CS536 Lecture 7

Tuesday 10 February 2015

Reminders:

- HW2 due tonight, HW3 assigned.
- Reading

Last class:

- Limitations of Regular Expressions
- Context-Free Grammars
    - Definition
    - Examples

Today:

- Makefiles
- CFG Review
- Ambiguity in grammars and resolving it

Makefiles

Basic structure:

```
<target>: <dependency list>
(tab)    <command to "make" the target>
```

Example:

```
Example.class: Example.java IO.class
    javac Example.java

IO.class: IO.java
    javac IO.java
```

Can view it as a dependency graph.

A file is rebuilt *if one of its dependencies changes*.

Variables

```
JAVA_COMPILER = /usr/bin/javac
JAVA_FLAGS = -g
```

Use: `$(JAVA_COMPILER) $(JAVA_FLAGS) Example.java`

"Phony" targets

CFG Review

Grammar:

Language of a CFG

BNF

List Grammars

To repeat a structure arbitrarily often.

```
Stmts → Stmts SEMICOLON Stmt | Stmt
```

Parse tree form:

List Grammars (cont'd)

How about this one?

Stmts → *Stmt SEMICOLON Stmt***s** | *Stmt*

Parse tree form:

List grammars (cont'd)

Allow both skews:

Stmts → *Stmt**s*** *SEMICOLON* *Stmt**s*** | *Stmt*

Parse tree form:

Derivation Order and Ambiguity

Leftmost Derivation:

Rightmost Derivation:

Even with a fixed derivation order, it may be possible to derive the same string in multiple ways.

A grammar G is “ambiguous” if

- there exists >1 leftmost derivation of a string w
- there exists >1 rightmost derivation of a string w
- there exists >1 parse tree for w

These are all equivalent.

Example of an Ambiguous Grammar

```
Expr → INTLIT
     | Expr MINUS Expr
     | Expr TIMES Expr
     | LPAREN Expr RPAREN
```

Derive: `4 - 7 * 3` (assume tokenization is complete)

Our position on ambiguity

Resolving Ambiguity 1: Precedence

Intuitive Problem:

Fix precedence by

-
-

Instead of

```
Expr → INTLIT
     | Expr MINUS Expr
     | Expr TIMES Expr
     | LPAREN Expr RPAREN
```

we have

```
Expr → Expr MINUS Expr
     | Term

Term → Term TIMES Term
     | Factor

Factor → INTLIT
       | LPAREN Expr RPAREN
```

Example Derivation

```
4 - 7 * 3
```

Try to build the wrong parse tree: