Course Number: 3460:477/577
Course Name: Introduction to Parallel Processing
Course Credits: 3
Schedule: Spring odd-numbered years

Syllabus Date: September 8 2003
Modification Date: September 13, 2005
Prepared By: Tim O’Neil, Kathy Liszka

Prerequisites:
Completion of 316 with a grade of C- or better and knowledge of C.

Text:

Bulletin Description:
Commercial processors past and present. Parallel languages, models of parallel computation, parallel algorithm design and performance evaluation. Parallel paradigms with relation to real world applications.

Detailed Description:
A broad introduction to basic parallel programming techniques using simple examples. Emphasis is placed on developing parallel algorithms.

Course Goals:
To write parallel programs on each of the two major parallel programming platforms (shared memory multiprocessors and message-passing networks of workstations) using different techniques in applicable situations. Along the way the student will examine problems which can be efficiently solved via parallel programming.

Topics:
1. The Need for Parallel Programming
2. Shared Memory Multiprocessor Systems
   a. Data Sharing
   b. Language Constructs for Parallelism using Pthreads
3. Message-Passing Computing
   a. Programming Options (Process Creation, Message-Passing Routines, etc.) using MPI and/or PVM
   b. Debugging and Evaluating Parallel Programs
4. Embarrassingly Parallel Computations
   a. Examples: Mandelbrot Set, Monte Carlo Methods
5. Partitioning and Divide-and-Conquer Strategies
   a. Examples: Bucket Sort, Numerical Integration, N-Body Problems
6. Pipelined Computations
   a. Computing Platforms for Pipelined Computations
b. Examples: Adding and Sorting Numbers, Prime Number Generation, Back-solving Upper-triangular Systems of Linear Equations

7. Synchronous Computations
   a. Implementing Barriers
   b. Local Synchronization
   c. Deadlock
   d. Data Parallel Computations and Synchronous Iteration
   e. Examples: Solving Linear Systems by Iteration, Heat Distribution Problems, Cellular Automata

8. Load Balancing and Termination Detection
   a. Dynamic Load Balancing
   b. Distributed Termination Detection Algorithms
   c. Example: Shortest Path Problems

9. Numerical Algorithms
   a. Matrix Addition
   b. Matrix and Matrix-Vector Multiplication
      i. Direct, Recursive, and Mesh Matrix Multiplication
   c. Relationship of Matrices to Linear Equations
      i. Solving a Linear System: Parallel Implementation of Gaussian Elimination
      ii. Iterative Methods of Solving Linear Systems: Jacobi iteration, fast convergence methods
   d. Finding Roots of Nonlinear Equations

**Computer Usage:**
Four or five parallel programs using C++ and either pthreads, MPI or PVM, executing on either a network of PCs or a cluster computer.

**References:**
Pacheco, Parallel Programming with MPI, Morgan Kaufmann, 1996.
Quinn, Parallel Programming in C with MPI and OpenMP, McGraw-Hill, 2003.