1 Introduction

The subject of formal language (automata) theory is traditionally taught as a paper-and-pencil mathematics course, even though it is largely concerned with algorithms on the subject’s formalisms: regular expressions, automata, grammars and Turing machines. In contrast, this course will approach the subject in a way that balances proof with experimentation, carried out using the Forlan toolset, which implements the algorithms of formal language theory, and has been developed by the instructor over the last decade. Forlan is embedded in the functional programming language Standard ML, a language whose notation and concepts are similar to those of mathematics. It is strongly typed and interactive, properties that make sophisticated experimentation robust and enjoyable.

This course complements, and may be taken before or after, COMP 170 (Computation Theory). Prerequisite: COMP 15 (Data Structures) and MATH 22 (Discrete Mathematics).

2 Textbook

We will be using the Spring 2012 draft of my textbook on formal language theory, entitled *Formal Language Theory: Integrating Experimentation and Proof*. The book is available via the course website. The book will be posted in installments, as the semester progresses.

You may find it useful to also study the related sections of another book on formal language (automata) theory. Note, however, that the notation and definitions used by any particular book won’t be identical to the ones we’ll be using.

3 Forlan Toolset

Many of the algorithms that we will study are implemented as part of the Forlan toolset—a collection of tools for experimenting with formal languages that I’ve been developing since
2001. Forlan is implemented as a set of Standard ML (a functional programming language) modules. It’s used interactively. In fact, a Forlan session is nothing more than an SML session in which the Forlan modules are available.

As the course progresses, you’ll learn how to use Forlan, and you’ll be using it when solving some of the homework exercises. A link to the Forlan website can be found on the course website.

Forlan is available on the Department of Computer Science’s Linux system as ~stough/bin/forlan. You should either add ~stough/bin to your shell’s search path, or copy ~stough/bin/forlan to your own bin directory. (E.g., if your shell is bash, put

\[
\text{declare -x PATH=~stough/bin:$PATH}
\]

in the file .bash_profile of your home directory.)

See the Forlan website for instructions for installing Forlan on personal computers.

The best way to run Forlan or SML is as a sub-process of the Emacs text editor, using the SML mode for Emacs. On the Department of Computer Science’s Linux system, you can make SML mode available by putting the line

\[
(\text{load } \\
\text{"/h/stough/emacs/sml-mode-4.1/sml-mode-startup")}
\]

in the file .emacs of your home directory.

JForlan, a Java program for creating and editing Forlan automata and trees, is available on the Department of Computer Science’s Linux system as ~stough/bin/jforlan. You should either add ~stough/bin to your shell’s search path, or copy ~stough/bin/jforlan to your own bin directory. (Otherwise forlan won’t be able to find jforlan.)

4 \LaTeX Document Preparation System

When preparing solutions to assignments, you may wish to make use of the \LaTeX document preparation system. \LaTeX handles mathematics very well, and is widely used by academic computer scientists.

A link to information about \LaTeX can be found on the course website.

5 Assessment

There will be seven problem sets, each worth 100 points, plus a final project, worth 300 points.

Each assessment unit will be graded using the following 100 point scale: A+ (100), A (92), A- (84), B+ (76), B (68), B- (60), C+ (52), C (44), C- (36), D+ (28), D (20), D- (12), F (0). (E.g., a grade of 80 points should be thought of as being halfway between a B+ and an A-.)
Late work will be assessed a penalty of 20% during the first twenty-four hours. Work that is more than twenty-four hours late will not be graded.

I will take the elegance and simplicity of your work into account when doing grading.

I reserve the right to award extra credit to students who find errors in, or suggest improvements to, course materials.

A student’s grade for the course will be the letter (possibly followed by a + or -) whose value is nearest to the weighted average of the grades of the student’s exercise sets and final project. Ties will be resolved by selecting the better of the two grades.

6 Academic Integrity

Students should read the Tufts brochure on academic integrity located at:

http://uss.tufts.edu/studentaffairs/judicialaffairs/Academic%20Integrity.pdf

The work that you hand in must be your own. You may discuss assignments with others, but you may not base your work on other people’s work.

7 Email Communication

I will often communicate with the class by email, and so it is important that you read your email frequently.