CS3200 – CYK & CNF (cont.)

Note Title 1/27/2016

- HW due Saturday

Last time:

Chomsky Normal Form:

$A \to BC$

$X \to x$

and ε <u>only</u> from start state

Why?

CYK algorithm requires CNF.

Conversion example:

$$S \to ASB$$
$$A \to aAS \mid a \mid \varepsilon$$
$$B \to SbS \mid A \mid bb$$

4 steps:

- remove ε transitions & get new start state
- remove unit pairs
- need only 2 non-terminals
- replace terminals in pairs w/ dedicated nonterminal

① New start state, & eliminate ε rules:

$$S_0 \to S$$
$$S \to ASB$$
$$A \to aAS \mid a \mid \varepsilon$$
$$B \to SbS \mid A \mid bb$$

$$S_0 \to S$$
$$S \to ASB \mid SB$$
$$A \to aAS \mid a \mid aS$$
$$B \to SbS \mid A \mid bb \mid \varepsilon$$

$$S_0 \to S$$
$$S \to ASB \mid SB \mid AS \mid S$$
$$A \to aAS \mid a \mid aS$$
$$B \to SbS \mid A \mid bb$$

② Remove Unit rules

$S_0 \to ASB \mid SB \mid AS$

$\cancel{S_0 \to S}$

$S \to ASB \mid SB \mid AS \mid \cancel{S}$

$A \to aAS \mid a \mid aS$

$\cancel{B \to SbS \mid \underline{A} \mid bb}$

$B \to SbS \mid bb \mid aAS \mid a \mid aS$

(3) Fix so that we have all pairs

$$S_0 \to \underbrace{AS}B \mid SB \mid AS$$

$$S \to \underbrace{AS}B \mid SB \mid AS$$

$$A \to \underbrace{aAS} \mid a \mid aS$$

$$B \to \underbrace{SbS} \mid bb \mid \underbrace{aAS} \mid a \mid aS$$

$$S_0 \to AU_1 \mid SB \mid AS$$

$$U_1 \to SB$$

$$S \to AU_1 \mid SB \mid AS$$

$$A \to aU_2 \mid a \mid aS$$

$$U_2 \to AS$$

$$B \to SU_3 \mid bb \mid aU_2 \mid a \mid aS$$

$$U_3 \to bS$$

④ Finally, need only non-terminal pairs

$S_0 \rightarrow AU_1 \mid SB \mid AS$ ✓

$U_1 \rightarrow SB$ ✓

$S \rightarrow AU_1 \mid SB \mid AS$ ✓

$A \rightarrow \overset{U_4}{\cancel{a}} U_2 \mid a \mid \overset{U_4}{\cancel{a}} S$

$U_2 \rightarrow AS$

$B \rightarrow SU_3 \mid \overset{U_5 U_5}{\cancel{bb}} \mid \overset{U_4}{\cancel{a}} U_2 \mid a \mid \overset{U_4}{\cancel{a}} S$

$U_3 \rightarrow \overset{U_5}{\cancel{b}} S$

$U_4 \rightarrow a$

$U_5 \rightarrow b$

Running time:

Actually depends on the order steps are performed in, since some operations undo others.

ie: deleting ε-rules then eliminate right hands > 2

$\Rightarrow$ exponential blow-up

but reverse is a linear operation

Bottom line: $O(n^2)$

And the why: CYK algorithm

in CNF!

An algorithm which, given a grammar and a word, decides if the word can be produced by the grammar.

Runtime: $O(n^3)$

How?

$$
\begin{aligned}
S &\rightarrow NP\ VP \\
VP &\rightarrow VP\ PP \\
VP &\rightarrow V\ NP \\
VP &\rightarrow eats \\
PP &\rightarrow P\ NP \\
NP &\rightarrow Det\ N \\
NP &\rightarrow she \\
V &\rightarrow eats \\
P &\rightarrow with \\
N &\rightarrow fish \\
N &\rightarrow fork \\
Det &\rightarrow a
\end{aligned}
$$

CYK table

S						
	VP					
S						
	VP			PP		
S		NP			NP	
NP	V, VP	Det.	N	P	Det	N
she	eats	a	fish	with	a	fork

Pseudocode: (length of word)^3 (# of productions)

```
let the input be a string S consisting of n characters: a1 ... an.
let the grammar contain r nonterminal symbols R1 ... Rr.
This grammar contains the subset Rs which is the set of start symbols.
let P[n,n,r] be an array of booleans. Initialize all elements of P to false.
for each i = 1 to n
  for each unit production Rj -> ai
    set P[1,i,j] = true
for each i = 2 to n -- Length of span
  for each j = 1 to n-i+1 -- Start of span
    for each k = 1 to i-1 -- Partition of span
      for each production RA -> RB RC
        if P[k,j,B] and P[i-k,j+k,C] then set P[i,j,A] = true
if any of P[n,1,x] is true (x is iterated over the set s, where s are all the indices for Rs) then
  S is member of language
else
  S is not member of language
```