GE523: Discrete Event Dynamic Systems
Spring, 2013

Instructor: Prof. R.S. Sreenivas
155 Coordinated Science Laboratory (Primary Office)
110 Transportation Building (Secondary Office)
e-mail: rsree@illinois.edu
TR, 2:00-3:20PM, 203 Transportation Building

Course Description: Traditional system theory concerns systems with continuous-time or discrete-time variables that can be modeled by difference or differential equations, possibly including random or non-deterministic elements. Modern technology in the form of computers, manufacturing processes, communication networks, intelligent vehicle/highway systems (IVHS), etc., has forced upon us event-driven dynamics (or information-driven dynamics): systems in which the state changes only at discrete points in time in response to the occurrence of particular events. There is a growing need for the study of systems whose states have logical or symbolic, rather than numerical, values that change with the occurrence of events. We call such systems *Discrete Event Dynamic Systems* (DEDS). This course introduces the various issues in modeling, analysis, control, and performance evaluation of DEDS. The main objective is to present, in a consolidated form, research in the theory of DEDS over the last decade.

The secondary objective of this course is to introduce you to a variety of computational techniques and tools against the backdrop of DEDS. You will become proficient in C/C++ by the end of this course.

Text: Course Notes written by the Instructor.

1 Tentative Syllabus

"You must fill your heads with wisdom before you can break boards with it."
−− *Karate instructor on "The Simpsons"*.

Lectures (Exact coverage will depend on Class' skills and needs).

1. **First Day of Class**: Administrivia
   (a) Compiler
      i. Windows users – *Microsoft Visual Studio 2010* from the UI Webstore (Free).
      ii. Mac users – *Xcode* (version 3.2.6 or higher; current version 4.5.2) (Free).
   (b) Software Packages
      i. Install the MILP solver lp_solve (see handout).
      ii. Install the NEWMAT Matrices/Linear-Algebra package (see handout).

(c) *Introduction to DEDS via examples.*

2. **Overview of C++**.

   (a) **Lesson 1**: (Lightning) Review of the basics (of C++); Recursion (in C++); Tower of Hanoi Problem; Backtracking via Recursion; Asymptotic Analysis of Computation-time ($O(\bullet), \Theta(\bullet)$ and $\Omega(\bullet)$ notation); The Master Theorem; Karatsuba's $O(n^{1.585})$ algorithm for multiplying two n bit numbers; Strassen's $O(n^{2.81})$ algorithm for multiplying two $n \times n$ matrices.

3. **Methods/Tools from Combinatorics**

   (a) **Lesson 2**: Generating Functions and Recurrence Relations; Inclusion and Exclusion Principle; Polya Theory; Schur Functions; Matching Theory; Designs; Ramsey Theory; Counting the number of Lattice-Points in bounded polytopes; (C++ Computation) Illustrations.

4. **Methods/Tools from Combinatorial Optimization**

   (a) **Lesson 3**: (Integer) Linear Programs; (C++ code) for LP/ILP models using the Lpsolve API; (C++ code to solve) Some problems from Graph Theory.

5. **Methods/Tools from Computation Theory**

   (a) **Lesson 4**: *Turing Machines*, *Halting Problem* and *Decidability*; Approximation Algorithms, Non-approximability (*Gap* Theorem); (C++ Code for) Approximate solutions using *Duality*; *Propositional Logic*, *First-order Logic*, *Second-order Logic*, *Soundness*, *Completeness*; *(Linear) Temporal Logic* (LTL), Verification, *Büchi* Automata; (Basics of) LTL model checking using SPIN; *Petri Nets* (PNs), (C++ code for) a variety of PN problems; *Max-Plus* (*Min-Plus*) Algebras.

6. **Methods/Tools from Statistics and Simulation**

   (a) **Lesson 5**: (C++ code for) Pseudo Random number generation and Uniform Distributions; (C++ code for) Inverse Transform Technique for other distributions; Efficient Discrete Random Variate generation; (C++ code for) Geometric Distributions; (C++ code for) Binomial Random Variates; The *Box-Muller Transform* and (C++ code for) Unit-Normal Variate generation; (C++ code for) Multivariate Gaussian; Generating constrained Random Variates; Generating Random Variates from Empirical Data; Basics of Discrete-time Markov Chains and some related computational problems (in C++); *Infinitesimal Perturbation Analysis* via Examples; Generalized Semi-Markov Processes (GSMP); *Urn Models* and their application to DEDS.

7. **Qualitative Control of DEDS**:

(a) **Lesson 6**: The *"Forbidden State Problem"* and the *"Forbidden String Problem"*; Distributed Supervisory Control and the Normality Theorem; Complexity of the Sequential Supervisory Control Problem; Decidability (Computability) issues in Sequential Supervisory Control; Infinite Sequential Behaviors, Büchi Automata; ω-languages, *Liveness* and *Safety* issues in Supervisory Control.

8. **Performance Evaluation of DEDS**:

   (a) **Lesson 7**: Survey of *Queuing Theory*; (C++ code for) Markov Decision Problems; Case studies from Financial Engineering – Option Pricing using the Binomial Lattice via Recursion (in C++); Pricing a path-dependent (American-Asian) Option using Recursion (in C++); Computational limitations of the Binomial Model; Discretization of asset-price in to b-many values; (C++ code for an) $O(b^2 T)$-algorithm for pricing American Option of duration T discrete-steps using Dynamic Programming; (C++ code for an) $O(b^3 \log T)$-algorithm for pricing an European Option using Dynamic Programming and the method of *repeated-squaring*; Replication Portfolios; (C++ code for) Edirisinghe et al's approach to pricing European Options with transaction costs using Linear Programming. (C++ code for) Carr and Madan's approach to price an European Option with the *Fast Fourier Transform* (FFT).

2 Grade Composition

- $\approx$ 10 Programming Assignments (70%).
- $\approx$ 4 Homework Sets (20%).
- 1 Project (10%).

3 General Instructions

Tentatively, I am planning to have about 10 programming exercises. I will provide you with appropriate help/directions for you to be able to do these assignments. There is a knack to getting programming assignments done – it is a valuable skill-set for you to have. I will do my best to work with you if you need help. Your performance in these assignments will decide 70% of your final grade.

I plan to have $\approx$ 4 (traditional) homework sets for the course. Since the pace of the course will be dictated by the class needs/skills, the exact number of these assignments might vary. The contribution to the grade will be 20% independent of the number.

I would like you to apply any (one) of the methods/techniques to a problem that is close your research, this “project” will decide 10% of your grade. I am envisioning a couple of pages, together with some code, for this part.

Since this is a 500-level course, your proficiency in the course material will be tested primarily in your homework and programming assignments. I do not plan to have any examinations for the course. I will be available to help with the homework and programming assignments. I do not plan to have designated office hours for the course. We can set a time for us to talk following an initial e-mail from you.

I intend to use the $\pm$-grading system. My lecture notes for the course can be found on the University of Illinois' *Compass* website. I suggest you print the appropriate lesson before class and follow-along. This will free you from the tedium of copying material off the board during class, you can use that time to follow the material presented in class instead.

I prefer that you turn-in an electronic (PDF) version of your HWs and programming assignments. I will get back to you with my comments and I expect you to correct all egregious errors and send the revised version back to me. This way I will make sure you have a firm grasp of everything that is asked of you in each HW/programming-assignment. Once your work has been accepted by me, you will receive full-credit for it.

It is your responsibility to check the above URL regularly for updates/due-date-announcements as the course progresses.