Course Syllabus  
CSCI 4576: High-Performance Scientific Computing

1. Course number and name  
CSCI 4576: High-Performance Scientific Computing

2. Credits and contact hours  
4 credits, 150 min lecture, 110 min lab, 5 hours office hours.

3. Instructor's or course coordinator's name  
Thomas Hauser

4. Text book, title, author, and year  

a. other supplemental materials
   a. Stefan Goedecker and Adolfy Hoisie, Performance Optimization of Numerically Intensive Codes, http://dx.doi.org/10.1137/1.9780898718218 (available through the CU-Boulder libraries)
   b. Pacheco, Parallel Programming with MPI
   d. Ian Foster, Designing and Building Parallel Programs http://www.mcs.anl.gov/~itf/dbpp/ (parallel program design)

5. Specific course information
   a. brief description of the content of the course (catalog description)
      Introduces computing systems, software, and methods used to solve large-scale problems in science and engineering. Students use high-performance workstations and a supercomputer. First course in a two-semester sequence.
   b. prerequisites or co-requisites
      Recommended CSCI 3656
   c. indicate whether a required, elective, or selected elective (as per Table 5-1) course in the program
      selected elective

6. Specific goals for the course
   a. Specific outcomes of instruction.
      The principal objective of this course is to introduce students to the use of high-performance computing systems in science and engineering. The course is designed both
for students whose main interest is in computer science and for students whose main interest is in some other field of science or engineering. The computer scientist should know how computers are used in practice so that the systems she or he studies, designs, and builds are useful, and so that he or she can work as part of a scientific team in an effective manner. The scientist or engineer who wants to use a computer as a tool ought to know about that tool, its strengths, its weaknesses, and how to use it effectively. Students will study algorithms and software designed to achieve good performance on today’s highly parallel supercomputing systems.

Problems of practical interest requiring parallel computing arise in such diverse areas of science as medicine, celestial mechanics, the study of fluid flows, the design of buildings and airplanes, and quantum mechanics among many others. The problems to be solved are computationally intensive and tend to require large amounts of data. We will see that data access patterns are critically important for achieving high speeds of computation.

A good understanding of algorithms for high-performance computing (not just scientific) requires a knowledge of numerical analysis. Without this knowledge you cannot understand why one algorithm might be more accurate than another or why one will produce a result of given accuracy in less time. Furthermore, and of great practical importance, you cannot be confident of the results of a computation without understanding the numerical methods used to produce them. Thus, we will study aspects of numerical analysis in the context of parallel computing.

Finally, we will study computer performance itself. We will study how to measure performance, how to characterize it in terms of performance parameters, and how to use tools to enhance performance.

b. explicitly indicate which of the student outcomes listed in Criterion 3 or any other outcomes are addressed by the course.

a) Apply Knowledge
b) Computing Requirements
c) Design System
d) Teamwork
g) Analyze Impacts
h) Professional Development
i) Current Techniques
j) Design Tradeoffs
k) Design & Development

7. Brief list of topics to be covered
Computer architecture, Single core optimization, Parallel computing architectures, Shared memory programming with OpenMP, Programming parallel computers using MPI, Hybrid programming with MPI and OpenMPI