Champley Normal Form

* Helpful to have a suiplified form for G.

Need to be efficient to check if wEL(G).

Should not have loops (in durnation).

+ Should not have surpty derivations, or weless rules.

* Should have simple rules that are every to check.

Del 2.8: A content-free grammar is in Chomsley if every rule is of the form Normal Form

A -> BC

where a is a terminal, A,B, C are nariables

and Band C should not be the start variable.

Aleo, ve allow 8 -> E

Theorem 2.9: May CFL is generated by a CFG in

Chomby Normal Form.

Proof: We connect any CFG into CNF.

Optimal Step: Remone walers symbols 2 productions.

1. Add new start variable So.

g -> S

This guarantees that So does not appear in the RHS of any rule.

2. lemme & rules.

Say 4 > E. Then modify sules with A in RHS. If R > n Av was a sule, then

R-> n Aulnu.

R-> A bermer R-> A/E

unless R-SE is already removed.

3. Remne unit rules like A >> B.
When B > u appears, add A >> u unless
it was already removed.

4. Perturture long RHS.

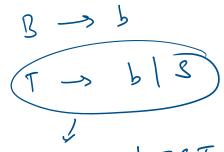
4. Perturture long RHS.

(1) Add So -> S.

(2) Remone B -> E

T > E

(3) Unit Rules



T-> 6 | TST | ST | TS | aB | a

(4) Restaucture rules with more than me symbol in RHS.

$$S \longrightarrow 2T | TS | ST | AB | a$$

→> b.

The main advantage of CNF is that there is "predictability" in the desiration of a storing brug storing of length in requires exactly 2n-1 steps for demation. Why?

Choneshy Normal Form results in an algorithm for cheeling if we is generated by b.

CYK algorithm

John Corke 1970 Daniel Yourger 1967 Tado Karani 1965

One name approach is to try out all derivations with 2n-1 steps. This is not time efficient. The CYK algorithm is a dynamic programming algorithm.)

Use subproblems to solve larger problems.

Vse subproblems to solve larger problems. O(n3) time. where each a; E E let 0 = a, a2 ... an wij = ai ain ... aj. for all 1 & i & j & v. CYK builds Tij for all 1 Ei Eij En, such that Tij = { A | A => wis}, leading up to Tin. Finally, check if SETIN. That S * WIN=W T11 T22 T33 T12 T23 T34 .. Th-1, N Order of T13 T24 ... 1 N-2, N anjuting Ti, is. T1, n-1 T2, N Tun

For k=0 to n-1, compute all T(i,i+k)

For k=0 to n-1, compute all '(i, 1+k) k=0. The second $\sum_{i=1}^{n} A = \sum_{i=1}^{n} \omega_{i,i} = A_{i,i}$.

All the rules A -> a:

For k70, ACTi, itk iff BETi, it and CETititi, itk and A-> BC is a sule.

If $\omega = \varepsilon$, anept if $s \rightarrow \varepsilon$ is a rule.

For i=1 to N

ACTii () A -> ai is a rule.

For k= 1 to n-1.

For i= 1 to N-1c

For 's=0 to k-1

Chech all rules R-> AB.

If Ti, it's contains A, and

Titit, it contains B

then Ti, itk = Ti, itk U ERJ.

If SET,,n, say we L(G).

If SETI, n, May WE LIGHT. Ehre. W& L(G).

Running time: O(13 x) where r is no. of rules.

Correctives is evident from the algorithm.

Example:

 $S \rightarrow AB \mid BC$ $A \rightarrow BA \mid A$ $B \rightarrow CC \mid b$ $C \rightarrow AB \mid A$ $C \rightarrow AB \mid A$

Exercise: Verify this materia.