CPCS-414 Syllabus

Catalog Description

CPCS-414 High Performance Computing
Credit: 3 (Theory: 3, Lab: 0, Practical: 0)
Prerequisite: CPCS-361
Classification: Elective

The objective of this course is to provide an in-depth overview of the current state of the art in high-performance computing. Additionally, the course provides more information about the architecture of high-performance computers.

Class Schedule

Meet 50 minutes 3 times/week or 80 minutes 2 times/week

Lab/Tutorial 90 minutes 1 times/week

Textbook

Michael Jay Quinn, , "Parallel Programming in C with MPI and OpenMP", McGraw-Hill Science Engineering;(2004)

ISBN-13 9780072822564 **ISBN-10** 0072822562

Grade Distribution

Week	Assessment	Grade %
3	Quiz	5
8	Exam 1	15
9	Homework Assignments	10
14	Exam 2	15
15	Lab Exam	10
15	Project (Individual)	15
16	Comprehensive Final Exam	30

Topics Coverage Durations

Topics	Weeks
Introduction to High Performance Computing	2
Dependence Analysis and General Optimizations	2
Parallel Programming Models	1
Shared Memory Programming	2
Programming Distributed Memory Systems	3
Heterogeneous Computing	2
Parallel Algorithm Design	3

Last Articulated

February 12, 2017

Relationship to Student Outcomes

a	b	c	d	e	f	g	h	i	j	k
	x	x				x	x			

Course Learning Outcomes (CLO)

By completion of the course the students should be able to

1. Describe grand challenge applications and the need for parallel computation (g)
2. **Deduce the maximum theoretical speedup and scalability of a system given a sequential code using Amdahl's law (b)**
3. **Identify and eliminate dependencies in sequential code to make it amenable to parallelism (b)**
4. Apply appropriate cache and general optimizations to sequential code to reduce its computational time (c)
5. Identify the different phases in parallel program design using domain or functional decomposition (b)
6. Use the concept of fork-join model to parallelize sequential code using OpenMP directives (c)
7. **Choose appropriate OpenMP directives for scheduling, implementing mutual exclusion and applying reduction (c)**
8. Develop MPI programs using point-to-point communication while avoiding deadlocks (c)
9. **Choose appropriate MPI collective communication calls to minimize communication between processes (c)**
10. Identify opportunities in code to apply appropriate MPI reduction operations (c)
11. **Recognize the need for heterogenous computing, coprocessors and newer architectures and programming paradigms (h)**
12. Distinguish between workflows to program MICs and GPUs (b)
13. Design a parallel matrix mutiplication algorithm using a distributed memory model (c)
14. Parallelize Floyd's algorithm using point-to-point communication (c)
15. Parallelize numerical algorithms to solve recurring problems in physics and mechanical engineering (c)

Coordinator(s)

Dr. Waseem Ahmed Kareem Abdulkhayoom, Assistant Professor