Syllabus:
CS 5334/4390: Parallel and Concurrent Programming
Spring 2014

Instructor:
Shirley Moore
Office: CCSB 3.0422
Phone: 915-747-5883
Email: svmoore@utep.edu
Webpage: http://www.cs.utep.edu/svmoore/
Office hours: TR 2:00-2:50pm, others by appointment

Class time and location:
TR 3:00-4:20pm, BUSN 326

Course website: http://svmoore.pbworks.com/

Course description:
The goal of this course is to introduce students to the foundations of parallel computing including the principles of parallel algorithm design, numerical and non-numerical parallel algorithms, programming models for shared and distributed memory systems, analytical modeling of parallel programs, analytical modeling of parallel execution, and debugging and performance optimization of parallel programs. The course will include material on emerging multicore hardware, new shared-memory programming models (Intel Thread Building Blocks, Cilk), programming models for GPUs, and problem-solving on large-scale clusters using Google's MapReduce. A key aim of the course is for students to gain hands-on experience by writing efficient parallel programs in some of the programming models covered n class.

Prerequisites: Programming experience in C and/or Fortran, CS 2302 Data Structures, and CS 3432 Computer Architecture I; or permission of the instructor

Textbook: *Introduction to Parallel Computing, Second Edition*, Ananth Grama, George Karypis, Vipin Kumar, Anshul Gupta, Addison-Wesley, 2003, ISBN: 0201648652.

Learning outcomes:
After successfully completing this course, students should be able to

- DESCRIBE the major approaches to parallelism used in a large parallel program
- DESIGN and IMPLEMENT a decomposition strategy and parallel algorithm to solve a given numerical or non-numerical problem discussed in class
- IDENTIFY the regions of a large time-consuming program that are most conducive to increased performance through parallelization
- SELECT and APPLY appropriate parallelization constructs to a large program to create a correct parallel version that exploits a state-of-the-art parallel computing environment
- ASSESS the correctness of a parallel program,
- APPLY available debugging methods to detect and correct errors in an erroneous parallel program
- ASSESS the performance of a parallel programs run with different input sizes and numbers of processors

- APPLY performance optimization methods to improve the parallel efficiency and scalability of a parallel program
- DISCUSS current and future trends in parallel architectures and programming models

Course topics:
The following is a list of topics to be discussed. The exact schedule may vary depending on previous background of class participants. Although the instructor will do some lecturing, we will try to use a "flipped classroom" format to some extent that will require reading and preparation by class participants prior to class and involve group work during class.

- Principles of parallel algorithm design
    - decomposition techniques
    - mapping & scheduling computation
    - templates
- Non-numerical algorithms
    - sorting
    - graphs
    - dynamic programming
- Numerical algorithms
    - dense matrix algorithms
    - sparse matrix algorithms
- Parallel computer architectures
    - shared memory systems and cache coherence
    - distributed-memory systems
    - interconnection networks and routing
- Programming shared memory systems
    - OpenMP
    - Cilk/Cilk++
    - Pthreads
- Programming distributed memory systems
    - message passing: MPI
    - global address space languages, e.g. UPC, CAF, Chapel
- Analytical modeling of program performance
    - speedup, efficiency, scalability
    - cost optimality
    - isoefficiency
- Correctness assessment and debugging of parallel programs
- Performance analysis and optimization of parallel programs
- GPU Programming with CUDA and/or OpenACC
- Problem solving on clusters using MapReduce

Assignment, exams, and grading:
There will be approximately six homework assignments, most of which will involve programming. There will be one "midterm" exam, to be given approximately 2/3 of the way through the course, and a final project. There will be an extra problem on the midterm exam for graduate students that will be optional/extra credit for undergraduates. Some homework assignments will also have an extra problem for graduate students. Graduate students will also be expected to do a more challenging project. Your lowest homework grade will be dropped. The grading breakdown will be approximately as follows:

Homework	35%
Midterm exam	25%
Class preparation and participation	15%
Final project	25%

Final project:
The final project will consist of:
1. an implementation of a significant parallel algorithm not included in a homework assignment
2. a report describing the background for the algorithm and design decisions for the implementation
3. a presentation during the final exam period describing and demonstrating your program

The specific algorithm can be of your choosing but you must have your topic pre-approved by the instructor. You may work individually on the final project or in teams of up to three people. In the case of group work, you must do clearly document the contributions of each team member and carry out the amount and difficulty of work proportional to the size of your team. For example, a team project might implement and compare a set of related algorithms, or compare various parallelization strategies for a given algorithm.

Make-up assignments and exam:
If you are unable to attend the midterm exam or to turn in a homework assignment on time due to a legitimate reason, such as a health problem or accident or pressing family matter, you will be allowed to make up the relevant exam or assignment.

Accommodations for Students with Disabilities

If you have a disability and need classroom accommodations, please contact The Center for Accommodations and Support Services (CASS) at 747-5148, or by email to cass@utep.edu, or visit their office located in UTEP Union East, Room 106. For additional information, please visit the CASS website at www.sa.utep.edu/cass.

Academic Honesty Policy

Make sure you understand the UTEP academic honesty policy. Students are encouraged to share ideas, but you must do your own homework and you must write your own code for the projects (you may copy code that is on the course website). If homework or program code is suspected of being duplicated or copied, you will receive an incomplete for the assignment, and your case will be referred to the Dean of Students for adjudication. If the instructor has reason to believe that you have cheated on a quiz or exam, your case will be referred to the Dean of Students for adjudication.