

# CS 838: Program Analysis (Spring 2009)

John Tang Boyland  
boyland@cs.uwm.edu  
EMS 925, 229-6986

Hours: M 12:30–1:30pm, W 10:00–11:00am, R 1:00pm–2:00pm

A program is any formal notation used to communicate ideas precisely between people or between a person and a machine. Programs are increasingly being written and used by non-programmers to customize applications. Program *analysis* is the task of deriving information from programs. In this course, we will look beyond simple analyses such as lexical analysis and parsing to more “semantic” analysis, determining properties such as absence of type errors, cyclic dependencies or race conditions, and presence of liveness, strictness or execution independence.

## 1 Objectives

Participants will learn the basic terminology for published accounts of analysis in the literature; they will learn successful techniques for program analysis, their benefits and shortcomings; they will learn ways to understand program analyses, especially those based on accepted techniques; they will learn how to design and implement analyses that are more likely to be correct, efficient, comprehensible and extensible.

## 2 Requirements

Participants are expected to have practical knowledge of

- parsing, context-free grammars and abstract syntax trees;
- (at least) two different modern programming languages (e.g. C++ and any functional language);
- typical checks that compilers make for these languages.

If students are not sure whether they meet these requirements, they are invited to meet with the instructor or send email at any time to discuss whether they should take the course.

## 3 Required Reading

The textbook for the course is *Principles of Program Analysis* by Nielson, Nielson and Hankin. We will concentrate primarily on Chapters 1, 2, 4, 6 and the appendices.

## 4 Grading

The grade for the course will be computed from the following parts:

### 40% Homework

There will be a homework assignment almost every week. Each will be due when the next is assigned.

### 10% Participation

Each student must attend lecture and contribute orally to the classroom experience.

## 50% Project

Each student will (1) write a report, (2) do an oral presentation or (3) implement a subset of a classic or recent analysis published in an international conference or journal.

Exceptions to these rules can be made in extraordinary circumstances. Advance notice of a need for an exception should be given if possible.

All graded assignments must be your own work (your own words), but you may work with other people as long as you list their names prominently on the first page of the turned in homework. Whether or not you have permission of the other, submitting someone else's work as your own is *plagiarism*, which is a serious academic offense. Everyone is responsible for learning the material themselves.

## 5 Supplementary Materials

Supplementary materials will be passed out in lecture and/or made available on the class web page:

<http://www.cs.uwm.edu/classes/cs838>

## 6 Schedule

The following schedule is subject to change.

Week	Topic	Reading
January 27	Introduction; Lattices	Ch. 1.1–2, A
February 3	Data Flow and CB Analysis (intro)	Ch. 1.3–4
February 10	Control Flow Graphs and DFA	Ch. 2.1
February 17	Algorithms	Ch. 1.7,6.0–3
February 24	Correctness of DFA	Ch. 2.2
March 3	Frameworks	Ch. 2.3
March 10	Solving	Ch. 2.4
March 17	Spring break	
March 24	Interprocedural Analysis (1)	Ch. 2.5.0–2
March 31	Interprocedural Analysis (2)	Ch. 2.5.3–6
April 7	Type and Effect Systems	Ch. 1.5, notes
April 14	Abstract Interpretation (intro)	Ch. 1.6
April 21	Abstract Interpretation (details)	Ch. 4.0–1
April 28	Widenings	Ch. 4.2
May 5	Presentations	