# Syllabus for: Parallel Computing for Science and Engineering SDS 374C/394C, 56520/56630

**Victor Eijkhout and Charlie Dey**

**Spring 2016**

| | | |
|---|---|---|
| Time and place | FAC 101B | 12:30 – 2:00 TTh |
| Listed instructor: | Victor Eijkhout | eijkhout@tacc.utexas.edu, ROC 1.425, 471-5809 |
| Co-instructor: | Charlie Dey | charlie@tacc.utexas.edu, ROC 1.114, 232-7591 |
| Office Hours: | | after class<br>or by appointment |
| TA: | Harshal Priyadarshi | harshal@cs.utexas.edu, hours TBD |

## 1 Checklist

1. First day of class: get yourself a TACC account and mail your usename to the instructors; section 7.
2. First week of class: download a copy of the two textbooks; section 5. If you are not familiar with Linux/Unix, do the tutorial in the IHPSC textbook.
3. First week of class: sign up for Piazza; section 7.2.
4. Second week of class: sign up for an account at `bitbucket.org` (preferred) or `github.com`; section 7.3.
5. Second week of class: if you are unfamiliar with LaTeX, do the tutorial in the IHPSC textbook.
6. Third week of class: if you are unfamiliar with `make`, do the tutorial in the IHPSC textbook.
7. End of March: think of a final project, get approval from the instructor(s).

## 2 Rationale; Course Aims and Objectives

This course is intended to teach the fundamentals of parallel programming for scientific and engineering applications. It includes the study of parallel computing principles, architectures, and technologies, and parallel application development, performance, and scalability. The course builds on existing programming courses: prior programming experience using C/Fortran on Unix/Linux systems is required.

Parallel computing has become the dominant method for achieving high performance in computational science, and engineering research and development. Computer systems containing from 10 to 100,000 CPUS are based on commodity processors from Intel, AMD, and IBM and are now commonplace in academia, government, and industry. The latest generation of microprocessors exposes an even greater opportunity for parallelism through technologies such as Hyper-Threading, multi-core and vector processors. Hence, parallel computing has become an important component of university curricula for scientists and engineers. Parallel Computing for Science and Engineering (PSCE) combines a structured approach, a broad view of parallel methods, and practical exercises to create a basic foundation in parallel computing.

The aim of this course is for the students to acquire the ability to develop high performance parallel applications that are scalable and can run efficiently on platforms ranging from desktop systems to supercomputers.

We begin with a comprehensive introduction to parallel computing theory, and discuss basic hardware features and software components (tools, utilities, etc.) that are essential for parallel computing. Next, the OpenMP and MPI programming paradigms are introduced. These sections prepare students for a discussion of key algorithms and examples of applications in various fields. A focus on application development, performance, and scalability will be maintained throughout all sections. The scope of this course is to prepare students to formulate and develop parallel algorithms and to implement them as effective and efficient applications for parallel computing systems.

## 3 Format and Procedures

The course is structured around lectures, in-class labs, assignments (both theory and programming), quizzes, and one course project.

### 3.1 Tentative Course Schedule

The following topics will be taught, though not necessarily strictly in this sequence:

1. Overview of Parallel Computing. Parallel computing concepts, essentials of parallel computer architectures and hardware, and standard programming models for parallel computers. Account setup, hands-on introduction to large-scale system environments for development and execution of parallel applications.
2. Shared Memory Parallel Programming with OpenMP. OpenMP directives, syntax, and operation; parallelize serial codes with OpenMP; control and synchronization; performance issues, scalability, and new features of OpenMP.
3. Distributed Memory Programming with the Message Passing Interface (MPI) library. MPI communication operations and data structures, syntax and features; parallelizing serial codes with MPI, performance issues, future of MPI.

4. Parallel Algorithm Design, Optimization, Hardware, and Scientific Applications Case Studies. The basics elements of parallel algorithm design, optimization and hardware operations for parallel programming will be presented.

### 3.2 Programming Projects and Topics

There are many practical areas of Parallel Computing to explore, ranging hardware to software. These include topics such as interprocessor communication, data-intensive computing, shared memory, vectorization, and threading in multi-core CPU systems and many-cores accelerator systems (GPU, ARM and MIC). Select, propose, or use your present research project for the basis of a computational project that explores, tests or develops a parallel algorithm, application, or concepts. This is a wide open adventure that may involve hardware, standard programming languages, parallel optimization, parallel I/O, etc. A portfolio of subjects and potential ideas will help jump-start everybody, so that students can be working on a project by the 4th Section.

Requirements: Must be computational (not just academic theory), use parallel techniques/hardware, relevant to High Performance Computing. Students are expected to turn in code as well as an approximately 5 page report.

## 4 Assumptions and prerequisites

- Students must have prior programming experience using either C/C++ or Fortran90 equivalent to the SDS course 222/292. Not sufficient are: Java, Matlab, Python.
- Students need to be familiar with the Linux/Unix operating system.
- Familiarity with matrix linear algebra is necessary, for instance 408D or 408M; some familiarity with calculus and multi-variate calculus is desirable.
- Experience with developing scientific codes is helpful but not required. Each student is expected to employ one or more of the parallel concepts and develop a course project. (Suggestions and guidance will be provided by the instructors and TA.)

## 5 Course Readings/Materials

The lecture content is derived from many different sources, including hardware manuals, journal articles and conference presentations and papers. The suggested texts are a good introduction to the material; however, the following references should be consulted for an in-depth understanding of certain subjects.

**Lecture slides and such** All lecture slides will be distributed through the repository https://bitbucket.org/VictorEijkhout/pcse2016. If you sign up for a BitBucket account (section 7.3) you can follow this repository and be notified of uploads.

**Reference works** The MPI standard is online: http://www.mpi-forum.org/docs/; index of all routines: http://www.mcs.anl.gov/research/projects/mpi/www/www3/. The OpenMP standard is likewise electronically available: http://openmp.org/wp/openmp-specifications/. It is a good idea to bookmark these pages.

**Required download** **(1)**. *Introduction to High-Performance Scientific Computing*, Victor Eijkhout, contains much background material on computer architectures and computing. The class will regularly refer to this book. It also contains tutorials that will be used in class. http://www.tacc.utexas.edu/~eijkhout/istc/istc.html. **(2)** *Parallel Computing for Science and Engineering*, Victor Eijkhout, is an unfinished textbook about MPI and OpenMP. Same link as the previous book. *These books are by the class instructor; he greatly appreciates receiving comments and corrections on this text.*

**Other tutorials** There are many books about parallel programming with MPI and/or OpenMP. The instructors recommend the following books for detailed information specific to MPI and OpenMP:

- Using MPI-2 / Using Advanced MPI. William Gropp, Ewing Lusk, Anthony Skjellum, The MIT Press, Cambridge, Massachusetts, 1999. *A very detailed book by some of the original authors of MPI. If you buy a used copy, make sure it's not the MPI-1 book.*
- Parallel Programming in OpenMP, Rohit Chandra, Leo Dagum, Dave Kohr, Dror Maydan, Jeff McDonald, Ramesh Menon, Academic Press Morgan Kaufmann Publishers, San Diego, CA, 2001.

More general texts sometimes do not discuss every feature of MPI and OpenMP, but they may be enlightening in providing a different view of the material. We recommend:

- Parallel Programming in C with MPI and OpenMP, Michael J. Quinn, McGraw-Hill, 2003. *No longer in print, but used copies can be found cheaply through services like bookfinder.com.*
- Designing and Building Parallel Programs, Ian Foster, http://www-unix.mcs.anl.gov/dbpp/ *This book has many general discussions about aspects of parallel computing. It predates MPI-2, so do not rely on its discussion of MPI. OpenMP is not mentioned.*
- High Performance Computing, 2nd Edition (Risc Architectures, Optimization & Benchmarks), Charles Severance and Kevin Dowd, Oeilly & Associates, Inc., Sebastopol, CA, 1998. Available under Creative Commons: http://cnx.org/

content/col11136/1.5/content_info. *This book is specifically about getting high performance out of your code, both sequentially and in parallel.*

- Practical MPI Programming , http://www.redbooks.ibm.com/redbooks/pdfs/sg245380.pdf *This is a very practical guide to parallel programing with lots of code examples and discussion of strategies.*

## 6 Assignments, Assessment, and Evaluation

There will be quizes on (1) OpenMP, (2) MPI, (3) other topics; instead of a final exam there will a final project for which a project report will be due the last day of class. There may also be pop quizes. There will be regular programming assignments.

Grades will be based on quizzes (35%), assignments (35%) the project (30%).

## 7 Electronic resources

**Computing Resources**

Students are free to use personal computers and laptops (or machines belonging to any department they have access to) for developing codes. In that case they need a C/C++ or Fortran compiler with OpenMP and an MPI installation.

However, assignments will ultimately have to be performed on TACC's Stampede cluster.

If you don't already have a TACC account, go to https://portal.tacc.utexas.edu/ and create an account there. Send mail to the instructor(s) what login name is you have created. if you already have a TACC account, for instance through working with an advisor, mail us that.

Students can use their own PCs/Macs/workstations to access the machines through SSH. Free SSH clients for PCs and Macs are available through BevoWare (SSH Secure Shell 3.2.9) and/or Putty. Later in the course, an X display manager will be needed. On windows, the Cygwin (Unix with X11 includes an X display manager) environment (http://www.cygwin.com/) and/or Xming (http://openmp.org/wp/openmp-specifications/) provide a free X display manager. On Macs and Unix workstations, use the X11 support. Alternatively, students can use a virtualization layer, such as vmware, to install and run a Linux guest operating system to use to login to TACC clusters. This would not interfere with the host (original) operating system and would run as a program within Windows/Mac.

### 7.1 Use of Canvas

In this class we will use Canvas, a Web-based course management system with password-protected access at http://courses.utexas.edu to distribute course materials and the Power Point slides used each day, to communicate and collaborate online, to post grades, to submit assignments, and surveys. The lecture slides will be posted before each class. You can find support in using Canvas at the ITS Help Desk at 475-9400, Monday through Friday, 8 a.m. to 6 p.m.

### 7.2 Piazza for discussions

Students can post discussion or questions and suggest answers on Piazza: go to https://piazza.com and enroll with your utexas email. Instructors will make an effort to check the forum at least once a day. Ask your questions here before mailing the instructor or TA!

### 7.3 Bitbucket code repositories

Programming exercises are to be submitted for grading through a source code repository. Sign up **with your utexas.edu email** at https://bitbucket.org/ for a free account and invite both instructors and the TA to your project. The use of source code repositories will be explained in class.

The use of gibhub.com is acceptable too.

Bitbucket is also used to distribute class materials; section 5.

## 8 Formal and informal policies

**Class attendance and participation policy**

We expect students to attend and participate in class in accordance with the UT Honor Code (see below). Students are encouraged to ask questions, especially relating to material used in their projects.

This class will have lecture and lab sessions. During the lectures there are to be no laptops open, as these are distracting to the student, as well as others who can see the screen. Tablets with non-raised screens are allowed.

**Academic Integrity**

University of Texas Honor Code
The core values of The University of Texas at Austin are learning, discovery, freedom, leadership, individual opportunity, and responsibility. Each member of the university is expected to uphold these values through integrity, honesty, trust, fairness, and respect toward peers and community.

Each student in this course is expected to abide by the University of Texas Honor Code. Any work submitted by a student in this course for academic credit will be the student's own work. Collaborations will be allowed for the course project.

You are encouraged to study together and to discuss information and concepts covered in lecture and the sections with other students. You can give ”consulting” help to or receive ”consulting” help from such students. However, this permissible cooperation should never involve one student having possession of a copy of all or part of work done by someone else, in the form of an e-mail, an e-mail attachment file, a diskette, or a hard copy.

Should copying occur, both the student who copied work from another student and the student who gave material to be copied will both automatically receive a zero for the assignment. Penalty for violation of this Code can also be extended to include failure of the course and University disciplinary action.

During examinations, you must do your own work. Talking or discussion is not permitted during the examinations, nor may you compare papers, copy from others, or collaborate in any way. Any collaborative behavior during the examinations will result in failure of the exam, and may lead to failure of the course and University disciplinary action.

**Religious Holy Days**

By UT Austin policy, you must notify the instructors of your pending absence at least fourteen days prior to the date of observance of a religious holy day. If you must miss a class, an examination, a work assignment, or a project in order to observe a religious holy day, they will give you an opportunity to complete the missed work within a reasonable time after the absence.

**Use of E-mail for Official Correspondence to Students**

All students should become familiar with the University’s official e-mail student notification policy. It is the student’s responsibility to keep the University informed as to changes in his or her e-mail address. Students are expected to check e-mail on a frequent and regular basis in order to stay current with University-related communications, recognizing that certain communications may be time-critical. It is recommended that e-mail be checked daily, but at a minimum, twice per week. The complete text of this policy and instructions for updating your e-mail address are available at http://www.utexas.edu/its/help/utmail/1564.

**Documented Disability Statement**

Any student with a documented disability who requires academic accommodations should contact Services for Students with Disabilities (SSD) at (512) 471-6259 (voice) or 1-866-329-3986 (video phone). Faculty are not required to provide accommodations without an official accommodation letter from SSD. (Note to Faculty: Details of a student disability are confidential. Faculty should not ask questions related to a student condition or diagnosis when receiving an official accommodation letter.)

Please notify me as quickly as possible if the material being presented in class is not accessible (e.g., instructional videos need captioning, course packets are not readable for proper alternative text conversion, etc.).

Please notify me as early in the semester as possible if disability-related accommodations for field trips are required. Advanced notice will permit the arrangement of accommodations on the given day (e.g., transportation, site accessibility, etc.).

Contact Services for Students with Disabilities at 471-6259 (voice) or 1-866-329-3986 (video phone) or reference the SSD website for more disability-related information: http://www.utexas.edu/diversity/ddce/ssd/for_cstudents.php

**Behavior Concerns Advice Line (BCAL)**

If you are worried about someone who is acting differently, you may use the Behavior Concerns Advice Line to discuss by phone your concerns about another individual's behavior. This service is provided through a partnership among the Office of the Dean of Students, the Counseling and Mental Health Center (CMHC), the Employee Assistance Program (EAP), and The University of Texas Police Department (UTPD). Call 512-232-5050 or visit http://www.utexas.edu/safety/bcal.

**Q drop Policy**

The State of Texas has enacted a law that limits the number of course drops for academic reasons to six (6). As stated in Senate Bill 1231: Beginning with the fall 2007 academic term, an institution of higher education may not permit an undergraduate student a total of more than six dropped courses, including any course a transfer student has dropped at another institution of higher education, unless the student shows good cause for dropping more than that number.

**Emergency Evacuation Policy**

Occupants of buildings on the UT Austin campus are required to evacuate and assemble outside when a fire alarm is activated or an announcement is made. Please be aware of the following policies regarding evacuation:

- Familiarize yourself with all exit doors of the classroom and the building. Remember that the nearest exit door may not be the one you used when you entered the building.
- If you require assistance to evacuate, inform me in writing during the first week of class.
- In the event of an evacuation, follow my instructions or those of class instructors.
- Do not re-enter a building unless youe given instructions by the Austin Fire Department, the UT Austin Police Department, or the Fire Prevention Services office.