Exercises on Chomsky Normal Form and CYK parsing

1. Convert the following grammar to CNF

S -> A (S) B | “”

A -> S | S B | x | “”

B -> S B | y

Exercise 2

Which of the following properties of a grammar are preserved by the "Epsilon Production Elimination" algorithm

- Ambiguity No, counter-example ?
  - If a grammar is "ambiguous" does it remain ambiguous after removing epsilon productions ?
- LL(1) No, counter-example ?
- No Left Recursion No, counter-example ?
- Unambiguity ? Yes, proof ?

Exercise 2
[Part 2]

Which of the following properties of a grammar are preserved by the "Unit Production Elimination" algorithm

- Ambiguity No, counter-example ?
- Left Recursion May be
  - What about these rules: B -> A | a , A -> B ?
- LL(1) Yes, proof ?
- Unambiguity Yes, proof ?

Exercise 3

Show the CYK parsing table for the string “aabbbab”

for the grammar

S -> AB| BA | SS | AC | BD

A -> a

B -> b

C -> SB

D -> SA

What should be done to construct a parse for the string

Exercise 4

Should a grammar be strictly in CNF form for CYK to work ? If not, what are the properties that can be relaxed ?

A CYK for Any Grammar

grammar G, non-terminals $A_1,\ldots,A_K$, tokens $t_1,\ldots.t_L$

input word: $w = w_{(0)}w_{(1)} \ldots w_{(N-1)}$

$w_{p..q} = w_{(p)}w_{(p+1)} \ldots w_{(q-1)}$

Triple (A, p, q) means: $A \Rightarrow^* w_{p..q}$, A can be: A_i, t_j, or ε

$P = \{(w_{(i)},i,i+1) \mid 0 \leq i < N-1\}$

repeat {

choose rule $(A::=B_1\ldots B_m) \in G$

if $((A,p_0,p_m) \notin P$ &&

$((m=0 \ \&\& \ p_0=p_m) \ || \ (B_1,p_0,p_1), \ldots,(B_m,p_{m-1},p_m) \in P))$

$P := P \cup \{(A,p_0,p_m)\}$

} until no more insertions possible

What is the maximal number of steps?

How long does it take to check step for a rule?

$\Big\}$ for grammar in given normal form

Observation

- How many ways are there to split a string of length Q into m segments?

$$\binom{Q+m}{m} = \frac{(Q+m)!}{Q!\,m!}$$

- Exponential in m, so algorithm is exponential.
- For binary rules, m=2, so algorithm is efficient.

Exercise 5

Assume a grammar in CNF has *n* non-terminals. Show that if the grammar can generate a word with a derivation having at least 2^n steps, then the recognized language should be infinite

Closure Properties of CFG

- Concatenation
  - If L1 and L2 are context-free languages is $L = \{xy \mid x \in L1, y \in L2\}$ also context-free ?
- Union
- Closure
- Complement ?
  - Not always a CFG, but sometimes possible
- We can convert any regular expression to a CFG

Complement of a Grammar

- Compute the complement of the grammar
  - S -> A | B
  - A -> a A | “”
  - B -> b B | “”