CSE322 Theory of Computation (L20)

$L = \{ \langle D \rangle \mid D$ is the smallest DFA w.r.t. the number of states accepting $L(D) \}$

$L' = \{ \langle N \rangle \mid N \dots$ NFA $\}$

$S \to w_1 \to w_2 \to \dots$

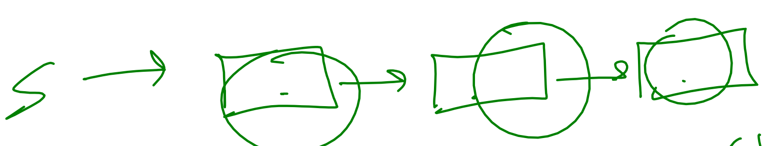
$\to w_{2n-2} \to w$

$S \to \varepsilon$ $\times$

PDA P, string w :-

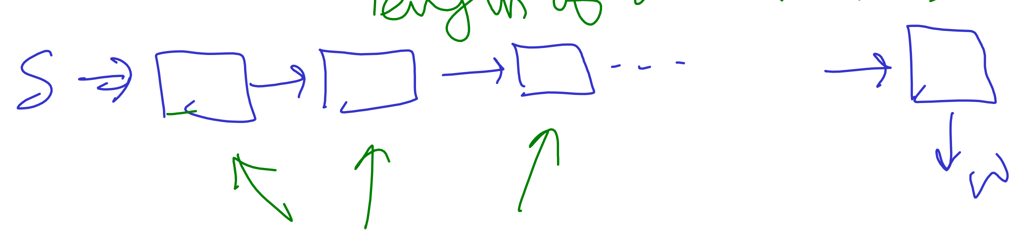
||

G $\qquad S \Rightarrow^* w$



$\leq |w|$

length of derivation is $2n-1$



has length $\leq n$

is a string over $V \cup T = V'$

Chomsky Normal Form

Given G in CNF, & w, it is possible to decide if G derives w.

Every rule is of the form: $\boxed{A \to BC}$ n-1

or, $\boxed{A \to a}$ (a is terminal)

n

S does not appear on the right side

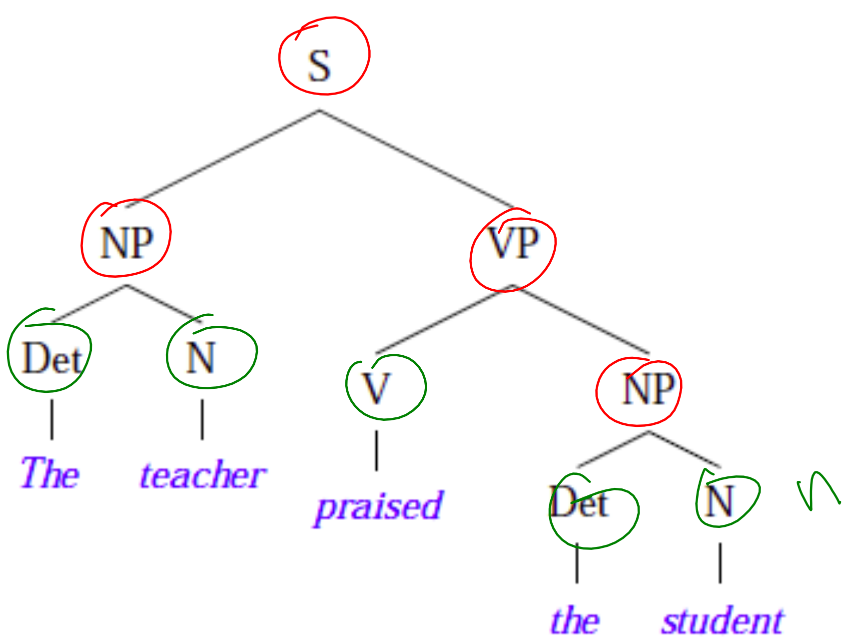
$\leq$ n

Additional rule, if needed: S -> e

S → ☐ → ☐ - - -

Try all possible derivations of length 2n-1.

Thm: Any CNF derivation of an n-length string has 2n-1 steps.



\# leaf nodes = ?

\# internal nodes = ?

Minimum depth = 1 + ceil(log n)

Maximum depth = n

$D \to A$

$\cancel{A \to \varepsilon}$

$B \to CADA$

$B \to CADA \mid CDA \mid CAD \mid CD$

1. Ensure S does not appear on right side of any rule

    Add: S0 -> S

2. Remove e-rules "X -> e" for all X (except S)

    Find all nullable variables.

    Λ

    A is nullable if there is a rule: A -> e or there is a rule A -> B1 B2 B3 in which all Bi are nullable.

    Remove A -> e for all nullable A.

    For every rule B -> ... A ..., create a copy with A replaced by e.

3. Remove unit rules: "X -> Y" for all Y

    A-derivable $= \{B \in V : A \to B$ or $A \to C \to \dots \to B\}$

    B is A-derivable if A->B or there exists some A-derivable C s.t. C -> B

    For all (A,B) s.t. A =>* B (B is A-derivable)

    $\cancel{A \to B}$ $\qquad B \to w$

    For every rule non-unit production B -> x, add a rule A -> x

    $A \to w$

    Remove all unit productions

4. Simplify rule "A -> t1 t2 t3 ... tk" (ti could be terminal or variable)

    Replace: A -> CxB by A -> CXB and X -> x ✓

    Replace A -> BCD by A -> BE, E -> CD

Convert to CNF

$S0 \to S$

$S \to ASA$

$S \to aB$

$A \to B$

$A \to S$

$B \to b$

$B \to \cancel{\varepsilon}$ $\emptyset$

$A \to B$ ✗

$B \to \emptyset$

$A \to \emptyset$

B, A

$S \to ASA \mid AS \mid SA \mid S$

$S \to \ldots$ $\quad S \to ASA$

$\to \quad SA \to \ldots$

Finding nullable variables

$N = \{ X \in V : X \to \varepsilon \text{ is a rule} \}$

do {

for every rule $X \to Y_1 Y_2 \cdots Y_K$ //only variables

if $Y_1 \cdots Y_K$ are all nullable

add X to N

} while (N is updated)

Finding A-deriverable variables

$N = \{ \text{variable } X \text{ s.t. } A \to X \text{ is a rule} \}$

do {

for every unit production $B \to C$ (variable)

if B is A-derivable

add C to N

} while (N is updated)

CYK algorithm

$X_{i,j} = \{A, B \ldots\}$

which can generate

$w_i \ldots w_j$

$X2,2 = ?$

$X2,3 = ?$

$X_{1,5} =$				
$X_{1,4} =$	$X_{2,5} =$			
$X_{1,3} =$	$X_{2,4} =$	$X_{3,5} =$		
$X_{1,2} = A,S$	$X_{2,3} =$	$X_{3,4} =$	$X_{4,5} =$	
$X_{1,1} = B$	$X_{2,2} = A,C$	$X_{3,3} =$	$X_{4,4} =$	$X_{5,5} =$
w_1	w_2	w_3	w_4	w_5
b	a	a	b	a

Does this CFG produce baaba?

$$
\begin{array}{lcl}
S & \rightarrow & AB \mid BC \\
A & \rightarrow & BA \mid a \\
B & \rightarrow & CC \mid b \\
C & \rightarrow & AB \mid a
\end{array}
$$

Add to $X_{i,j}$ any V s.t.

V->YZ exists,

for some k, Y in X_{ik} & Z in X_{kj}

Look at all pairs YZ from

$X(i,i)$ & $X(i+1,j)$,

$X(i,i+1)$ & $X(i+2,j)$ ...

$X(i,j-1)$ & $X(j,j)$

Add V s.t. V -> YZ

A-PDA = { ⟨P, w⟩ | L(P) ∋ w }.

A-CFG = { ⟨G,w⟩ : G generates w }

A-CFG is decidable.

G → G′ (CNF) check if G′ generates w.

Claim

If L is a CFL, } ∃ G s.t. L = L(G). M-A-CFG(G, w) to decide
then L is decidable } if w ∈ L.

Context-Free Languages are Decidable

Suppose C is a CFL.

Therefore, there exists a PDA / CFG to recognize C.

Construct MC -- use PDA / CFG

MC (w) : // accept if w is in C, reject otherwise

1. Run A-CFG(⟨G,w⟩) ...

M(x) :
run M-A-CFG(G, x) ← decider for ACFG.
and do whatever it does.

Claim : If x ∈ L,
M accepts.

Claim :- If x ∉ L,
M rejects.

$L = \{\langle P,q\rangle$: P is a PDA, q is its state, q is a not a useless state (not involved in any transition for any input string) $\}$

Is L recognizable? Is L decidable?

$ML(\langle P,q\rangle)$:

P': copy of P with "q removed" ✗

Decide if $L(P) = L(P')$. ✗

$EQPDA = \{\langle P_1, P_2\rangle : L(P_1) = L(P_2)\}$

$ML'(\langle P,q\rangle)$:

Non-deterministically guess a w

Run P(w) making non-deterministic choices at every step.

If ever q is reached, goto q_{acc}.

Claim:- If $\langle P,q\rangle \in L$, for some w & some nondet. choice, P will reach q $\Rightarrow$ ML' accepts.

Claim:- Exercise.

L1 = {<P,q> : P is a PDA, q is its state, q is a useless state (not involved in any transition for any input string) }

Is L1 decidable? Hint: What is complement of L (from above)?

Exercise :-

$$\text{A-TM} = \{\langle M, w\rangle : \text{TM } M \text{ accepts } w\}$$

Thm. A-TM is recognizable. Use a UTM to recognize.

1936: HALTING (similar) is not decidable (but recognizable).

$$= \{\langle M, w\rangle : \text{TM } M \text{ halts on } w\}$$

$MH(\langle M, w\rangle)$:

Run M on w using UTM.

If M either accepts or rejects, goto q_{accept}.

$$\text{POLYSOL} = \{p(x1, x2, \ldots xn) : p \text{ is a polynomial with integer solutions}\}$$

Thm. POLYSOL is recognizable.

1970: POLYSOL is not decidable (but recognizable).

A-TM = {<M,w> : TM M accepts w}

	y_1 1	y_2 2	3	4	....
M_1	X				
M_2		X			
M_3			X		
M_4				X	
⋮					

Thm. A-TM is undecidable.

Proof: Proof by contradiction and diagonalization against all TMs.

Let A-TM be decidable by TM D.

Proof by contradiction: Construct a TM (using D) and show that this TM cannot be in the list of all TMs.

$\forall i$, $U(i)$ is different from $M_i(i)$.

U(y):

M = "get encoding of y-th TM"

Run D(<M,y>)

If D accepts, // M(y) accepts ~~w~~

U rejects

If D rejects, // M(y) does not accept ~~w~~

U accept

Suppose $U = M_{53}$

Some y s.t.

$\langle y \rangle = 53$

$U(y) =$

This is the first method for proving undecid.

Thm. U(y) != My(y) for all y, therefore U != My for all y.