☆ Regular Grammars & Types

Grammar which follows,

$A \rightarrow Bx$ + LLG

RLG + $A \rightarrow xB$

$A \rightarrow x$

$A, B \in V$

$x \in T^*$

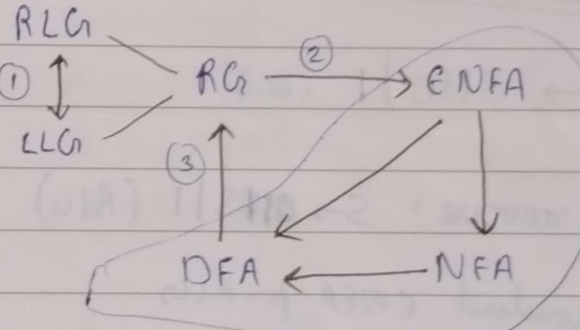
Right reg. grammar / Right linear " (RLG)

Left reg gra. Left linear " (LLG)

RLG ⟷ LLG

* Equivalence of RG & FA

We have to show the equivalency of 1, 2, 3 →



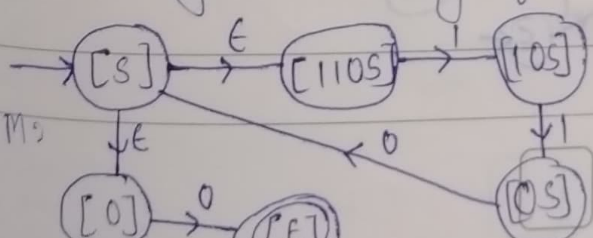
This is → equivalent as we already know

RLG → ϵ-NFA

1) G: $S \rightarrow 110S \mid 0$, $\Sigma = \{0, 1\}$ $L(G) = \{0, 110, 1101100, 110110110 0, \ldots\}$

Write all the states, $Q = \{[S], [110S], [10S], [0S], [0], [\epsilon]\}$; $S = [S]$; $F = \{[\epsilon]\}$

(States will consist of firstly the left variable, then all the whole words generated by var. then right substring of each word and an end state ϵ.)



Steps :- Create an initial state with start var.
- Create ϵ transition with state (consist of whole word that can be generated)
- Logically built the rest with some terminal as a transition.

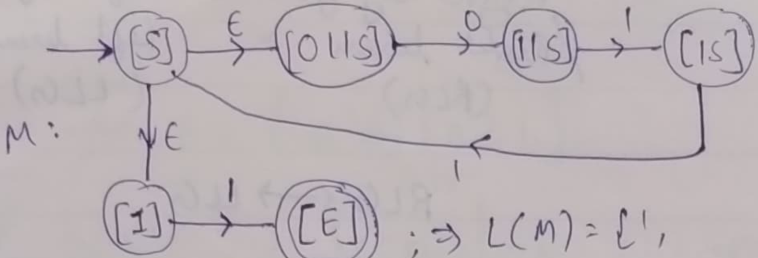
↳ $L(M) = \{0, 1100, 1101100, 1101101100, \ldots\} = L(G)$

2) G: $S \to 011S \mid 1 \Rightarrow L(G) = \{1, 0111, 0110111, \ldots\}$

$Q = \{[S], [011S], [11S], [1S], [1], [\epsilon]\}$

$S = [S]$; $F = \{[\epsilon]\}$

M:

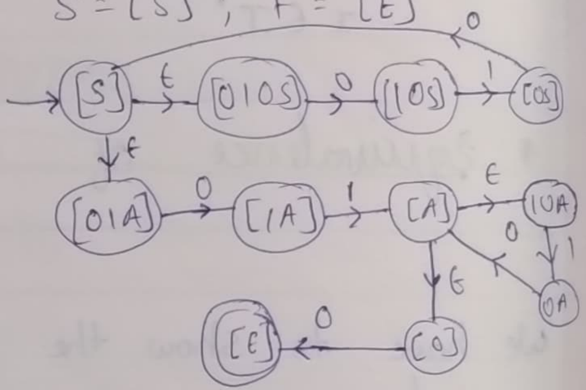


; $\Rightarrow L(M) = \{1, 0111, 0110111, \ldots\} = L(G)$

3) $S \to 010S \mid 01A$
$A \to 10A \mid 0$
$\Sigma = \{0, 1\}$

$Q = \{[S], [01A], [1A], [A], [010S], [10S], [0S], [0], [10A], [0A], [\epsilon]\}$

$S = [S]$; $F = [\epsilon]$



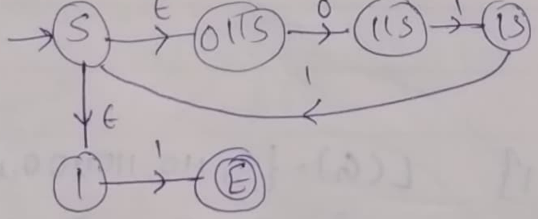
LLG → ε NFA

1) $S \to S110 \mid 1$ (LLG)

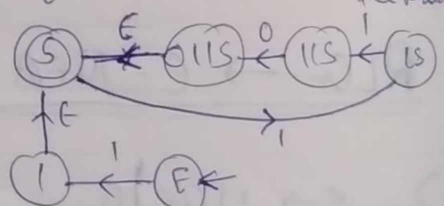
Steps

a) reverse: $S \to 011S \mid 1$ (RLG)

b) construct εNFA for RLG



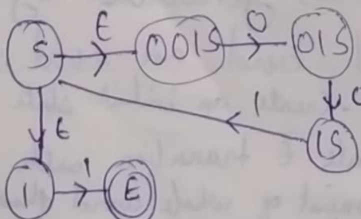
c) reverse εNFA
i.e. make initial as final & final as initial & reverse the transition arrow



2) $S \to S100 \mid 1$

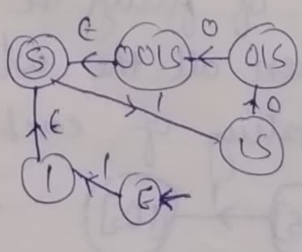
a) $S \to 001S \mid 1$

b)

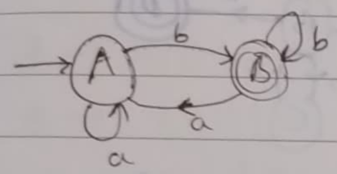


⇒

c)



DFA → RLG

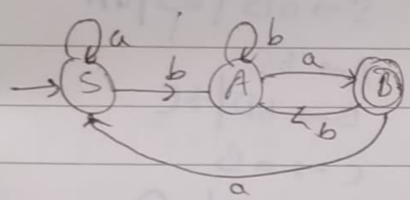
1) 

⇒ starts

$A \rightarrow aA \mid bB$ write all the outgoing transitions

$B \rightarrow bB \mid aA$

$B \rightarrow \epsilon$ ($\because$ B is the final state)

2) 

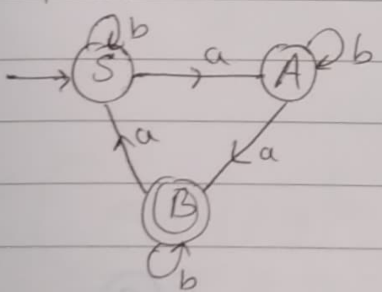
⇒

$S \rightarrow aS \mid bA$

$A \rightarrow bA \mid aB$

$B \rightarrow \epsilon \mid bA \mid aS$

3) $|W_a| \equiv 2 \bmod 3$



⇒

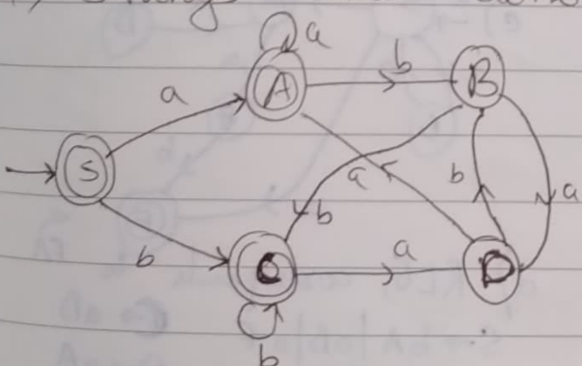
$S \rightarrow bS \mid aA$

$A \rightarrow bA \mid aB$

$B \rightarrow bB \mid aS$

$B \rightarrow \epsilon$

4) Strings that donot end in 'ab' or 'ba'



⇒

$S \rightarrow aA \mid bC \mid \epsilon$

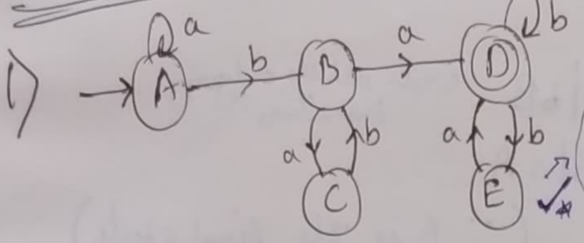
$A \rightarrow aA \mid bB \mid \epsilon$

$C \rightarrow bC \mid aD \mid \epsilon$

$B \rightarrow aD \mid bC$

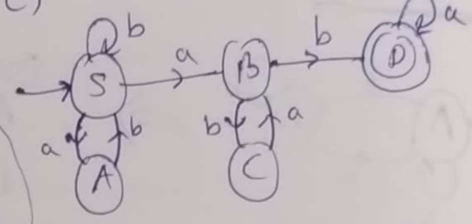
$D \rightarrow aA \mid bB$

FA → LLG

1) 

step a,b,c
direct reverse FA can also be done

c) FA for reversed RE



a) RE : $a^*b(ab)^*a(b+ba)^*$

b) reverse RE : $(ab+b)^*a(ba)^*ba^*$

c) reverse RLG (words of production rules on RHS of RLG)

$S \rightarrow Ba \mid Sb \mid Aa$
$A \rightarrow Sb$
$B \rightarrow Db \mid Cb$
$C \rightarrow Ba$
$D \rightarrow \epsilon \mid Da$

d) RLG for ↑

$S \rightarrow aB \mid bS \mid aA$
$A \rightarrow bS$
$B \rightarrow bD \mid bC$
$C \rightarrow aB$
$D \rightarrow \epsilon \mid aD$

RLG → LLG

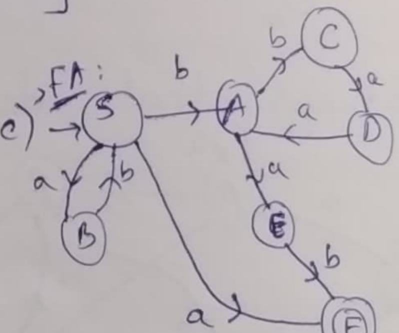
1) $S \rightarrow aA \mid baB$ $[a(ba)^* + ba(aab)^*b]$
$A \rightarrow baA \mid \epsilon \Rightarrow [(ba)^*]$
$B \rightarrow aabB \mid b \Rightarrow [(aab)^*b]$

a) RE :
$a(ba)^* + ba(aab)^*b$

b) Reverse RE :
$b(baa)^*ab + (ab)^*a$

We can also skip step a, b, c by directly constructing FA for grammar & reverse the FA

c) FA :



FA

d) RLG corresponds to FA

$S \rightarrow bA \mid aB \mid aF$
$B \rightarrow bS$
$A \rightarrow aE \mid bC$
$C \rightarrow aD$
$D \rightarrow aA$
$E \rightarrow bF$
$F \rightarrow \epsilon$

e) Reverse the words of production rules on RHS of RLG

$S \rightarrow Ab \mid Ba \mid Fa$
$B \rightarrow Sb$
$A \rightarrow Ea \mid Cb$
$C \rightarrow Da$
$D \rightarrow Aa$
$E \rightarrow Fb$
$F \rightarrow \epsilon$

LLG → RLG

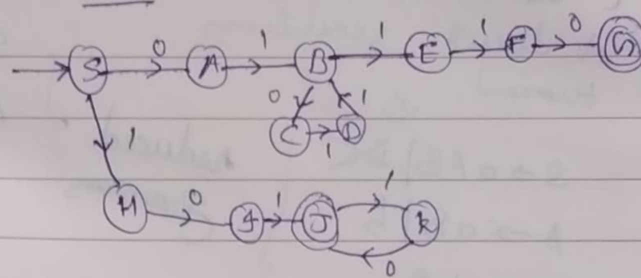
1) $A \to A10 \mid B110 \mid 101$
   $B \to B011 \mid 01$

a) reverse RHS of product rules
   $A \to 01A \mid 011B \mid 101$
   $B \to 110B \mid 10 \Rightarrow (110)^*10$

b) RE:
   $(01)^*101 + 011(110)^*10$

c) reverse RE:
   $01(011)^*110 + 101(10)^*$

d) FA:



e) RLG:

$S \to 0A \mid 1H$ $D \to 1B$ $H \to 0I$
$A \to 1B$ $E \to 1F$ $I \to 1J$
$B \to 0C \mid 1E$ $F \to 0G$ $J \to \epsilon \mid 1K$
$C \to 1D$ $G \to \epsilon$ $K \to 0J$

☆ Context Free Grammars (CFG)

CFG ⇓ CFL → PDA

$A \to \alpha$ (→ string of var. & term) ; $A \in V$, $\alpha \in (V+T)^*$

⇓ single variable

Ex:
CFG: $S \to aSb$ ← (It is not RG as RG doesn't allow aSb it only allows aS as Sa)
$S \to \epsilon$

CFL: $L = \{a^n b^n \mid n \geq 0\}$

* Optimization of CFG:
↳ Elimination of useless symbols/variables & rules

* Useless Symbols/Production

Ex:

1) $S \to aA$
   $A \to b$
   $\times B \to b$
   (We can eliminate $B \to b$ as it isn't derivable from S)

2) $S \to aA \mid bB$
   $A \to b$
   We can remove bB as B doesn't derive a terminal symbol.

3) $S \to aAb \mid Ba$
   $A \to aB \mid ba$
   $B \to aBC$
   $S \to aAb$
   $A \to ba$
   (C is useless variable, remove it. Also B is useless as it doesn't derive any terminal)

* Order of optimization
1) remove ϵ - prod
2) remove unit - prod
3) remove useless prod

4) $S \to aAB \mid BC$
$A \to aB \mid b$
$B \to bA \mid BC$

C can be removed as it doesn't generate any terminal

⇓

$S \to aAB \mid \cancel{BC}$
$A \to aB \mid b$
$B \to bA$

reduced grammar

5) $S \to aAB \mid BaC \mid BC$
$A \to aA \mid Ba \mid AC \mid a$
$B \to BAC \mid BB \mid b$

⇓

$S \to aAB$
$A \to aA \mid Ba \mid a$
$B \to BB \mid b$

* Unit Productions

$A \to B$ can be simplify $A, B \in V$

Ex:

1) $S \to aA$
$A \to B \mid b$
$B \to a$

Here as $A \to B \to a$ we can simplify it as
$S \to aA$
$A \to a \mid b$

2) $S \to AaB$
$A \to B \mid aB \mid a$
$B \to C \mid bB \mid b$

C is useless we can remove it & $A \to B$ is a unit production
$\therefore$ $S \to AaB$
$A \to aB \mid a \mid bB \mid b$
$B \to bB \mid b$

3) $S \to ABa \mid BC$
$A \to aA \mid B \mid a$
$B \to D \mid b$
$D \to b \mid C$

C is useless, so we can drop it & after simplifying unit production

⇓

$S \to ABa$
$A \to aA \mid a \mid b$
$B \to b$
$D \to b$

⇒

$S \to ABa$
$A \to aA \mid a \mid b$
$B \to b$

as D can't be reached / derived from S

* ϵ - productions

$A \to \epsilon$; $A \in V$

Ex:

1) $S \to ABC$
$A \to a$
$B \to \epsilon$
$C \to b$

⇓

$S \to AC$
$A \to a$
$C \to b$

2) $S \to AaB$
$A \to b \mid \epsilon$
$B \to C \mid \epsilon$

C is useless & we remove it

⇓

$S \to AaB$
$A \to b \mid \epsilon$
$B \to \epsilon$

⇓

$S \to Aa$
$A \to b$

3) $S \to AaB$
$A \to b \mid \epsilon$
$B \to c \mid \epsilon$

Substitute ϵ in place of A & B in RHS of S one by one

⇓

$S \to AaB \mid aB \mid Aa \mid a$
$A \to b$
$B \to c$

4) $S \to bAB \mid BaA$
$A \to aA \mid \epsilon \mid AC$
$B \to bB \mid \epsilon \mid BC$

We can remove C as it is useless

Now, we have to eliminate ϵ - productions by substituting $A \to \epsilon$

$S \to bAB \mid BaA \mid bB \mid Ba \mid bA \mid aA \mid b \mid a$

$A \to aA \mid a$
$B \to bB \mid b$

* Normal forms : Standard forms ; Grammar does not generate ϵ

↳ A grammar is in a normal form if its production rules have a special structure.

1) CNF : Chomsky NF
2) GNF : Greibach NF

* CNF : *If a grammar can covert in the following type*

A CFG is in CNF if all production rules satisfy one of the following:

$A \rightarrow BC$ $\quad$ $A, B, C \in V$

$A \rightarrow a$ $\quad$ $a \in T$

Ex: 1) $S \rightarrow aSb \mid ab$

$A \rightarrow a$ $\quad$ $S \rightarrow AB$

$B \rightarrow b$ $\quad$ $C \rightarrow AS$

$S \rightarrow CB$

CNF
ASB
$C \to AS$
$S \to CB$

2) $S \rightarrow aSa \mid bSb \mid \epsilon$

$S' \rightarrow S \cup \{\epsilon\}$ [doesn't CNF]

$A \rightarrow a$ $\quad$ $S \rightarrow CA$

$B \rightarrow b$ $\quad$ $S \rightarrow DB$

$C \rightarrow AS$

$D \rightarrow BS$

CNF

Augmented Grammar

3) $S \rightarrow aAb \mid aB \mid abA$

$A \rightarrow a$ ✓
$B \rightarrow b$ ✓ already in CNF

$S \rightarrow AAB$
$S \rightarrow AB$
$S \rightarrow ABA$

understanding

$\Rightarrow$ $C \rightarrow AB$

$S \rightarrow AC$

$S \rightarrow AB$

$S \rightarrow CA$

$A \rightarrow a$

$B \rightarrow b$

4) $S \rightarrow CB \mid AB$

$C \rightarrow AS$

$B \rightarrow b$

$A \rightarrow a$

In CNF $\quad w_1 = ab \quad\quad w_2 = aabb$

To derive this

$S \rightarrow AB$
$\rightarrow aB$
$\rightarrow ab$

3 derivation

$S \rightarrow CB$
$\rightarrow ASB$
$\rightarrow AABB$
$\rightarrow aABB$
$\rightarrow aaBB$
$\rightarrow aabB$
$\rightarrow aabb$

7 derivation

Gen: string of len n.

↓ CNF

$2n-1$

derivatives of prod. rules

* GNF Greibach NF

↳ $A \to a\alpha$; $A \in V$, $a \in T$, $\alpha \in V^*$

strings of variables

Ex 1) $S \to aSb \mid ab$

$B \to b$
$S \to aB$
$S \to aSB$ } GNF

2) $S \to aSa \mid bSb \mid aa \mid bb$

$A \to a$
$B \to b$
$S \to aA$
$S \to bB$
$S \to aSA$
$S \to bSB$ } GNF

3) $S \to aAb \mid bBa$
$A \to bAa \mid b$
$B \to bBa \mid a$

we have to somehow replace this a with the variable as it appears (A→bAa, B→bBa) the conditions

∴ $C \to a$
$S \to aAD \mid bBC$
$A \to bAC \mid b$
$B \to bBC \mid a$
$D \to b$

4) $S \to Aa \mid bBa$
$A \to bB \mid a$
$B \to bBa \mid b$

$X \to a$
$A \to bB \mid a$
$B \to bBX \mid b$
$S \to bBX \mid aX \mid \cancel{bb}$

⇒ Here A is unreachable

$X \to a$
$B \to bBX \mid b$
$S \to bBX \mid aX$ } GNF

∴ starting var → S

$S \to Aa$ isn't valid
$S \to bBa \mid aa$ "
$S \to bBX \mid aX$ valid

5) $S \to aSb \mid ab$

$B \to b$
$S \to aSB \mid aB$ } GNF

$w_1 = ab$
$S \to aB$
$\to ab$ } 2 derivⁿ

$w_2 = aabb$
$S \to aSB$
$\to aaBB$
$\to aabB$
$\to aabb$ } 4 derivⁿ

* Gen: String of len n ↓GNF n productionⁿ to generate strings

* Decision Properties of CFG (Shows decidability) ↳ ∃ a comp. prog.

1) * Emptiness & Non-emptiness

{ Check whether the lang generated by CFG is an empty lang or non-empty

Ex (1) G: $S \to aA \mid aB$
$A \to aB \mid b$

Non-empty lang

B is useless here as B can't generate any terminal. After removing B, we get a grammar which generate a single string

Step: Construct a reduced grammar first.

If it can form a single string it is non-empty

2) G: $S \to AB \mid aB$
$A \to a \mid b$
$B \to AB$

Empty lang

B doesn't generate any terminal, hence it is useless
But after removing B our start symbol gets removed

If we can create a comp. program to check for the property (of CFG) then it can be enclude in decision propertis

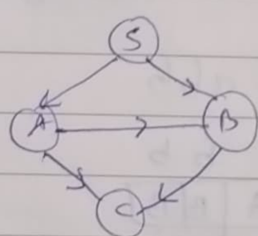
* **Finiteness / Infiniteness** : does the G generate finite lang.
↳ as we have algo. (VDG), it is decidable.

There is a special graph, **Variable Dependency graph** (VDG) in which all the variables (of grammar) are vertices in graph & nodes are connected based on the productⁿ rule. (no need of terminals in a graph).

If graph has <u>loops or cycles</u> ⇒ Infinite lang.
else ⇒ finite lang.

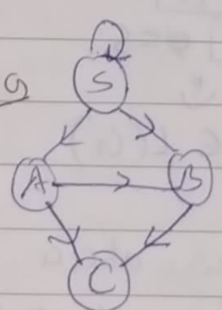
Ex:

1) $S \to AB$
$A \to BC \mid a$
$B \to CC \mid b$
$C \to d$
⇒ VDG



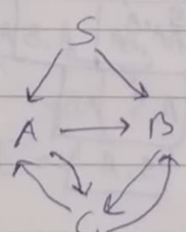
$\xRightarrow{\text{no loops/cycles}}$ Finite lang. (lang gener. by grammar)

2) $S \to SS \mid AB$
$A \to BC \mid a$
$B \to CC \mid a$
$C \to d$
⇒ VDG



⇒ a loop on S ⇒ Infenite lang.

3) $S \to AB$
$A \to BC \mid a$
$B \to CC \mid b$
$C \to AB$
⇒



⇒ ∃ cycles ⇒ Infenite lang

Membership : given a word ω & grammar G, is $\omega \in L(G)$? } → Done using algo. CYK → Kasami, Cocke, Young
condiⁿ: G has to be in CNF
As we have an algorithm (CYK), it $\boxed{\text{is}}$ decidable

CYK is a DP algo. (Dynamic Programming)

Ex: 1) G: $S \to AB$

$A \to BA \mid SA \mid a$

$B \to BB \mid BS \mid b$

CNF (we can apply CYK)

a) $w_1 = ab$

- we have to write var. in cell which can generate the upper terminals.
- ~~Next~~, we have to write var. which can generate ~~the~~ above var. pair

a	b
A	B
S	

As we reached start symbol

$w_1 \in L(G)$

b) $w_2 = ba$

$w_2 \in L(G)$

b	a
B	A
A	

NOT start ← A

As we get a single var. generating pair of var, we can't go further ∵ a var. can't generate single var. in CNF & the last var. we get isn't the start symbol ⇒ $w_2 \notin L(G)$

c) $w_3 = aba$

a	b	a
~~A~~	B	A
A	A	

stucked →

a	b	a
A	B	A
S	A	
A		

→ NOT start

⇓

$w_3 \notin L(G)$

d) $w = bba$

b	b	a
B	B	A
B	A	
A		

$\neq S$

⇓

$w \notin L(G)$

e) $w = abb$

a	b	b
A	B	B
A	B	
S		

$\cancel{\phi} = S$

⇓

$w \in L(G)$

f) $w = aab$

a	a	b
A	A	B
A	S	
~~ϕ~~		

stucked but only possible way

⇓

$w \notin L(G)$

2) G: $S \to AB \mid AA$

$A \to BA \mid SA \mid a \mid b$

$B \to BB \mid BS \mid a$

a) $w = abba$

a	b	b	a
A,B	A	A	A,B
S,A	A	A,B	
A,S	A,B		
S,A			

⇒ one of the possibility = S

$w \in L(G)$

⇒ T

b) $w = abaa$

a	b	a	a
A,B	A	A,B	A,B
S, A, A, BA	A,B	A,B	
A,S	A,B		
S,A			

→ ∃ S ⇒ $w \in L(G)$

we can cross check by going backwards from last cell to top (terminals)