BASICS OF PARALLEL AND DISTRIBUTED COMPUTING
CIS 4930.001 AND CIS 6930.007
SPRING 2011
MONDAY 2-4:45PM, ENC 1002

Instructor:	Anda Iamnitchi	Office:	ENB 334
Email:	anda@cse.usf.edu	Phone:	974-5357
Office Hrs:	Wed 10am-12pm or by appointment	Fax:	974-5456

TAs:	Colin Arnold	Josh Finnis
Email:	cdarnold@mail.usf.edu	jfinnis@mail.usf.edu
Office Hrs:	MTR 9:30-10:30am	R 9:30-10:30am, W 2-3pm
Office:	ENB 325	

COURSE OVERVIEW

This course introduces the basics of parallel and distributed computing with an experimental focus on online social network analysis. The analysis of online social networks has been chosen as the target class of applications for applying the knowledge introduced in lectures because of several factors: it is an active research topic in computer science; the scale challenges of very large social networks require parallel computing approaches; and parallelizing graph algorithms is not trivial.

The course is project-oriented. Class time will be spent on a combination of lectures on topics in parallel programming and models, in-class discussions of research papers, mini-lectures on Social Network Analysis (SNA), brainstorming sessions on parallelizing some SNA-relevant algorithms, and interdisciplinary team project meetings. There are no tests. Final evaluation will be based on class participation (20%), programming projects (40%), and final project and presentation (40%). Lectures, readings and class discussions will introduce the design of parallel programs, performance metrics, distributed systems architectures, time and global state, coordination and agreement, distributed transactions and web services. Programming projects will introduce the practice of parallel and distributed programming with threads, MPI, Google's MapReduce framework, and Amazon Web Services. The final project will be an interdisciplinary team project that involves graduate students enrolled in the *Social Network Analysis* course (SYA 6933.904) taught by Professor Skvoretz from the Department of Sociology.

The course is intended for advanced undergraduate students and graduate students in CSE.

COURSE OBJECTIVES

- To learn how to design parallel programs and how to evaluate their execution
- To understand the characteristics, the benefits and the limitations of parallel systems and distributed infrastructures
- To expose students to writing code in different parallel programming environments
- Build experience with interdisciplinary teamwork

LEARNING OUTCOMES

- Be able to reason about ways to parallelize a problem
- Be able to evaluate a parallel platform for a given problem
- Become familiar with programming with MPI and MapReduce/Hadoop
- Know general concepts on Cloud computing, grid computing, and peer-to-peer systems
- Become familiar with evaluation of online social networks and their potential
- Become familiar with interdisciplinary teamwork

PREREQUISITES

All students registered for this class should be comfortable with programming in general and C and/or Java in particular. In addition, the following prerequisites should be met:

- Undergraduate students: COP 4600 (Operating Systems)
- Graduate students in CS: undergraduate Operating System
- Graduate students outside CS: instructor's permission (email anda@cse.usf.edu)

RECOMMENDED TEXTBOOK

Patterns for Parallel Programming, Timothy G. Mattson, Beverly A. Sanders, Berna L. Massingill. Addison-Wesley Professional; 1 edition (September 25, 2004). ISBN: 0321228111 [MSM]

Parallel Programming: Techniques and Applications Using Networked Workstations and Parallel Computers, by Barry Wilkinson, Michael Allen. Prentice Hall; 2nd edition (March 14, 2004). ISBN: 0131405632 [WA]

Principles of Parallel Programming, by Calvin Lin and Larry Snyder, Addison-Wesley; 1st edition (March 7, 2008). ISBN: 0321487907 [LS]

There is no required textbook for this class.

WORKLOAD AND GRADING

IN-CLASS PARTICIPATION (20%) will include discussions on reading, problem solving, and short class presentations.

MINI PROJECTS (40%): There will be at least 3 mini projects, one involving MPI and two involving the Hadoop/MapReduce framework. A warm-up mini project with threads is considered, as well as a project using the Amazon cloud infrastructure, if time permits.

FINAL PROJECT (40%) Final projects will be interdisciplinary team projects involving (most likely) two students in this course and one student registered in the *Social Network Analysis* course (SYA 6933.904) taught by Professor Skvoretz. In general, SNA students will have the responsibility to situate the project in the social network literature and identify relevant datasets and network metrics to compute. Designing, implementing the algorithms to execute the necessary analysis, and collecting the results will be the primary responsibility of CS students. The team will have collective responsibility for the interpretation and write-up of results.

Deliverables include:

- **PROJECT PROPOSAL (5%)**. Brief (1000 words) proposal for the team project, due 21 February 2011. Members of each team will receive instructor feedback in the following week's class on 28 February 2011.
- **MIDTERM REPORT (10%).** 4-page report due 28 March 2011 that details the computer science aspect of the project, including choice of and motivation for the parallel platform to use, dataset description and initial analysis (or code to collect it and preliminary data collected), detailed plan of work, and preliminary results. The purpose of this component is to make sure you start early enough to be able to finish by the end of the semester with meaningful results and result interpretation. This component is the responsibility of the CS team members.
- **FINAL REPORT AND PRESENTATION (25%).** The last two weeks of class meetings will be devoted to presentations of team projects. The final report of the project is expected to be of publication quality and is due Monday 2 May 2011. It is expected that the final report will address the feedback received during project presentation.

For final letter grades, I will use the standard scale of A (100-90), B (89-80), C (79-70), D (69-60), and F (59-0). I might also use pluses and minuses on final grades to indicate either a borderline grade or exceptionally outstanding work (A+).

POLICIES

ATTENDANCE: Since much of the lecture material is collected and compiled from multiple sources and the class meetings are heavily interactive (including paper discussions, in-class problem solving and programming), class attendance is mandatory. If you have a legitimate excuse for an absence, it is important that you provide the supporting documentation.

ACADEMIC INTEGRITY: Everything you turn in for this class must be your own work. If you are caught cheating, you will receive an FF grade for the class. Material copied from the Web and submitted as your work *is cheating*. Please see the University's Undergraduate Catalog regarding these policies at http://www.ugs.usf.edu or the Graduate Catalog's section on Academic Policies at http://www.grad.usf.edu/catalog.asp.

ACCOMMODATIONS: Students in need of academic accommodations for a disability may consult with the office of Students with Disabilities Services to arrange appropriate accommodations. Students are required to give reasonable notice prior to requesting an accommodation.

TENTATIVE SCHEDULE

Week 1:	The power and potential of parallelism; Goals.	[LS 1, MSM 1.1-1.2]
	Parallel architectures: a brief introduction	[LS 2, MSM 2.2]
Week 2:	*Holiday*	
Week 3:	Reasoning about performance of parallel programs	[LS 3, 11, MSM 2.4, 2.5]
Week 4:	Basics of MPI	
Week 5:	Meeting with the SNA class. Basics of SNAs; basics of parallel computation; reading discussions; final project rules and ideas.	
Week 6:	Data and task parallelism.	[LS 4]

Week 7:	Independent parallelism; assigning work to processes	[LS 5]
Week 8:	Grid and cloud computing	
Week 9:	MapReduce and Hadoop	
Week 10:	*Spring Break*	
Week 11:	Basics of distributed systems	
Week 12:	Basics of distributed systems	
Week 13:	Peer-to-peer systems	
Week 14:	Other distributed systems	
Week 15:	Final project presentations	
Week 16:	Final project presentations	

READING LIST (TO BE FINALIZED)

1. Implementing and evaluating multithreaded triad census algorithms on the Cray XMT. George Chin, Andres Marquez, Sutanay Choudhury, and Kristyn Maschhoff. In *Proceedings of the 2009 IEEE International Symposium on Parallel & Distributed Processing* (IPDPS '09).
2. Measuring social dynamics in a massive multiplayer online game. Michael Szell, Stefan Thurner. *Social Networks*, Vol. 32, No. 4. (October 2010), pp. 313-329.
3. A view of cloud computing. Michael Armbrust, Armando Fox, Rean Griffith, Anthony D. Joseph, Randy Katz, Andy Konwinski, Gunho Lee, David Patterson, Ariel Rabkin, Ion Stoica, and Matei Zaharia. 2010. *Commun. ACM* 53, 4 (April 2010), 50-58.
4. The Eucalyptus Open-Source Cloud-Computing System. Daniel Nurmi, Rich Wolski, Chris Grzegorczyk, Graziano Obertelli, Sunil Soman, Lamia Youseff, and Dmitrii Zagorodnov. 2009. In *Proceedings of the 2009 9th IEEE/ACM International Symposium on Cluster Computing and the Grid* (CCGRID '09). IEEE Computer Society, Washington, DC, USA, 124-131.
5. A test of structural features of Granovetter's strength of weak ties theory, N. Friedkin. *Social Networks*, Vol. 2, No. 4. (1980), pp. 411-422.
6. Structure and tie strengths in mobile communication networks. J. P. Onnela, J. Saramäki, J. Hyvönen, G. Szabó, D. Lazer, K. Kaski, J. Kertész, A. L. Barabási. Proceedings of the National Academy of Sciences, Vol. 104, No. 18. (1 May 2007), pp. 7332-7336.

Every part of this syllabus is subject to adjustment as the semester progresses. Please contact me as soon as possible if you have particular interests in material that is relevant to the class topic but not covered in enough detail; I will be happy to accommodate reasonable requests for modifications.