

SE498 Parallel Computing

Dr. Walter Schilling

Fall, 2013

You may use 1 8.5 x 11 inch sheet of paper with notes and other supporting material for the exam.

1 Detailed Outcomes for the Exam

1. Week #1

(a) Lecture #1 Introduction

- i. Explain the concept of Moore's law
- ii. Define high performance computing
- iii. Calculate the power for a given microprocessor at a given voltage and a given frequency
- iv. Define the term cluster
- v. Explain the relationship between voltage, frequency, and dissipated power.

(b) Lecture #2 Parallel Hardware

- i. Draw a picture of the von Neumann computer architecture
- ii. Explain the concept of caching
- iii. Define spatial locality and temporal locality
- iv. Define pipelineing and multiple issue
- v. List and define the categories of processing under Flynn's taxonomy

(c) Lab

- i. Explain the difference between interactive and batch processing systems
- ii. Understand how programs execute in a supercomputing environment.
- iii. Write and submit a simple job to a supercomputer for processing.

2. Week #2

(a) Lecture #1 Parallel Hardware - Part 2

- i. Compare and contrast a shared memory system with a distributed memory system from an architectural standpoint
- ii. Compare and contrast NORMA, NUMA, and UMA multicore systems
- iii. Draw a picture of a switched interconnect topology using a crossbar connection
- iv. Explain the concepts of a ring, a toroidal mesh, bisected ring, fully connected network, and hypercube architecture.
- v. Define the terms latency and bandwidth, and explain the implication on parallel computing.
- vi. Define the terms diameter, bisection width, and bisection bandwidth.
- vii. Given a topology and a node count, calculate the diameters and bisection width.

(b) Lecture #2 Software Overview

- i. Define the terms load balancing, parallelization, embarrassingly parallel
- ii. Define the concepts of speedup and efficiency
- iii. Explain Amdahl's law
- iv. Calculate the maximum speedup that can be obtained when parallelizing an algorithm
- v. Explain Foster's methodology and apply Foster's methodology to a simple problem

(c) Lab

- i. Explain the difference between interactive and batch processing systems

- ii. Understand how programs execute in a supercomputing environment.
- iii. Write and submit a simple job to a supercomputer for processing.
- iv. Analyze the performance of pthreads when parallelized
- v. Understand the impact of cache hits and misses on program performance

3. Week #3

(a) Lecture #1 OpenMP

- i. Explain the difference between OpenMP and pthreads
- ii. Compile and link a simple OpenMP Program
- iii. Explain the usage of the `#pragma omp` directive.
- iv. Explain the appropriate mechanism to check whether the compiler supports openMP
- v. Draw an image representing the openMP Programming model.
- vi. Define the terms team, master, slave, and implicit barrier as they are related to openMP.

(b) Lecture #2 OpenMP

- i. Explain the relationship between private and shared variables in OpenMP.
- ii. Explain the concept of the OpenMP critical pragma
- iii. Construct a segment of code using a reduction operator and OpenMP.
- iv. Explain the purpose for the schedule clause in openMP
- v. Implement a simple OpenMP algorithm for calculating a mathematical value.

(c) Lab

- i. Explain how the Jacobi algorithm works to calculate future diffuse values
- ii. Construct an openMP program which uses the Jacobi Algorithm to simulate heat transfer over a grid.
- iii. Simulate the heat flow in a simple system
- iv. Construct the fastest application to perform a given simulation.

4. Week #4

(a) Lecture #1 Parallel Algorithms

- i. Explain the operation of the bubble sort algorithm for sorting a list of numbers
- ii. Explain the difference between the bubble sort and the odd-even transposition sort
- iii. Construct a simple implementation of an odd-even sort which runs in parallel.
- iv. Explain the purpose of the Jacobi algorithm for solving mathematical equations
- v. Construct a simple parallel implementation of the Jacobi algorithm using OpenMP.

(b) Lecture #2 MPI Introduction

- i. Explain the limitations of OpenMP
- ii. Define the acronym MPI.
- iii. Explain the advantage of MPI over OpenMP in terms of hardware architecture.
- iv. Define in terms of MPI the term communicator, rank, and size.
- v. Explain how to execute an MPI program on a UNIX system.
- vi. Draw a picture showing how an MPI program is compiled.
- vii. Explain the purpose for the MPI Init, Finalize, Comm Size, and Comm rank methods.
- viii. Construct a simple MPI program using MPIU Send and MPI receive.

5. Week #5

(a) Lecture #1 MPI Part 2

- i. Explain how MPI programs deal with I/O and the side effects that may occur from using simply IO in an MPI program.
- ii. Explain the concept of tree structured communications.
- iii. Construct an MPI application using MPI reduce.
- iv. Explain the concept of a broadcast message.
- v. Explain the concept of scattering and gathering data.
- vi. Explain the concept of an MPI derived datatype.

- (b) Lecture #2 Application Development with Eclipse (Note: Not tested on exam)
 - i. Explain how one can use Eclipse to develop a program and run it remotely on the OSC computers
 - ii. Construct an MPI program to calculate a list of prime numbers.
- 6. Week #6
 - (a) Lecture #1 Midterm Exam Review
 - (b) Lecture #2 Midterm Exam
- 7. Week #7
 - (a) Lecture #1 Image Processing Introduction
 - i. Explain how an image is represented in memory.
 - ii. Explain the concepts of RGB encoding.
 - iii. Convert an RGB image into gray scale.
 - iv. Convert an RGB image into HIS color space.
 - v. Explain what happens if a constant value is added or subtracted from a picture value.
 - vi. Use the XOR operation to Overlay an image with another image.
 - vii. Stretch and compress the contrast of an image.
 - (b) Lecture #2 Image Processing Continued
 - i. Define the concept of convolution.
 - ii. Using convolution blur the output of an image.
 - iii. Using convolution, detect edges in an image.
 - iv. Explain the purpose for median filtering of an image.
- 8. Week #8
 - (a) Lecture #1 Open ACC
 - i. Draw a picture of a computer architecture which would be capable of using openACC.
 - ii. Define offloading and accelerator.
 - iii. Draw an architectural picture of a CPU using an accelerator.
 - iv. Draw a diagram showing the steps necessary to perform a data manipulation on an accelerator.
 - v. Explain how openAcc uses macros to support parallelization.
 - vi. Explain the purpose for the C restrict keyword.
 - (b) Lecture #2 OpenACC Part 2
 - i. Define the concepts of gangs and workers as they are related to openMP.
 - ii. Explain the purposes for the various openAcc data clauses.
 - iii. Compile an openACC program on the OSC machines.
- 9. Week #9
 - (a) Lecture #1 Cache Coherence
 - i. Explain the concept of cache coherence.
 - ii. Draw a shared memory system which potentially can exhibit a cache coherence problem.
 - iii. Explain the operation of snooping cache coherence.
 - iv. Critique the usage of snooping cache coherence as the scale of the system gets larger.
 - v. Explain the concept of directory based cache coherence.
 - vi. Explain the concept of false sharing.
 - (b) Lecture #2 PRAM Algorithms
 - i. Define the concept of a PRAM model of computation.
 - ii. Explain how to perform a merge sort in parallel
 - iii. Explain how using a matrix of relationships, it is possible to sort an array of n elements in $O(1)$ operations
- 10. Week #10
 - (a) Lecture #1 Special Topics
 - (b) Lecture #2 Exam Review and Course Evaluations