Lecture 11+12

Context-Free Languages

CS 241: Foundations of Sequential Programs
Winter 2018

Troy Vasiga et al
University of Waterloo

Big Picture of Compilation

Big picture:

- lexical analysis
- syntactic analysis
- context-sensitive (semantic) analysis
- synthesis (code geneneration)

Other points:

- staging can improve error messages
- staging uses the "right tool for the job"
- doing extra-work at an early stage is possible but over-complicating

Non-regular languages

Give a DFA over $\Sigma = \{a, b\}$ for

$$\mathsf{L} = \{\mathsf{w}\text{: numbers of a's in w} = \text{number of b's in w}\}$$

Why the previous language is important

- L is not a regular language, but it is *context-free*
- Note that a compiler needs to do this and then some:

Contex-Free Languages

- Context-free languages are built from:
  - finite sets
  - concatenation
  - union
  - recursion

- recognizers for regular languages use a finite amount of memory

- recognizers for context-free languages use a finite amount of memory plus one (unbounded) stack

Context-Free Grammars

A way to specify a context-free language.

A CFG Example

$$
\begin{aligned}
S &\rightarrow aSb \\
S &\rightarrow D \\
D &\rightarrow cD \\
D &\rightarrow \epsilon
\end{aligned}
$$

Discussion of example

- G:
- L(G):
- a word:
- a derivation:
- alternation and concatenation
- recursion vs. repetition

Formal Definition

A context-free grammar (CFG) consists of

- N
- T
- P
- S

Example, more formally

From the example, what is N, T, P, S?

$$
\begin{aligned}
S &\rightarrow aSb \\
S &\rightarrow D \\
D &\rightarrow cD \\
D &\rightarrow \epsilon
\end{aligned}
$$

Example: balanced parentheses

Example words:

CFG:

A sample derivation:

Binary expressions

Words are composed of binary numbers (no leading zeros, other than 0) with + or - signs in infix notation.

Example words: 1001, 10+1, 11-11110+0

CFG:

1.

2.

3.

4.

5.

6.

7.

8.

Derivations

Two derivations of 10+1 using the previous grammar:

Formal Definitions

We say that $\alpha A \beta$ *directly derives* $\alpha \gamma \beta$ if there exists a production rule $A \to \gamma$.
Also called a *derivation step*.

We say that $\alpha A \beta$ *derives* $\alpha \gamma \beta$ if

Left-most and right-most derivations (formally)

Formal Defintions (continued)

G derives $w \in T^*$ if

L(G) =

L is context-free if

Derivations as Proofs

Parse Trees

Example:

Discussion:

Meaning of a parse tree

Problems that grammars can encounter: Ambiguity

A real-world example of ambiguity:

Ambiguity in CFGs: Formal Definition

- A string x is ambiguous if...

- A CFG G is ambiguous if...

Problems that grammars can encounter: Ambiguity

Terminology:

Problems that grammars can encounter: Ambiguity

Consider the binary expression grammar, and 1-10+11

Fixing ambiguity

Let's rewrite our binary expression grammar differently.

1.

2.

3.

4.

5.

6.

7.

8.

Ambiguity in programming languages

- Some programming languages have ambiguous grammars
- Pascal has a "dangling else"
- "Fixed" by way of a footnote

Associativity

1-10+11

Fixing associativity problems

1.

2.

3.

4.

5.

6.

7.

8.

Precedence Problems

Adding multiplication to the grammar.

Consider 1*10+11.

Consider 1+10*11

Fixing precedence problems: grammar

Fixing precedence problems: parse tree

Regular Langages are Context-Free Languages

Regular Langages vs. Context-Free Languages

Assignment Tips

- WLP grammar
- `.cfg` file format
- recursion is your friend

Summary

- Context-free grammars are used to specify a language
- Derivations (or parse trees) act as proof of a word $w \in L(G)$
- We would really like to generate derivations automatically