CS420 Overview

CS 420 “Compiler Construction”

This course covers the topics in design of programming language translators, including scanning, parsing, error recovery, code generation, and code improvement. A term project building a compiler and interpreter is given and its achievement is evaluated. The usage of many tools and programming skills can be learned through this course.

Prerequisite: Programming Languages, Data Structures

Instructor:
Name: Dongsoo Han
dshan@kaist.ac.kr
Office: N1, 704

TA: Kyuho Son
ableman@kaist.ac.kr

Basis for grades
25% mid-term exam
30% final exam
10% home work
30% term project
5% attendance

Overview

We will cover the following topics:

1. Lexical Analysis(scanning)
2. Syntax Analysis(parsing)
3. Context-sensitive Analysis
4. Intermediate Representation
5. Code Generation
6. Code Improvement Techniques
7. Software Patent (It is an experimental trial!)

The textbook for this course is “Compilers:
Principles, Techniques, and Tools” by Aho, Lam, Sethi, and Ullman

The reference book for this course is “Advanced
Compiler Design and Implementation” by Steven S. Muchnick

Overview(cont)

Programming Project

- Lexical/Syntax Analysis and Error Reporting
- Compiler Construction for a simple Programming Language
- Internal Data Structure Construction for Interpreters
- Debugger Tracing Value Changes of Variables

Learn to do the normal things – edit, compile, debug, make

Policies

- No late projects, incompletes allowed

Compilers

What is a compiler?

- A program that translates an executable program in one language into an executable program in another language
- The compiler typically lowers the level of abstraction of a program
- The compiler must generate a correct executable
- For optimizing compilers, we also expect the program produced to be better, in some way, than the original

Motivation

Why build compilers?

Why study compiler construction?

Why attend class?

Reasons:

- Compilers provide an essential interface between applications and architectures
- Compilers embody a wide range of theoretical techniques
- Compiler construction teaches programming and software engineering skills

Role of Compilers

High-level programming languages

- Increase programmer productivity
- Better maintenance
- Portable

Low-level machine details

- Instruction selection
- Addressing modes
- Pipelines
- Registers and cache
- Instruction level parallelism

Compilers are needed to efficiently bridge the gap!

Isn't it a solved problem?

> "Optimization for scalar machine is a problem that was solved 20 years ago"
>
> David Kuck
> Fall 1990 at Rice University

Machines (and languages) have changed since 1980

Changes in architecture (and languages)

⇨ changes in compilers

- new features present new problems
- changing costs lead to different concerns
- must re-engineer well-known solutions

Significant differences in performance

We have entered into the smart phone era!!

Interest

Compiler construction shows us a microcosmic view of computer science.

Artificial Intelligence: greedy algorithms, learning algorithms

Algorithms: graph algorithms, union-find, network flows, dynamic programming

Theory: dfa's for scanning, parser generators, lattice theory for analysis

Systems: allocation and naming, locality, synchronization

Architecture: pipeline management, memory hierarchy management, instruction set use

Inside a compiler, all these things come together.

As a result, compiler construction is challenging and fun

Compiler Construction

Compilers are large, complex pieces of software.

By working on compilers, you'll learn to use

- programming tools(compilers, debuggers)
- program-generation tools(lex, yacc)
- software libraries

Hopefully you will also enhance your software engineering skills.

Experience

You have used several compilers.
What qualities do you want in a compiler?

1. Correct code
2. Output runs fast
3. Compiler runs fast
4. Compile time proportional to program size
5. Support for separate compilation
6. Good diagnostics for syntax errors
7. Works well with the debugger
8. Good diagnostics for flow anomalies
9. Cross language calls
10. Consistent, predictable optimization

Some Advice

My emphasis on CS420 is mostly on generating a working program.

Compiler construction is one of the most time-consuming programming courses, along with operating systems and databases.

Be prepared to spend a lot of time on projects. The term project will be tough, challenging, and exciting at the same time.