Review Session

Exam Comments

- 3:00pm-5:00pm Room **SB111** on 11/14/19
  - Closed book, closed notes, no cell phone, no calculator
  - Allow to bring a 4''x6'' "cheat card"
  - Anything on the card **MUST HAND WRITING**
- All lectures included on the test
  - Focus on things taught in class (lecture notes, in-class discussions, homework, programming assignment)
- Exam structure:
  - Short answer questions
  - `big' questions
    - Programming design related questions
    - Performance analysis/evaluation questions
- Good luck!

Work Opportunities

- Research opportunities for graduate students:
  - Always look for self-motivated and hard-working grad students
  - Ph.D. students: CS597 and CS691
  - MS students: CS591 “Research and Thesis for MS Degree”
  - Take CS546 & CS550, check my research projects, send me your CV
- Research opportunities for undergrad students:
  - NSF REU (Research Experiences for Undergraduates) with Prof. Xian-He Sun
    - Various project topics, including development of scheduling simulator, analysis of system logs, ….
    - UG students: CS491
    - Pay per hour
    - If interested, contact Prof. Sun (sun@iit.edu)

Introduction

- Evolution of Computing:
  - Machine, Grid, Pervasive, Cloud, Edge, and Big Data
- Overview of parallel processing:
  - What is, driven force, aspect
  - Where the performance come from?
  - Multicore, and manycore architecture
  - Challenges of parallelism
  - Memory-wall, and memory hierarchy
  - Principle of locality
  - Shared and distributed memory machines
- Technology advance

Architecture

- Basic components of any architecture:
  - Processors and memory (processing units)
  - Interconnect: static & dynamic (performance factore)
- Logic classification based on:
  - Control mechanism (Flynn's Taxonomy)
    - SISD, <u>SIMD, </u>MISD, <u>MIMD</u>
  - Address space organization
    - Shared Address Space (NUMA)
    - Distributed Address Space
- Cluster
  - Interconnect
  - Parameters, performance

Software

- Important software
  - OS, Compiler, Libraries, Tools, Timing and Profiling, Schedulers, Debugger, Distributed file systems, Parallel virtual file systems
- Software design
  - Style of parallel processing
  - Approach to parallel programming
  - Thread
    - Ptherad
    - Synchronization primitives
- Parallel File Systems
  - What is parallel I/O
  - Data Distribution

Data and Task Parallelization

- Data parallel: C program loops where each iteration of a loop is independent and represents a simple statement and is executed on a different processor

```
for (i=0;i<1000;i++)
    a[i]=b[i]+c[i];
```

- Task parallel: multiple C program loops which cannot be parallelized individually, but the different code blocks are independent and are executed on different processor

```
for (i=0;i<1000;i++)  /*block 1 */
    b[i+1]=b[i]+c[i]
…
for (j=0;j<5;j++)
   a[j+1]=a[j]+d[j];
```

Dependencies

- No dependence (can run in parallel)

  S1: X=K+3;

  S2: Y=Z*5;

- True dependence (cannot run in parallel)

  S1: X=3;

  S2: Y=X*5;

- Anti dependence (cannot run in parallel)

  S1: Y=X*4;

  S2: X=3;

- Output dependence (cannot run in parallel)

  S1: X=Y*4;

  S2: X=3;

- Loop-carried dependence

Methodology of Parallel Algorithm Design

- Programming model
  - Shared memory, Message passing, Data Parallel, Hybrid
- Four stages of parallel algorithm design:
  - Partitioning (decomposition, assignment): deal with machine independent issues and affect concurrency and scalability
  - Orchestration
  - Mapping
    - Orchestration & mapping: deal with machine dependent issues and affect locality and other performance issues
- Performance goals
- Application structures
  - SPMD, Embarrassingly Parallel, Master/Slave, Work Pool, Divide and Conquer, Pipeline, Competition
- Example

What have covered in MPI lecture

- What is MPI?
- How to write a simple program in MPI
  - MPI Communicators and communication
- Running your application with MPICH
- Slightly more advanced topics:
  - Non-blocking communication in MPI
  - Group (collective) communication in MPI
- Homeworks

Algorithm Design: Tridiagonal solvers

- Parallel Algorithms
  - The Partition Method
  - The PPT Algorithm
  - The PDD Algorithm
  - The LU Pipelining Algorithm
  - The PTH Method and PPD Algorithm Partitioning : deal with machine independent issues and affect concurrency and scalability
- Communication patterns
- Algorithm analysis
- Program assignment

Algorithm Design: Sorting

- Algorithm design
  - Parallelizing sequential alg., design new parallel algorithm, borrowing know alg.
- Existing techniques
  - Balanced tree, Doubling, Partition, Divide-and-Conquer, Pipelining,
- Sorting
  - Bubble sort, Mergesort, Quicksort, Bitonic sort
- Algorithm analysis

What have covered in OpenMP lecture

- What is OpenMP?
  - MPI + X
- OpenMP Basic and API
  - Compiler Directives
  - Runtime Library Routines
  - Environnent Variables
- OpenMP slight more Advanced
  - Accelerator
  - Library/optimization
  - Examples
- Homework

Performance Evaluation

- Performance metrics
  - Speedup $$S_p = \frac{T_s}{T_p}$$
  - Efficiency $$E_p = \frac{S_p}{p}$$
  - Scalability
- **<u>Amdahl's law</u>** gives a limit on speedup in terms of α (fraction of program (algorithm) that is <u>serial</u> and <u>cannot be parallelized</u>)

$$T_p = \alpha T_s + \frac{(1-\alpha)T_s}{p}$$

$$S_p = \frac{T_s}{\alpha T_s + \frac{(1-\alpha)T_s}{p}} = \frac{1}{\alpha + \frac{1-\alpha}{p}}$$

Performance Evaluation

- Model of speedup
  - Fixed-size
  - Fixed-time
  - Memory-bounded
  - Generalized
- Characteristics of parallel code
  - granularity
  - load balance
  - locality
  - communication and synchronization
- Scalable computing
  - Scalability
  - Examples

Performance Evaluation 2

- Scalable computing
  - Scalability
  - Examples
- Isospeed scalability
  - Definition
  - Examples

Memory Performance Optimization

- Multicore
- Operation of Memory Hierarchy
  - Metrics
  - Cache, localities, cache miss (4Cs), associative, policies
  - Memory hierarchy performance and optimizaiton
- Application Level Optimizations
  - Loop transformations
  - Array padding
  - Cache blocking, etc.
  - Replacement policies
- Examples

Memory Performance Optimization 2

Metrics and their calculation

- AMAT
- APC
- C-AMAT
  - Motivation, definition, recursive
  - Calculation, relation with other metrics
  - optimization
- LPM and Sluice Gate Theory
- Examples
- Revisit

What have covered in performance tool lecture

- What and why Performance tuning?
- Tools
  - Mesurément and profile
  - Pitfalls
- Linux
- Assignment

Lecture – File system basics

- What is the main role of file systems?
- How are data represented?
- How can we search for our data?
- How can we share data between users and applications?
- How are file systems generally organized?
  - Client-server
  - Cluster
  - Symmetric

Lecture – PFS

- What are the main design goals of PFS? Why do we need them?
- What kind of workloads require a PFS?
- Typical access patterns?
  - Shared file
  - File-per-process
- Data distribution in PFS
- What is POSIX I/O?
- How is concurrent access achieved efficiently?

Lecture – HDFS

- What is MapReduce and how does it work?
- What are the core concepts of Hadoop?
- Which workloads are suitable for Hadoop?
- Learn a few examples:
  - Word count in class
  - Can you design your own solution for sorting integers using MapReduce?

Lecture – HDFS (cont.)

- Basic design principles of HDFS?
- Highlight the HDFS architecture
- Data distribution in HDFS
  - Block placement
- Fault tolerance
  - Heart beats
- Data replication
  - Replica placement
  - Replica selection

Lecture – HDFS (cont.)

- Data pipelining
  - Describe all three ways to achieve it
- Namenode
  - Performance
  - Failures
- HDFS optimizations
  - Space reclamation
  - Communication protocols
  - Metadata

Distributed Memory Architecture

- Overview of distributed memory parallel architectures
- Interconnection networks
  - Connection topologies, 2-D mesh, hypercube
- Message passing issues
  - Routing, store-and-forward
  - Group communication

Message Passing Performance & Shared Environments

- The Log(P) model
- The memory Log(P) model
- The probability model for non-dedicated environment
- Why do we need them and how to use them
- The matrix multiple performance analysis

Highlight

- General concepts
  - Architecture, dependence, etc.
- Algorithms
  - Parallelism and analysis
- Performance
  - Speedup, scalability
- Memory performance
  - Hierarchy and C-AMAT
- I/O and file systems
  - PFS and HDFS
- Communication
- Programming

Exam Comments

- 3:00pm-5:00pm Room **SB111** on 11/14/19
  - Closed book, closed notes, no cell phone, no calculator
  - Allow to bring a 4''x6'' "cheat card"
  - Anything on the card **MUST HAND WRITING**
- All lectures included on the test
  - Focus on things taught in class (lecture notes, in-class discussions, homework, programming assignment)
- Exam structure:
  - Short answer questions
  - `big' questions
    - Programming design related questions
    - Performance analysis/evaluation questions
- Good luck!