CS344 - Context Free Languages

Note Title 1/30/2012

Announcements

- HW due Friday

7329 ← decimal: no left 0

0x7341 ← hexidecimal

0 ← octal

flex

A demo or two

```
flex filename.lex
gcc lex.yy.c -lfl
```

Context-Free Languages

$A \to BCD$

$B \to bc$

Recall that for any context free languages there are an infinite # of grammars that can produce it.

We wish to somehow give a definition of a "good" set of productions.

Goal: Parsing (well) - given a language, detect if a string is in that language.

Ex: Bad example

$S_0 \rightarrow S \mid X \mid Z \mid A$

~~$S \rightarrow A$~~

$A \rightarrow B$

$C \rightarrow Aa$

$X \rightarrow C$

$Y \rightarrow aY \mid a$

$Z \rightarrow \varepsilon$

- B is useless
- Y is unreachable

(S, A)

(S, B)

Chomsky Normal Forms (CNF)

Each rule in the grammar is either:

- $A \rightarrow BC$
  where neither B or C is the start variable, & both are nonterminals
- $A \rightarrow a$
  where a is a terminal
- $S \rightarrow \varepsilon$
  where S is the start symbol

Thm: All grammars can be converted to CNF.

Procedure:

① Create a new start symbol S_0, & send $S_0 \to S$

Eliminate useless rules

(just delete ones that can't be reached)

O(n)

② Remove nullable variables.
$A \to \varepsilon$

How?
Remove all ε productions.
Then fix.

$A \to CBC \mid CC$

⋮

~~$B \to \varepsilon$~~

$A \to \varepsilon$ ← to remove, just add another rule to anything with A on right hand side

$X \to YZ,\ Z \to \varepsilon$

(3) Remove unit rules: $O(n^3)$

$S \to A$

How? Must have:

$X \to Z_1,\ Z_1 \to Z_2,\ \ldots,\ Z_k \to Y$

(since we removed ε-transitions in (2))

find all these pairs $X \Rightarrow Y$

$Y \to a$

$Y \to BCD$

$Y \to aB$

$\to$

For each unit pair (A, B) and rule $B \to w$, $B \to YZ$, ...

add $A \to w$ to a new grammar.

(Note that (A, A) is a unit pair, so all rules $A \to w$ will stick around.)

④ Get rid of "long" righthand sides.

4a: Create $V_c \to c$ for every character.

Replace c with V_c everywhere.

Now either

$$A \to CDEF$$

or

$$V_c \to c.$$

4b: $A \rightarrow B_1 B_2 B_3 \dots B_k$

How to replace with only 2 nonterminals on the right?

$A \to B \to \varepsilon$

Ex:

$S \to ASA \mid aB$

$A \to B \mid S$

$B \to b \mid \varepsilon$

① Removing ε: $S \to ASA \mid aB \mid a \mid AS \mid SA$

$A \to S \mid B$

$B \to b$

② Removable pairs: (A,B); (A,S)

Ex (cont):

Now — why do we care??

Parsing: building those parse trees we saw

In general, there are an exponential number of parse trees for a given input.

So how to check quickly?

Even in CNF, might be 2^n possible parse trees.

Cocke - Younger - Kasami (CYK) algorithm

Uses a table & dynamic programming to give a parse tree in $O(n^3)$ time.

Grammar must be in CNF!