

CS344 - Context Free Languages

Note Title

1/30/2012

Announcements

- HW due Friday

7329 ← decimal: no left 0

0x7341 ← hexadecimal

0 octal

flex

A demo or two

flex filename.lex

gcc lex.yy.c -lfl

Context-Free Languages

$A \rightarrow BCD$

$B \rightarrow bc$

Recall that for any context free language, 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

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

~~$S \rightarrow A$~~

$A \rightarrow B$

$C \rightarrow A_a$

$X \rightarrow C$

$\rightarrow Y \rightarrow_a Y \mid a$

$Z \rightarrow \varepsilon$

- 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

- $S \rightarrow \epsilon$

where S is the start symbol

Thm: All grammars can be converted
to CNF.

Procedure:

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

Eliminate useless rules

(just delete ones that
can't be reached)

② Remove nullable variables.
 $A \rightarrow \epsilon$

$O(n)$

How?

Remove all ϵ productions.
Then fix.

$A \rightarrow CBC \mid CC$

⋮

~~$\rightarrow B \rightarrow \epsilon$~~

$A \rightarrow \epsilon$

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

$$X \rightarrow YZ, Z \rightarrow \epsilon$$

③ Remove unit rules:

$$O(n^3)$$

$$S \rightarrow A$$

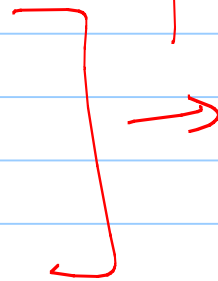
How? Must have:

$$X \rightarrow z_1, z_1 \rightarrow z_2, \dots, z_k \rightarrow Y$$

(since we removed ϵ -transitions in ②)

find all these pairs $X \Rightarrow Y$

$$\begin{array}{l} Y \rightarrow a \\ Y \rightarrow ABCD \\ Y \rightarrow aB \end{array}$$



For each unit pair (A, B)
and rule $B \rightarrow w$, $B \rightarrow YZ, \dots$
add $A \rightarrow w$ to a new
grammar.

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

④ Get rid of "long" righthand sides.

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

Replace c with V_c everywhere.

Now either

$A \rightarrow CDEF$

or

$V_c \rightarrow c.$

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

How to replace with only
2 nonterminals on the
right?

$$A \rightarrow B \rightarrow \epsilon$$

Ex:

$$S \rightarrow ASA \mid aB$$

$$A \rightarrow B \mid S$$

$$B \rightarrow b \mid \epsilon$$

① Removing ϵ :

$$S \rightarrow ASA \mid aB \mid a$$

$$AS \mid SA$$

$$A \rightarrow S \mid B$$

② Removable pairs: $B \rightarrow b$
 $(A, B), (A, S)$

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!