

# CS 594 – 004

## Scientific Computing for Engineers

---

Web page for the course:

<http://www.cs.utk.edu/~dongarra/WEB-PAGES/cs594-2007.htm>

CS 594 – 004  
Wednesday's 1:30 – 4:30

---

- ◆ **Scientific Computing for Engineers**
- ◆ **Spring 2007 - 3 credits**
  - Jack Dongarra
  - with help from:
    - » George Bosilca
    - » Shirley Moore
    - » Stan Tomov
- ◆ **Class will meet in Room C211, Claxton Building**

## To Get Hold of Us

---

- ◆ **Email:** [dongarra@cs.utk.edu](mailto:dongarra@cs.utk.edu)
  - **Room:** 413, Claxton
  - **Phone:** 974-8295
- ◆ **Office hours:**
  - **Wednesday 11:00 - 1:00, or by appointment**
- ◆ **TA:** Erika Fuente [efuentes@cs.utk.edu](mailto:efuentes@cs.utk.edu)
- ◆ **350 Claxton Complex, 974-7622**
  - **OH:** 10am-11am TR, or by request

3

## Four Major Aspects Of The Course:

---

1. Start with current trends in high-end computing systems and environments, and continue with a practical short description on parallel programming with MPI, OpenMP, and pthreads.
2. Illustrate the modeling of problems from physics and engineering in terms of partial differential equations (PDEs), and their numerical discretization using finite difference, finite element, and spectral approximation.
3. Deal with solvers: both iterative for the solution of sparse problems of part II, and direct for dense matrix problems. Algorithmic and practical implementation aspects will be covered.
4. Various software tools will be surveyed and used. This will include PETSc, Sca/LAPACK, MATLAB, and some tools and techniques for scientific debugging and performance analysis.

4

## Grades Based on:

---

- ◆ 30% on weekly homework  
(the lowest homework grade will be dropped)
- ◆ 30% on a written report and presentation  
(20 pages circa.)
- ◆ 30% on a final exam (2 hours)
- ◆ 10% on class participation.

5

## Homework

---

- ◆ Usually weekly
- ◆ Lowest grade will be dropped
- ◆ Must be turned in on time (no late homework)
- ◆ Don't copy someone else's homework.
- ◆ Sometimes problems, sometimes programming assignment, sometimes requiring running a program to find the solution.

6

## Homework (continued)

---

- ◆ We expect an analysis and detailed discussion of the results of your efforts.
  - The program itself is not very interesting.
- ◆ Programming in C or Fortran.
- ◆ Will go over the assignments the week they are due.
- ◆ See class web page weekly for details.

7

## Using the SInRG Clusters

---

- ◆ **Boba Cluster**
  - 32 Dell Precision 530s
  - Dual Pentium IV Xeon 2.4 GHz Processors
  - 512 KB Cache
  - 2 GB Ram
  - 2 73GB MAXTOR 6L080J4 Disk Drives
  - On board 3Com Corporation 3c905C NIC
  - Intel e1000 100/1000 NIC
- ◆ **Frodo Cluster**
  - 65 dual AMD-Opteron 240 nodes
  - 2 GB RAM per node
  - Myrinet 2000 interconnect
- ◆ **Neo Cluster**
  - 16 Dual 450MHz UltraSPARC-II 64-bit RISC processors with 4MB L2 cache
  - 512MB (4x128) ECC SDRAM DIMM memory installed
  - SCSI: Integrated dual 40 MB/sec.UltraSCSI channels
  - 27.36B (1-18GB, 1-9GB) UltraSCSI 10,000 RPM Hard Drives
  - TP Ethernet 10/100BASE-T†
  - SysKonnnect Gigabit Ethernet

8

## ICL Machines

---

- ◆ 16 node Intel P4 cluster Connected with Mellanox and Myrinet 10G
- ◆ 24 node AMD Opteron cluster connected with Silverstorm
- ◆ 64 node Intel EM64T cluster connected with Myrinet 2000
- ◆ 64 node AMD Opteron cluster connected with Myrinet 2000
- ◆ Intel Itanium cluster

9

## Project

---

- ◆ Topic of general interest to the course.
- ◆ The idea is to read three or four papers from the literature (references will be provided)
- ◆ Implement the application on the cluster you build
- ◆ Synthesize them in terms of a report (~20 pages)
- ◆ Present your report to class (~30 mins)
- ◆ New ideas and extensions are welcome, as well as implementation prototype if needed.

10

## Remarks

---

- ◆ **Hope for very interactive course**
- ◆ **Willing to accept suggestions for changes in content and/or form**

11

## Final Exam

---

- ◆ **In class**
- ◆ **Will cover the material presented in the course**
- ◆ **~2 hours**

12

## Material

---



- ◆ **Book:**

- **The Sourcebook of Parallel Computing**, Edited by Jack Dongarra, Ian Foster, Geoffrey Fox, William Gropp, Ken Kennedy, Linda Torczon, Andy White, 2002, 760 pages, ISBN 1-55860-871-0, Morgan Kaufmann Publishers.

- ◆ For each lecture a set of slides will be made available in pdf or html.

- ◆ Other reading material will be made available electronically if possible.

- ◆ The web site for the course is:

- <http://www.cs.utk.edu/~dongarra/WEB-PAGES/cs594-2007.htm>

13

## Other Sources

---

- ◆ Will use material from the internet (manuals, papers)
- ◆ Will use a variety of book sources; including
  - **Ian Foster**
    - » Designing and Building Parallel Programs
  - **Alices E Koniges**
    - » Industrial Strength Parallel Computing
  - **Jack Dongarra, Iain Duff, Danny Sorensen, Henk van der Vorst**
    - » Numerical Linear Algebra for High Performance Computers
  - **Ananth Gramma et al.**
    - » Introduction to Parallel Computing
  - **Michael Quinn**
    - » Parallel Programming
  - **David E. Culler & Jaswinder Pal Singh**
    - » Parallel Computer Architecture
  - **George Almasi and Allan Gottlieb**
    - » Highly Parallel Computing

14

## Important Place for Software

---

- ◆ **Netlib - software repository**
  - Go to <http://www.netlib.org/>

15

## What will we be doing?

---

- ◆ **Learning about:**
  - **High-Performance Computing.**
  - **Parallel Computing**
  - **Performance Analysis**
  - **Computational techniques**
  - **Tools to aid parallel computing.**
  - **Developing programs using PVM, MPI, HPF, and perhaps OpenMP.**

16

## Outline of the Course

---

1. January 10 Introduction to High Performance Computing
2. January 17 Message Passing
3. January 24 Message Passing continued
4. January 31 HPC Architectures and Parallel Programming
5. February 7 Clusters part 1
6. February 14 Clusters part 2
7. February 21 Projection and Its Importance in Scientific Computing
8. February 28 Discretization of PDEs
9. March 7 Memory Hierarchy and Cache, Dense Matrix Ops
- March 14 Spring Break
10. March 21 Sparse Matrices
11. March 28 Dense Linear Algebra part 1
11. April 4 Dense Linear Algebra part 2
12. April 11 Iterative Methods in Linear Algebra part 1
13. April 18 Iterative Methods in Linear Algebra part 2
14. April 25 Performance Analysis Tools
15. May 2 Class Final Reports

17

## What you should get out of the course

---

**In depth understanding of:**

- ◆ **When is parallel computing useful?**
- ◆ **Understanding of parallel computing hardware options.**
- ◆ **Overview of programming models (software) and tools.**
- ◆ **Some important parallel applications and the algorithms**
- ◆ **Performance analysis and tuning**

18

## Background

---

- ◆ C and/or Fortran programming
- ◆ Knowledge of parallel programming
- ◆ Some background in numerical computing.

19

## Computer Accounts

---

- ◆ For much of the class computing you can use one of our set of computer clusters. More on this later
- ◆ If you have an account in the Department you have access to the TORC cluster: torc1 through torc8.
- ◆ Cluster of PC's:
  - <http://icl.cs.utk.edu/iclhelp/> look under clusters

20

## CS 594 - Scientific Computing for Engineers

Homework #1  
January 10, 2007  
Due: January 24, 2007

I would like you to implement a version of the following mathematical operations:

- the 2-norm of a vector,

$$\|x\|_2 = \sqrt{x^T x} = \sqrt{\sum_{i=1}^n x_i^2}$$

- matrix - vector multiplication,

$$y = y + A * x$$
$$y_i = y_i + \sum_{j=1}^n A_{ij} * x_j, \text{ for } i = 1, \dots, n$$

- matrix multiplication

$$C = C + A * B$$
$$C_{ij} = C_{ij} + \sum_{k=1}^n A_{ik} * B_{kj}, \text{ for } i, j = 1, \dots, n$$

The point of this assignment is not to write software, but to look at the performance for each of your implementations and try to explain why you are getting the performance you see and what you could do to increase the performance. You should produce a software implementation for each and run some experiments on various systems, in particular use processors from bobo, fredo, and neo clusters. I would like to see a report and analysis of your results, perhaps some plots of your performance data for  $n$  between say 10 and 1000. Please verify and convince me that you are computing the correct results in each case. Let me know what computers you used and how you are getting the performance results as well.

Our TA, Erika Fuentes [efuentes@cs.utk.edu](mailto:efuentes@cs.utk.edu), will have a set of timer you can use to measure the execution time of your programs.

You can find out information on various processors at:  
<http://www.cpu-world.com/CPU/index.html>  
<http://www.cpu-world.com/sspec/index.html>

