

1 Basic Course Information

Course Rubric and Name CS 441 : Compiler Construction

Prerequisites CS 440 (Programming Languages and Translators)

Meeting Times TBA

Dates TBA

Location TBA

Instructor Dr. Mattox Beckman

Text We will use these items:

1. *Modern Compiler Implementation in ML*, Andrew Appel, ISBN 987-0-521-58274-1
2. Selections from (or summaries of) *Implementing Functional Languages a tutorial*, Simon Peyton Jones and David Lester
3. Course Handouts

2 Course Objectives

The purpose of this course is to give the student a working knowledge of modern compiler implementation, and practical experience implementing them.

During the class you will...

- Learn the basic algorithms necessary for modern compiler writing.
- Develop some increasingly sophisticated compilers.
- Implement the following compiler-related algorithms
 1. Abstract Syntax Tree Generation
 2. Symbol Tables
 3. Activation Records / Stack Frames / Heap Frames
 4. Imperative and Functional Intermediate Representations
 5. Basic Blocks and dead code elimination
 6. Instruction Selection
 7. Liveness analysis
 8. Register allocation

3 Lecture Topics

- Introductory Material (3 lectures)
 - Overview of compilers; the parts and functions
 - Overview of target architecture
 - Basic techniques for programming (Abstract Syntax Trees, fix-point operations, etc.)
Source language will be Haskell.
- Simple Compilation (2 lectures)
 - Source-to-source compiling
 - Jump optimization
- Parsing (4 lectures)
 - Review of Regular, LL, and LR parsing (1 lecture)
 - Legacy tools (lex/yacc) (1 lecture)
 - Monadic Parsers (2 lectures)
- Abstract Syntax Trees, including Generic Algebraic Data Types
- Symbol Tables (1 lecture)
- Function handling (2 lectures)
 - First order function (C-like)
 - Higher order functions (closures)
- Intermediate Representation (3 lectures)
 - IR Trees
 - G Machines (or similar system) (2 lectures)
- Basic Blocks / Traces (2 lectures)
- Instruction Selection / Maximal Munch
- Liveness Analysis
- Imperative Register Allocation
- Functional Register Allocation
- 2 exams (two lecture periods)
- In-class programming (6 lectures)