Concepts of programming languages

Gaurish Sharma

September 14, 2023

Contents

Lecture 1: Compilers di 12 sep 2023 19:41

1 Bridging the gap

Definition 1. A compiler translates from source code to a target language, typically machine code.

Ex. C, C++, Haskell, Rust

Definition 2. An interpreter executes a program as it reads the source code Examples: perl, python, javascript

Note: Some languages make use of a hybrid approach. First translating the source language to an intermediate language (abstract or virtual machine), then interpret that. Examples: Java, C#

2 Stages of a compiler

Definition 3. The first stage of a compiler is called a lexer, which given an input string of source code , produces a stream of tokens or lexemes, discarding irrelevant information like whitepsace or comments.

Definition 4. A parser converts the stream of tokens from the lexer into a parse tree or abstract syntax tree.

Definition 5. The structure of lexemes expected to produce certain parse trees is called a grammar.

Definition 6. Backus-Naur Form: Specify grammatical productions using a barebones. recursive notation. Noneterminals are in italics whereas Terminals are in in boldface. for examples can look up online but you've seen this in compilers plenty of times.

Once there is an abstract syntax tree, symantic analysis is done:

2.1 Symantic Analysis

- Checks variable scoping.
- Static semantic checks: most notably type checking.
- Adds extra information to the tree.

Then we perform optimization

2.2 Opmitization

- Loop unrolling, loop fusion.

- Inlining, specialization.
- Sometimes transforms the tree dramatically.

2.3 Code Generation

- Register allocation and explicit control flow.
- Links runtime system (eg Garbage collector).
- Selects appropriate machine instructions.