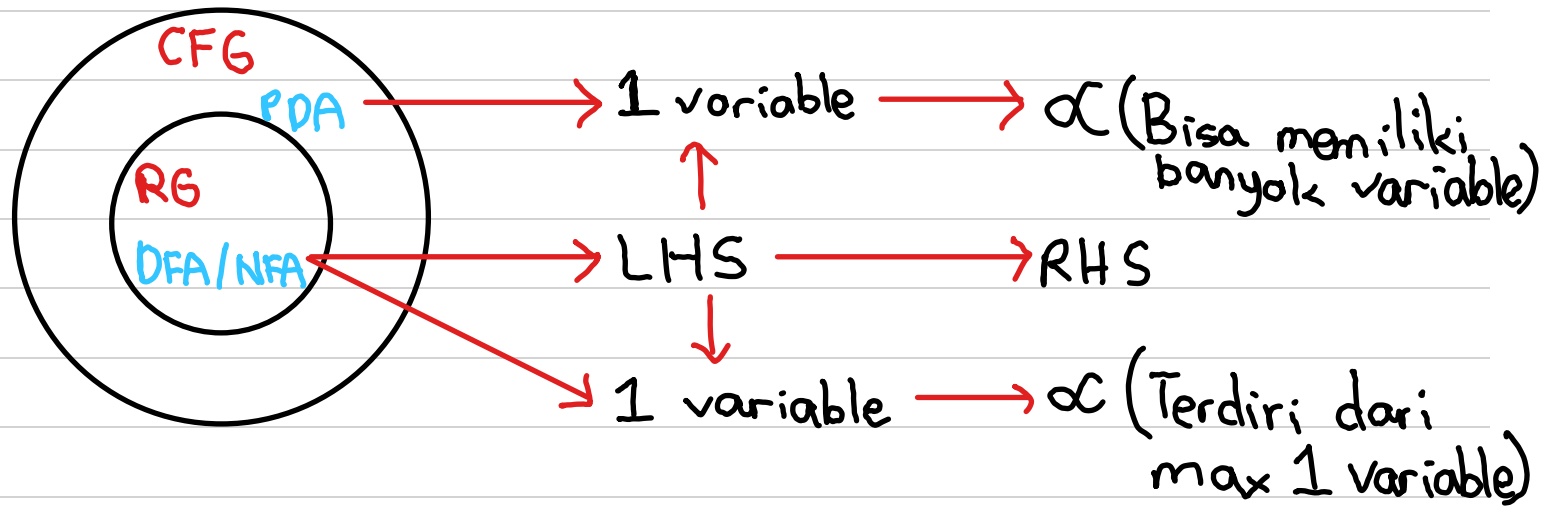
Week 8

- Chomsky Hierarchy :

Grammar Type	Grammar Accepted	Language Accepted	Automata
Type 0	Unrestricted Grammar	Recursively Enumerable Language	Turing Machine
Type 1	Context Sensitive Grammar	Context Sensitive Language	Linear Bounded Automata
Type 2 (Middle Linear Grammar)	Context Free Grammar	Context Free Language	Pushdown Automata
Type 3	Regular Grammar	Regular Language	Finite State Automata

-



- Grammar defined by 4 tuples : $G = (V, T, P, S)$

  ↳ V = Set of variables / non-terminal symbols
  T = Set of terminal symbols
  P = Production rules for terminals and non-terminals
  S = Start symbol

- A production rule has the form $\alpha \rightarrow \beta$ where α and β are strings on $V \cup T$ and at least 1 symbol of α belongs to V.

- Production adalah aturan yang diperbolehkan untuk melakukan substitusi variabel.

- 2 types of RG :

1. Right Linear Grammar

A grammar is said to be right linear if all productions are of the form.

$A \rightarrow xB$

$A \rightarrow x$

where $A, B \in V$ and $x \in T$

$S \rightarrow abS \mid b$

2. Left Linear Grammar

A grammar is said to be left linear if all productions are of the form

$A \rightarrow Bx$

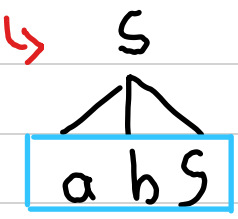
$A \rightarrow x$

where $A, B \in V$ and $x \in T$

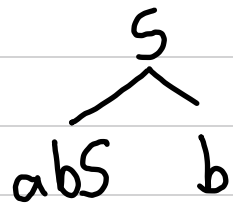
$S \rightarrow Sab \mid b$

- Contoh parse tree / pohon turunan :

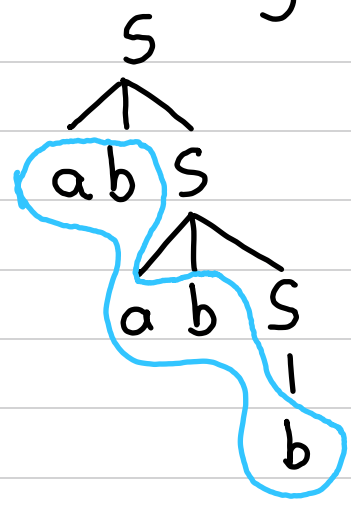
$S \rightarrow abS \mid b$

↳ 

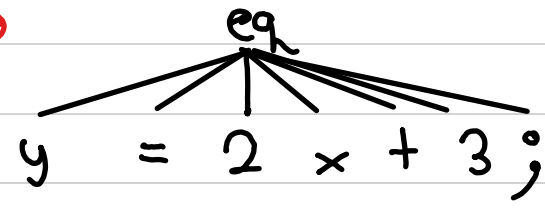
↳ NOT



↳ Contoh string ababb



- equation : $y = 2x + 3;$ →



- RG harus berurutan

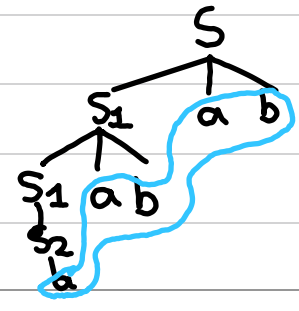
- $G = (\{S, S_1, S_2\}, \{a, b\}, P, S)$

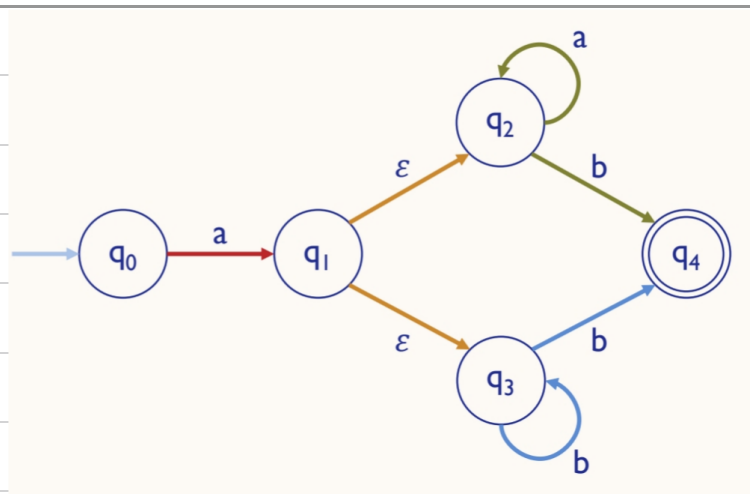
$S \rightarrow S_1 ab$

$S_1 \rightarrow S_1 ab \mid S_2$

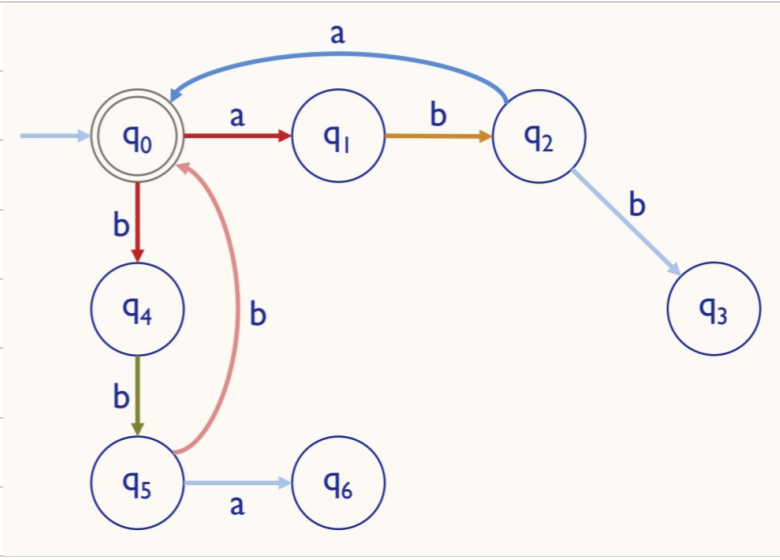
$S_2 \rightarrow a$

aabab ✓
ababa ✗
babab ✗
ababb ✗
babaa ✗





$S \to aA$
$A \to B \mid C$
$B \to aB \mid b$
$C \to bC \mid b$
$G = (\{S, A, B, C\}, \{a, b\}, \{S \to aA, A \to B|C, B \to aB|b, C \to bC|b\}, S)$

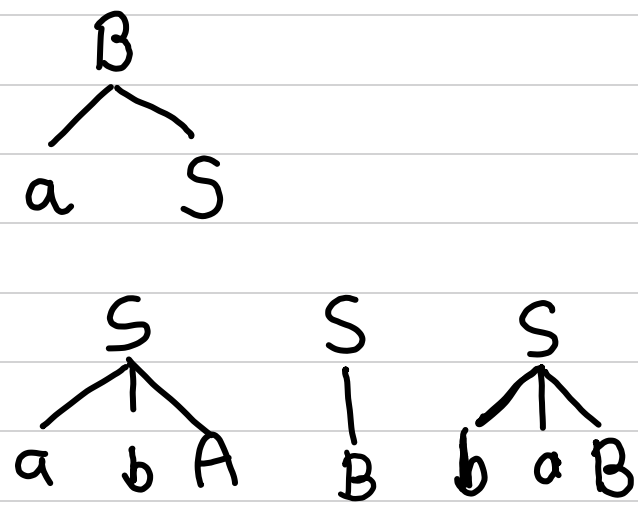


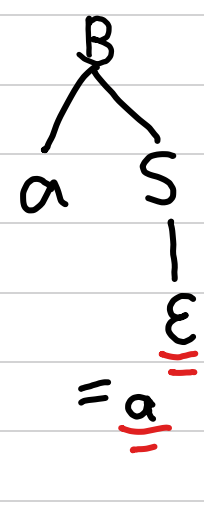
$S \to aA \mid bB \mid \varepsilon$
$A \to bC$
$B \to bD$
$C \to aS$
$D \to bS$

$A \to baS$
$B \to bbS$

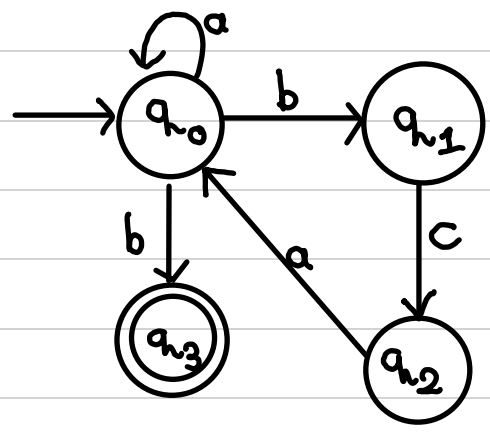
non-ε | ε

$S \to abA \mid B \mid baB \mid \varepsilon$
$A \to bS \mid b$
$B \to aS$

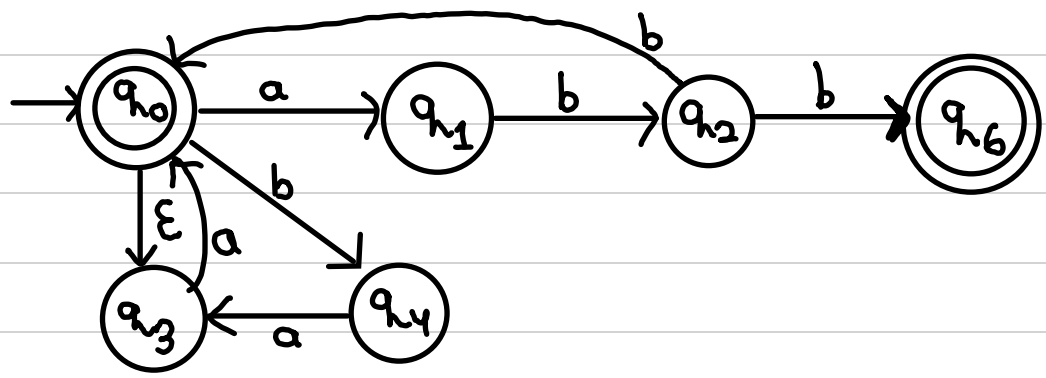




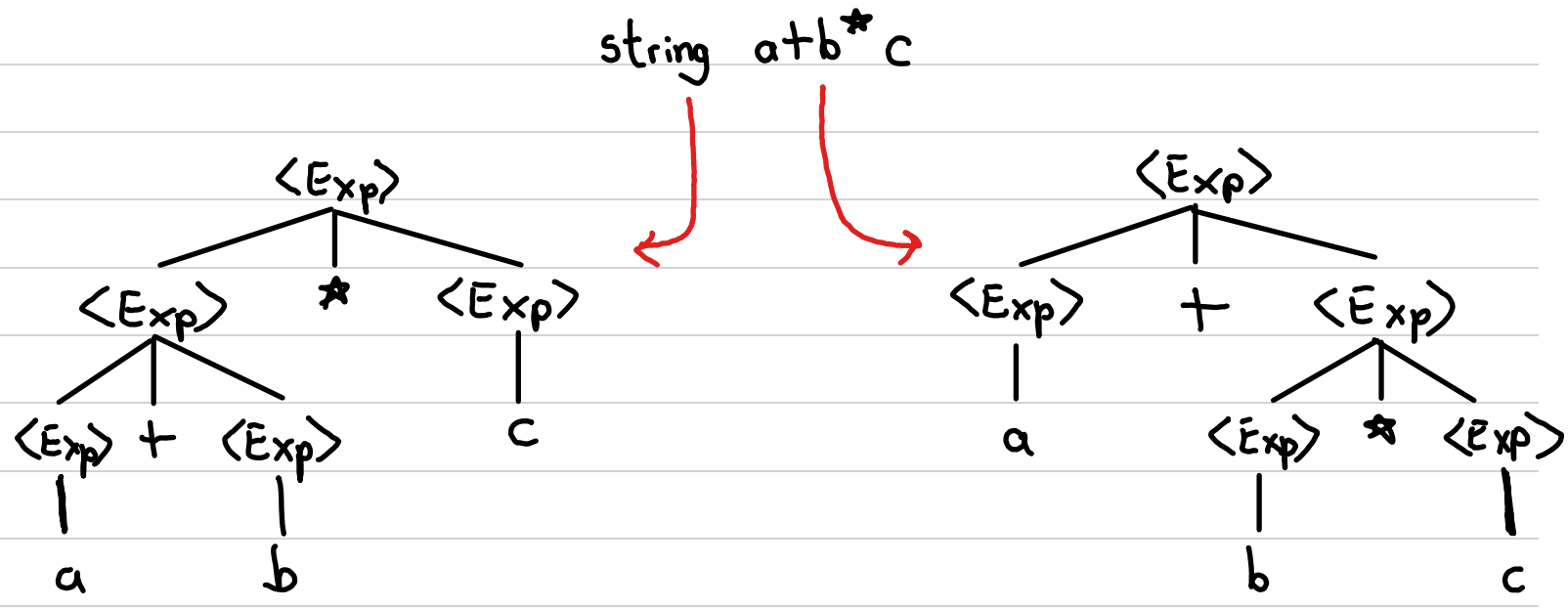
$S \to aS \mid bB \mid b$
$B \to cC$
$C \to aS$



- $S \rightarrow abA \mid B \mid baB \mid \varepsilon$
  $A \rightarrow bS \mid b$
  $B \rightarrow aS$



- 2 terminal yang berpasangan harus ada transisi statenya.
- RL sudah pasti CFG, tetapi sebaliknya belum tentu
- Ambiguous Grammar adalah grammar yang bisa menghasilkan sebuah kalimat yang memiliki lebih dari 1 derivasi/ 1 parse tree.
- Contoh ambiguous grammar :



Week 9

- CFG has production rule of the form $A \to \alpha$ where $\alpha \in (V \cup T)^*$ and $A \in V$
  - e.g. : $B \to CDeFg$
    $D \to BcDe$

- The set of all strings that can be derived from a grammar is said to be the LANGUAGE generated from that grammar.
  - $G_1 = (\{S, A, B\}, \{a, b\}, \{S \to AB, A \to a, B \to b\}, S)$

$$S \Rightarrow AB \Rightarrow ab$$
$$L(G_1) = \{ab\}$$

  - $G_2 = (\{S, A, B\}, \{a, b\}, \{S \to AB, A \to aA \mid a, B \to bB \mid b\}, S)$

$$S \Rightarrow AB \Rightarrow ab$$
$$S \Rightarrow AB \Rightarrow aAbB \Rightarrow aabb$$
$$S \Rightarrow AB \Rightarrow aAb \Rightarrow aab$$
$$S \Rightarrow AB \Rightarrow abB \Rightarrow abb$$
$$L(G_2) = \{ab, a^2b^2, a^2b, ab^2, \dots\} = \{a^m b^n \mid m \geq 1 \wedge n \geq 1\}$$

- Method to find whether a string belongs to a grammar or not:
  1. Start with the S and choose the closest production that matches to the given string
  2. Replace the variables with its most appropriate production. Repeat the process until the string is generated or until no other productions are left.

- Dalam parse tree tidak ada pembagian RMD dan LMD.

- Check grammar can generate string 00110101

$$S \to 0B \mid 1A$$
$$A \to 0 \mid 0S \mid 1AA \mid \varepsilon$$
$$B \to 1 \mid 1S \mid 0BB$$

↳ S

⇒ 0B (S → 0B)

⇒ 00BB (B → 0BB)

⇒ 001B (B → 1)

⇒ 0011S (B → 1S)

⇒ 00110B (S → 0B)

⇒ 001101S (B → 1S)

⇒ 0011010B (S → 0B)

⇒ 00110101 (B → 1)

- **Derivation Tree / Parse Tree** adalah ordered rooted tree that graphically represents the semantic information of strings derived from CFG.

- Parsing left derivation tree : Applying production to the leftmost variable

- Parsing right derivation tree : Applying production to the right most variable

- Contoh parsing :

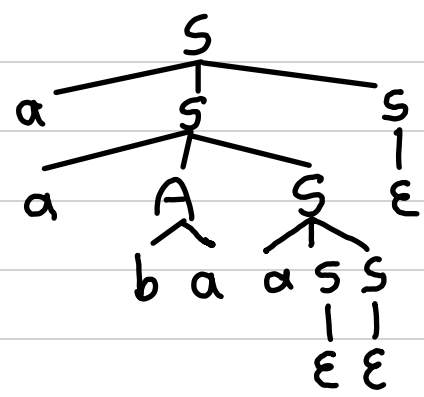
Generating the string aabaa from the grammar

$S \to aAS \mid aSS \mid \varepsilon$

$A \to SbA \mid ba$

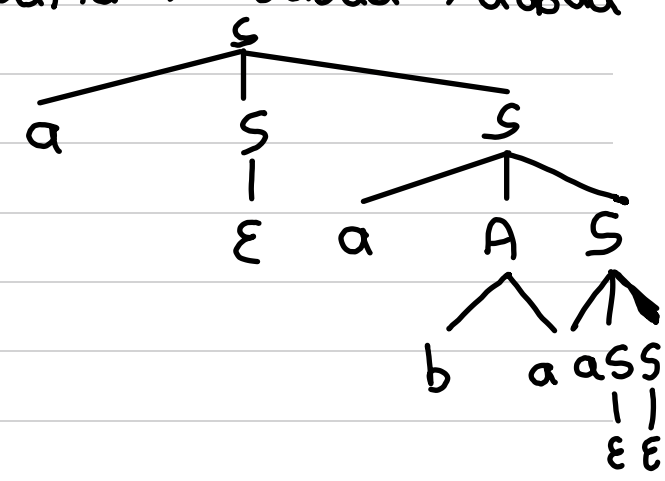
Leftmost

$S \Rightarrow aSS \Rightarrow aaASS \Rightarrow aabaSS \Rightarrow aabaaSS \Rightarrow aabaa$



Rightmost

$S \Rightarrow aSS \Rightarrow aSaAS \Rightarrow aSaAaSS \Rightarrow aSaAa \Rightarrow aSabaa \Rightarrow aabaa$

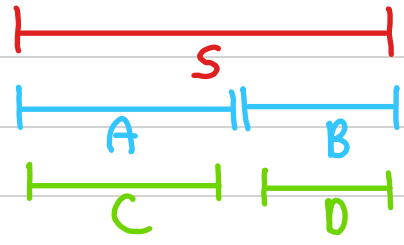


- Conversion of RE to CFG :

RE	CFG
$RE = \varepsilon$	$S \rightarrow \varepsilon$
$RE = \phi$	–
$RE = a$	$S \rightarrow a$
$RE = a+b$	$S \rightarrow a \mid b$
$RE = ab$	$S \rightarrow ab$
$RE = a^*$	$S \rightarrow aS \mid \varepsilon$
$RE = a^+$	$S \rightarrow aS \mid a$
$RE = (a+b)^*$	$S \rightarrow aS \mid bS \mid \varepsilon$
$RE = (a+b)^+$	$S \rightarrow aS \mid bS \mid a \mid b$
$RE = (ab)^*$	$S \rightarrow aA \mid \varepsilon ; A \rightarrow bS$
$RE = (ab)^+$	$S \rightarrow aA ; A \rightarrow bS \mid b$

- Contoh konversi RE ke CFG :

$$\underbrace{(ab+ba)^*}_{A}\underbrace{(abb)^*}_{B}$$



$S \rightarrow AB$
$A \rightarrow CA \mid \varepsilon$
$C \rightarrow ab \mid ba$
$B \rightarrow DB \mid \varepsilon$
$D \rightarrow abb$

$G = (\{S, A, B, C, D\}, \{a, b\}, P, S)$

Week 10

A. Eliminations of Useless Productions

- Useless symbols:
  - Variables/Terminals that do not appear in any derivation of a terminal string from S.
  - Symbols that are not generating
  - Symbols that are not reachable

- Contoh useless productions:

1. $S \to aSa \mid Abd \mid Bde$
   $A \to Ada$
   $B \to BBB \mid a$

   $\Rightarrow$

   $S \to aSa \mid Bde$
   $B \to BBB \mid a$
   Loop di A

2. $S \to Aa \mid B$
   $A \to ab \mid D$
   $B \to b \mid E$
   $C \to bb$
   $E \to aEa$

   $\Rightarrow$

   $S \to Aa \mid B$
   $A \to ab$
   $B \to b$
   D not reachable
   Loop di E
   C no derivation from any V

3. $S \to aAb \mid cEb$
   $A \to dBE \mid eeC$
   $B \to ff$
   $C \to ae$
   $D \to h$

   $\Rightarrow$

   $S \to aAb$
   $A \to eeC$
   $C \to ae$
   E not reachable, makes B useless at $A \to dBE$
   D no derivation from any V

4. $S \to aB$
   $A \to beD \mid dAC$
   $B \to e \mid Ab$
   $C \to bCb \mid adF \mid ab$
   $F \to cFb$

   $\Rightarrow$

   $S \to aB$
   $B \to e$
   $C \to bCb \mid ab$
   D not reachable
   Loop di A, F

5. $S \to aBD$
   $B \to cD \mid Ab$
   $D \to ef$
   $A \to Ed$
   $F \to dc$

   $\Rightarrow$

   $S \to aBD$
   $B \to cD$
   $D \to ef$
   F no derivation from any V
   E not reachable, makes A useless at $B \to Ab$ & $A \to Ed$

B. Elimination of Unit Productions

- Unit productions : Those of the form $A \to B$ for variables A and B.

- Contoh unit productions :

1. $S \to Sb \mid C$
   $C \to D \mid ef$
   $D \to dd$

   $\Rightarrow$

   $S \to Sb \mid dd \mid ef$
   $C \to dd \mid ef$
   $D \to dd$

C. Elimination of Null (ε) Productions

- Null (ε) productions : - Those of the form $A \to \varepsilon$ for some variable A
  - A variable A is nullable if $A \to \varepsilon$
  - If A is nullable, then whenever A appears in a production body, A might derive ε.

- Contoh null (ε) productions :

1. $S \to bcAd$
   $A \to bd \mid \varepsilon$

   $\Rightarrow$

   $S \to bcAd \mid bcd$
   $A \to bd$

2. $S \to A_a CD$
   $A \to CD \mid AB \mid \varepsilon$
   $B \to b \mid \varepsilon$
   $C \to d \mid \varepsilon$
   $D \to \varepsilon$

   $S \to A_a C \mid aC \mid A_a \mid a$
   $A \to C \mid AB \mid A \mid B$

   $\Rightarrow$

   $B \to d$
   $C \to d$

- Simplification of CFG is convert any CFG G into an equivalent CFG that has no useless symbols, unit productions, and ε-productions.

- Safe order of CFG simplification is :

1. Eliminate ε-productions → Muncul unit productions
2. Eliminate unit productions → Muncul useless productions
3. Eliminate useless productions / symbols

- Contoh dengan safe order :

$S \to AA \mid C \mid bd$
$A \to Bb \mid \varepsilon$
$B \to AB \mid d$
$C \to de$

1. Eliminate ε-productions :

$S \to A \mid AA \mid C \mid bd$
$A \to Bb$
$B \to AB \mid B \mid d$
$C \to de$

2. Eliminate unit productions :

$S \to Bb \mid AA \mid de \mid bd$
$A \to Bb$
$B \to AB \mid d$
$C \to de$

3. Eliminate useless productions :

$S \to Bb \mid AA \mid de \mid bd$
$A \to Bb$
$B \to AB \mid d$
C no derivation from any V

Week 11

- Prasyarat CNF / Chomsky Normal Form :

  "Setiap CFL non-ε diberntuk berdasarkan CFG yang semua produksinya adalah dalam bentuk $A \rightarrow BC$ atau $A \rightarrow a$, dimana $A \in V \cap a \in T$. Bentuk ini adalah CNF."

- To put a CFG in CNF, must satisfy :

  - CFG has no ε-productions
  - CFG has no unit productions
  - CFG has no useless productions/symbols

- After satisfy, baru bisa buat CNF hanya 2 bentuk yang diterima yaitu :
  - $A \rightarrow BC$
  - $A \rightarrow a$

- Step of conversion of CFG to CNF :

1. Arrange all bodies of length 2/more consist only of variables
   - ↳ For all T a that appears in a body of length 2/more, create a new variable, A (This variable has only 1 production, $A \rightarrow a$).
   - ↳ Use A in place of a everywhere a appears in a body of length 2/more.

2. Break bodies of length 3/more into a cascade of productions, each with a body of consisting of 2 variables.
   - ↳ Break those productions $A \rightarrow B_1 B_2 \ldots B_k$, for $k \geq 3$, into a group of productions with 2 variables in each body.
   - ↳ Introduce $k-2$ new variables, $C_1, C_2, \ldots, C_{k-2}$
   - ↳ The original production is replaced by the $k-1$ productions

     $A \rightarrow B_1 C_1$

     $C_1 \rightarrow B_2 C_2$

     ....

     $C_{k-3} \rightarrow B_{k-2} C_{k-2}$

     $C_{k-2} \rightarrow B_{k-1} B_k$

Contoh CNF

1. $S \to bA \mid aB$
   $A \to bAA \mid aS \mid a$
   $B \to aBB \mid bS \mid b$

$\Rightarrow$

$S \to P_1A \mid P_2B$
$A \to P_1P_3 \mid P_2S \mid a$
$B \to P_2P_4 \mid P_1S \mid b$
$P_1 \to b$
$P_2 \to a$
$P_3 \to AA$
$P_4 \to BB$

2. $S \to aAB \mid ch \mid CD$
   $A \to dbE \mid eEC$
   $B \to ff \mid DD$
   $C \to ADB \mid aS$
   $D \to i$
   $E \to jD$

$\Rightarrow$

$S \to P_1P_2 \mid P_3P_4 \mid CD$
$A \to P_5P_6 \mid P_8P_9$
$B \to P_{10}P_{10} \mid DD$
$C \to AP_{11} \mid P_1S$
$D \to i$
$E \to P_{12}D$

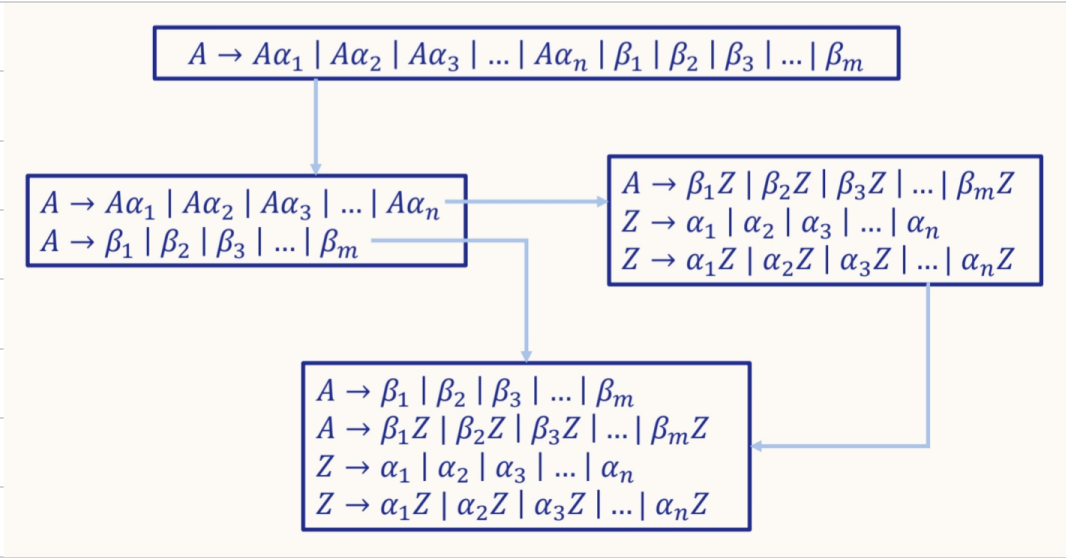
$P_1 \to a$ $\quad$ $P_5 \to d$ $\quad$ $P_9 \to EC$
$P_2 \to AB$ $\quad$ $P_6 \to P_7E$ $\quad$ $P_{10} \to f$
$P_3 \to c$ $\quad$ $P_7 \to b$ $\quad$ $P_{11} \to DB$
$P_4 \to h$ $\quad$ $P_8 \to e$ $\quad$ $P_{12} \to j$

Week 12

- Rekursif di production rules :

1. Left Rerursion : $S \to Sd,\ B \to Bad$
2. Right Recursion : $S \to dS,\ B \to adB$
3. Middle Recursion : $B \to aBd$

-



- Contoh kasar : $A \to \underbrace{ABa}_{\alpha} \mid \underbrace{aA}_{\beta_1} \mid \underbrace{b}_{\beta_2}$

Dengan rumus, $\boxed{A \to \beta_1 \mid \beta_2 \mid \beta_1 Z_1 \mid \beta_2 Z_1}$

maka, $A \to aA \mid b \mid aAZ_1 \mid bZ_1$
$Z_1 \to Ba \mid BaZ_1$

- Contoh elimination of left recursion :

1. $S \to \underbrace{Sab}_{\alpha_1} \mid \underbrace{aSc}_{\beta_1} \mid \underbrace{dd}_{\beta_2} \mid \underbrace{ff}_{\beta_3} \mid \underbrace{Sbd}_{\alpha_2}$
   - $S \to aSc \mid dd \mid ff \mid aScZ_1 \mid ddZ_1 \mid ffZ_1$
     $Z_1 \to ab \mid bd \mid abZ_1 \mid bdZ_1$

2. $S \to \underbrace{Sab}_{\alpha_1} \mid \underbrace{Sb}_{\alpha_2} \mid \underbrace{cA}_{\beta_1}$
   - $S \to cA \mid cAZ_1$
     $Z_1 \to ab \mid b \mid abZ_1 \mid bZ_1$

   $A \to \underbrace{Aa}_{\alpha_1} \mid \underbrace{a}_{\beta_1} \mid \underbrace{bd}_{\beta_2}$
   $A \to aZ_2 \mid bdZ_2 \mid a \mid bd$
   $Z_2 \to a \mid aZ_2$

Week 13

- Bentuk umum GNF / Greibach Normal Form :
  1. $A \to \alpha$ atau $A \to a BCDE_{cccc}$ (Diawali terminal dan diikuti variabel)
  2. $A \to a$ (Hanya 1 terminal)
- Prasyarat GNF adalah harus CNF dan tidak ada rekursif kiri, serta tidak menghasilkan ε.
- Konversi dari CNF ke GNF yaitu :
  1. Tentukan order variable / change the names of the non-terminal symbols into some A_i in ascending order of i.
  2. Alter the rule agar non-terminals berada di ascending order, such that if the production is of the form $A_i \to A_j \gamma$, then $i < j$ dan tidak akan $i \geq j$.
     - ↳ $i > j$: Substitusi
     - ↳ $i \leq j$: Remove left recursion
- Contoh GNF :

1. $S \to CA$ | $A \to a \mid d$ | $B \to b$ | $C \to DD$ | $D \to AB$ — Order : $S < A < B < C < D$ → $S \to aBDA \mid dBDA$ | $A \to a \mid d$ | $B \to b$ | $C \to aBD \mid dBD$ | $D \to aB \mid dB$
2. $A \to BC$ | $B \to CA \mid b$ | $C \to AB \mid a$ — Order : $A < B < C$

$C \to BCB \mid a \Rightarrow C \to CACB \mid bCB \mid a$ (α = $CACB$; β_1 = bCB; β_2 = a)

$Z_1 \to ACB \mid ACBZ_1$

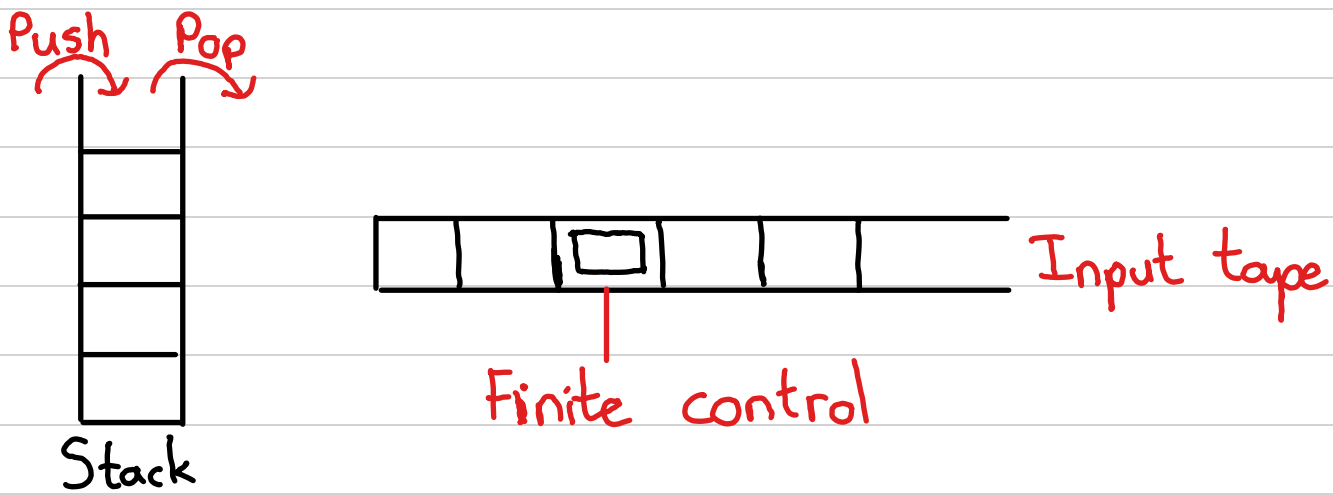
$C \to bCB \mid bCBZ_1 \mid a \mid aZ_1$

$Z_1 \to BCCB \mid BCCBZ_1 \Rightarrow Z_1 \to CACCB \mid CACCBZ_1 \mid bCCB \mid bCCBZ_1$

$Z_1 \to bCBACCB \mid bCBZ_1ACCB \mid aACCB \mid aZ_1ACCB \mid$

$bCCB \mid bCCBZ_1 \mid bCBACCBZ_1 \mid bCBZ_1ACCBZ_1 \mid aACCBZ_1 \mid aZ_1ACCBZ_1$

Week 14

- PDA adalah FA "Counter part" dari CFG
- Finite automation yang terdiri dari :
  1. Finite control
  2. Input tape
  3. Stack
- 

- PDA dapat mengenal language yang CFL seperti :

  $L = \{ wcw^R \mid w \text{ dalam } (0+1)^* \}$

  ↳ CFG yang sesuai untuk language tersebut yaitu :

  $G = (\{S\}, \{0,1,c\}, P, S)$ dengan produksi P : $S \to 0S0 \mid 1S1 \mid c$
- Konstruksi dan mekanisme kerja PDA yang menerima $L = \{ wcw^R$ dalam $w \in (0+1)^* \}$:

  (w ↑ w^R ↑)
  - FC 2 state : q_1, q_2
  - Stack symbol : Piring biru (B), hijau (H), merah (M)
  - Input symbol : 0, c, 1
- Mekanisme kerja :
  1. Saat awal, isi stack = M, start state = q_1
  2. Untuk input dalam w dan current state = q_1 :
     - Jika input = 0 → (Next state : q_1 & Stack : Push B)

a) Jika input = 1 → (Next state : q_1 & Stack : Push H)

3. Input = c & Current state = q1 → (Next state : q_2 & Stack : No operation)

4. Untuk input dalam w^R & Current state = q2 :
   a) Jika input = 0 & top stack = B → (Next state : q_2 & Stack : Pop B)
   a) Jika input = 1 & top stack = H → (Next state : q_2 & Stack : Pop H)

5. Setelah input w^R selesai, maka input = ε, top stack = M, & current state = q2 → (Next state : q2 & Stack : Pop M) agar empty stack

6. Di luar ketentuan di atas, PDA tidak bergerak

- Mekanismenya dalam bentuk tabel :

Piring	State	Input		
		0	1	c
	q_1	Push B tetap q_1	Push H tetap q_1	masuk q_2
Biru	q_2	Pop top tetap q_1	-	-
	q_1	Push B tetap q_1	Push H tetap q_1	masuk q_2
Hijau	q_2	-	Pop top tetap q_2	-
	q_1	Push B tetap q_1	Push H tetap q_1	masuk q_2
Merah	q_2	Pop top elemen dari stack		

- Defo formal PDA (7 tuple) : $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$,

dengan
Q = State
Σ = Alphabet
Γ = Alphabet dari stack
δ = Transisi
q_0 = Start state
Z_0 = Start dari tumpukan stack
F = Sisa tumpukan stack

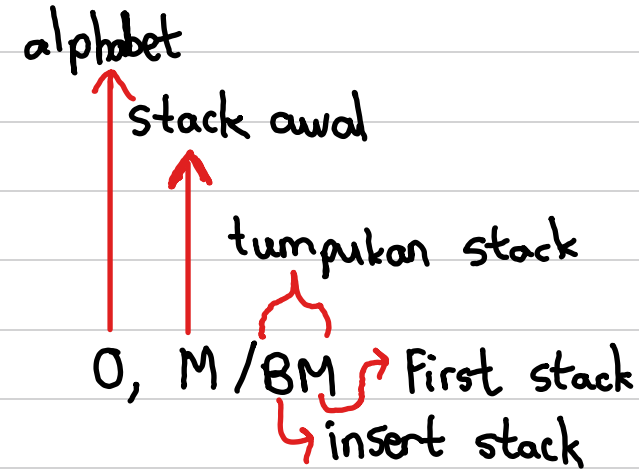
- PDA menerima language selain string habis terbaca dengan 2 cara yaitu :

1. Stack menjadi kosong, dinyatakan dengan $F = \phi$
2. FA masuk final state, dinyatakan dengan $F = q_i$, dimana $i = 0, 1, 2, \ldots$

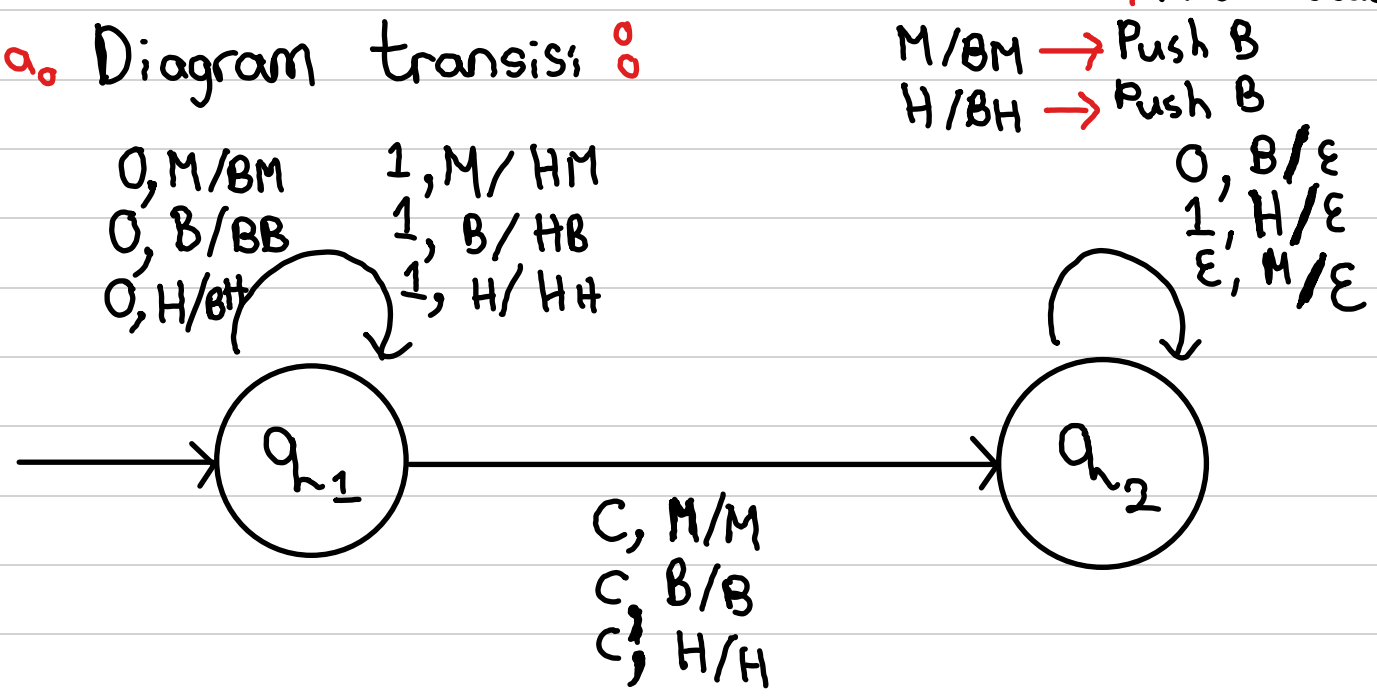
- Contoh PDA : PDA untuk menerima language : $L = \{wcw^R \mid w \text{ dalam } (0+1)^*\}$ dengan empty stack yaitu :

$M = (\{q_1, q_2\}, \{0, 1, c\}, \{M, B, H\}, \delta, q_1, M, \emptyset)$

1. $\delta(q_1, 0, M) = (q_1, BM)$
2. $\delta(q_1, 0, B) = (q_1, BB)$
3. $\delta(q_1, 0, H) = (q_1, BH)$
4. $\delta(q_1, c, M) = (q_2, M)$
5. $\delta(q_1, c, B) = (q_2, B)$
6. $\delta(q_1, c, H) = (q_2, H)$
7. $\delta(q_2, 0, B) = (q_2, \in)$
8. $\delta(q_2, \in, M) = (q_2, \in)$
9. $\delta(q_1, 1, M) = (q_1, HM)$
10. $\delta(q_1, 1, B) = (q_1, HB)$
11. $\delta(q_1, 1, H) = (q_1, HH)$
12. $\delta(q_2, 1, H) = (q_2, \in)$



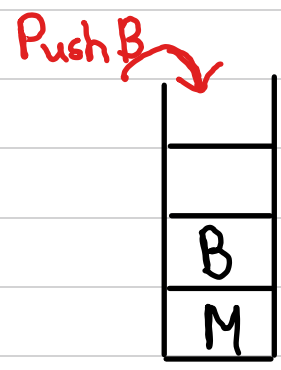
- Diagram transisi :



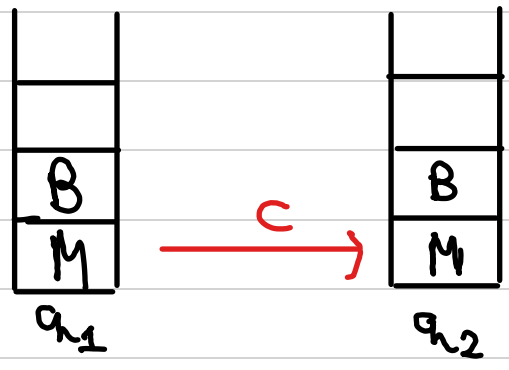
b. Instantaneous Description (ID) dari string "001c100"

$(q_1, 001c100, M) \rightarrow (q_1, 01c100, BM) \rightarrow$
$(q_1, 1c100, BBM) \rightarrow (q_1, c100, HBBM) \rightarrow$
$(q_2, 100, HBBM) \rightarrow (q_2, 00, BBM) \rightarrow$
$(q_2, 0, BM) \rightarrow (q_2, \varepsilon, M) \rightarrow (q_2, \varepsilon, \varepsilon) \rightarrow$
Accepted karena string habis terbaca dan stack menjadi kosong.

- $f(x, y, z)$:- $\delta(q_1, 0, M) = (q_1, BM)$



- $\delta(q_1, c, B) = (q_2, B)$



- $\delta(q_2, \varepsilon, M) = (q_2, \varepsilon)$