? Sometimes
  » Message Passing
  » Massive Scalability

• Range of Systems
  » Small scale, large scale
  » Multi-core coming in large numbers

Applications

• Three applications
  » Relevant Application Definition
  » Data Sets
  » Algorithms
  » Practical Aspects of Parallelism

• Parallel Sorting
  » Shared Address Space Algorithms
  » Heroic Sorting: Minute Sort, Terabyte Sort, Penny Sort
    – Initial Exposure to “Benchmarks”
  » Scalable Algorithms: Bucket Sort and Challenges
  » Realities of Implementation
    – What is easy, what is hard
Applications (cont.)

• Jacobi Iteration: Partial Differential Equation Solver
  » Widespread Use in Modeling: fluids, air, crashes, etc.
  » Floating Point and Linear Systems
  » Simple, Regular Structure
  » Parallelism Structure is often regular; Lots of Parallelism!
  » Threads can ensure correct execution by convention; not individual object locking -> much simpler, more flexible programs

• Web Search/Indexing
  » Practical Constraints of Web: Huge, Never Static!
  » Basic Indexing for Word Counts
  » Efficient and Scalable Retrieval of Matching Pages
  » PageRank Algorithm
  » Composing Responses to Page Ranked Requests

Exam Format

• Short Questions (~50 %)
  » Coverage of All Course Topics to Date (Parallelism, Java, Machines, Applications)
  » Write a sentence or two
  » Demonstrate Understanding and Depth in an Area

• Examples (guaranteed to NOT be on the exam!)
  » “Explain the notion that Java threads hold locks, and give an example of how this affects programming.”
  » “Define Implicit Parallelism and Explicit Parallelism from a Programming Perspective”
  » “What level of parallelism is present in current-day microprocessors, and how is it likely to change in the future?”
Exam Format (cont.)

- Two Longer Questions (~50 %)
  - Understand a Java Program
  - Modify Code
  - Write Code

- Illuminate the concepts and ideas that we have covered
- Homeworks problems are a fertile area for preparation here

TA Midterm Review Section

- Sagnik Nandy will hold a Review and answer questions about course material
  - Readings
  - Homeworks
  - Lectures
  - Sections

- Wednesday April 27, 2-4pm, Location 4218 APM
Summary

- Midterm has comprehensive Coverage of Course Material to this point
- Elements
  » Lectures, Homeworks, Readings, Sections
- Topics
  » Parallelism
  » Java, Threads, Synchronization
  » Parallel Machines
  » Applications

Questions?