

## WEEK 10: LECTURE NOTES

### Normal Forms for CFGs

$$G = (V, T, P, S)$$

- productions in  $G$  satisfy certain restrictions

$$A \rightarrow \alpha, A \in V, \alpha \in (VUT)^*$$

Two of them

- └ Chomsky Normal Form (CNF)
- └ Greibach Normal Form (GNF)

Chomsky Normal Form (Restrictions on the length of RHS and nature of symbols in RHS of production).

A CFG  $G$  is in CNF, if every production is of the form  $A \rightarrow a$  or  $A \rightarrow BC$  and  $S \rightarrow \epsilon$  is in  $G$  if  $\epsilon \in L(G)$

When  $\epsilon \in L(G)$  we assume that  $S$  does not appear on the R.H.S. of any production

Example:  $S \rightarrow AB|\epsilon, A \rightarrow a, B \rightarrow b$

- For a CFG in CNF, the parse tree has the following property :- Every node has at most two descendants, either two internal vertices or a single leaf

### Reduction to CNF

Example:  $S \rightarrow ABC|aC, A \rightarrow a, B \rightarrow b, C \rightarrow c$

- $S \rightarrow ABC$ , is not in the required form  
Replace them by  $S \rightarrow AE, E \rightarrow BC$

- $S \rightarrow aC$  is not in the required form  
Replace them by  $S \rightarrow DC, D \rightarrow a$

### Equivalent CFG

$$\begin{aligned} S &\rightarrow AE, E \rightarrow BC, S \rightarrow DC \\ D &\rightarrow a, A \rightarrow a, B \rightarrow b \\ C &\rightarrow c \end{aligned}$$

## Theorem (Reduction to CNF)

For every CFG, there is an equivalent grammar  $G_2$  in CNF

Proof:

Step 1: Elimination of null productions and unit productions

↓  
(except for  $S \rightarrow \epsilon$ , if  $\epsilon \in L(G)$  and  
 $S$  does not appear on R.H.S. of any  
productions)

Let  $G = (V, T, P, S)$  be the resulting CFG

Step 2: Elimination of terminals on R.H.S.

We define  $G_1 = (V', T, P', S')$ , where  $V'$  and  $S'$  are constructed as follows:

- All productions of the form  $A \rightarrow a$  or  $A \rightarrow BC$  are included in  $P'$ . All variables in  $V$  are included in  $V'$ .
- Consider  $A \rightarrow x_1 x_2 \dots x_n$  with some terminal in R.H.S.. If  $x_i$  is a terminal, say  $a_i$ , add new variable  $c_{a_i}$  to  $V'$  and  $c_{a_i} \rightarrow a_i$  in  $P'$ .

In production  $A \rightarrow x_1 x_2 \dots x_n$ , every terminal on the R.H.S. is replaced by the corresponding new variable and the variables on the R.H.S. are retained. The resulting production is added to  $P'$ .

Thus we get  $G_1 = (V', T, P', S')$

**Step 2 : Restricting the number of variables on R.H.S.**

for any production in  $P'$ , the R.H.S. consists of either a single terminal (or  $\epsilon$ , if  $S \rightarrow \epsilon$ ) or two or more variables.

Define  $G_2 = (V'', T, P'', S)$  as follows:

i. All productions in  $P'$  are added to  $P''$  if they are in the required form. All variables in  $V'$  are added to  $V''$ .

ii. Consider  $A \rightarrow A_1 A_2 \dots A_n$  when  $n > 3$ ,  $A_i \in V'$ .

We introduce new productions  $A \rightarrow A_1 C_1, C_1 \rightarrow A_2 C_2, \dots$

$C_{n-2} \rightarrow A_{n-1} A_n$  and new variables  $C_1, C_2, \dots, C_{n-2}$

These are added to  $P''$  and  $V''$  respectively.

Then  $G_2 = (V'', T, P'', S)$  is the required CNF.

**Example:** Reduce the following grammar  $G$  to CNF.

$G$  is  $S \rightarrow aAD, A \rightarrow aB \mid bAB, B \rightarrow b, D \rightarrow d$

**Sol:**

**Step 1:** No null productions or unit productions

**Step 2:** i.  $B \rightarrow b, D \rightarrow d$  are included in  $P'$

ii.  $S \rightarrow aAD$  gives rise to  $S \rightarrow C_a AD, C_a \rightarrow a$   
 $A \rightarrow aB$  gives rise to  $A \rightarrow C_a B,$

$A \rightarrow bAB$  gives rise to  $A \rightarrow C_b AB, C_b \rightarrow b$

$\therefore G' = (V', T, P', S), V' = \{S, A, B, D, C_a, C_b\}$

$P' = \{S \rightarrow C_a AD, A \rightarrow C_a B \mid C_b AB, B \rightarrow b, D \rightarrow d,$   
 $C_a \rightarrow a, C_b \rightarrow b\}$

**Step 3:**

i.  $A \rightarrow C_a B, B \rightarrow b, D \rightarrow d, C_a \rightarrow a, C_b \rightarrow b$ , are included  
in  $P_2$

- ii.  $S \rightarrow C_0 AD$  is replaced by  $S \rightarrow C_0 C_1, C_1 \rightarrow AD$   
 $A \rightarrow C_b AB$  is replaced by  $A \rightarrow C_b C_2, C_2 \rightarrow AB$

$$G_2 = \left( \{S, A, B, D, C_a, C_b, C_1, C_2\}, \{a, b, d\}, P_2, S \right)$$

where  $P_2 = \{ S \rightarrow C_0 C_1, A \rightarrow C_0 B \mid C_b C_2, C_1 \rightarrow AD, C_2 \rightarrow AB, B \rightarrow b, D \rightarrow d, C_a \rightarrow a, C_b \rightarrow b \}$

$G_2$  is in CNF and is equivalent to  $G$ .

Step 4: Claim:  $L(G) = L(G_2)$

To complete the proof we have to show that

$$L(G) = L(G_1) = L(G_2)$$

To prove  $L(G) \subseteq L(G_1)$

Let  $w \in L(G)$ . If  $A \rightarrow x_1 x_2 \dots x_n$  is used in the derivation of  $w$ , the same effect can be achieved by using the corresponding productions in  $P'$  and the productions in the new variables.

Hence,  $A \xrightarrow[G_1]{*} x_1 x_2 \dots x_n$ .

Thus,  $L(G) \subseteq L(G_1)$

To prove  $L(G_1) \subseteq L(G)$

Let  $w \in L(G_1)$ .

To show  $w \in L(G)$ , it is enough to prove

$$A \xrightarrow[G]{*} w \text{ if } A \in V, A \xrightarrow[G_1]{*} w \quad \text{--- (1)}$$

We prove it by induction on the number of steps in

$$A \xrightarrow{*_{G_1}} w$$

Base:

if  $A \xrightarrow{G_1} w$  then  $A \rightarrow w$  is in  $P'$

By construction of  $P'$ ,  $w$  is a single terminal.

So,  $A \rightarrow w$  in  $P$  i.e.  $A \xrightarrow{G} w$

Induction step:

Let us assume ① for derivations in at most  $K$  steps.

Let.  $A \xrightarrow{K+1}_{G_1} w$

We can split this derivation as

$$A \xrightarrow{G_1} A_1 A_2 \dots A_n \xrightarrow{K}_{G_1} w_1 \dots w_n = w$$

s.t.  $A_i \xrightarrow{*_{G_1}} w_i$

Now each  $A_i$  is i. either in  $V$ , or

ii. a new variable, say  $Cai$

i. when  $A_i \in V$ , then  $A_i \xrightarrow{*_{G_1}} w_i$  is a derivation in at most  $K$  steps, so by induction hypothesis.

$$A_i \xrightarrow{*_{G_1}} w_i$$

ii. when  $A_i = Cai$ , the production  $Cai \rightarrow ai$  is applied to get  $A_i \xrightarrow{*_{G_1}} w_i$

The production  $A \rightarrow A_1 A_2 \dots A_n$  in  $P'$  is induced by a production  $A \rightarrow x_1 x_2 \dots x_n$  in  $P$

where  $\begin{cases} x_i = A_i & \text{if } A_i \in V \\ x_i = w_i & \text{if } A_i = c_{ai} \end{cases}$

So,  $A \xrightarrow[G]{} x_1 x_2 \dots x_n \xrightarrow[G]{*} w_1 w_2 \dots w_n$

i.e.  $A \xrightarrow[G]{*} w$

Thus ① is true for all derivations

$$\therefore L(G) = L(G_1)$$

To prove  $L(G_1) = L(G_2)$

The effect of applying  $A \rightarrow A_1 A_2 \dots A_m$  in a derivation for  $w \in L(G_1)$  can be achieved by applying the productions

$$A \rightarrow A_1 C_1, C_1 \rightarrow A_2 C_2, \dots, C_{m-1} \rightarrow A_{m-1} A_m$$

Hence it is easy to say that  $L(G_1) \subseteq L(G_2)$

To prove  $L(G_2) \subseteq L(G_1)$ , we can prove an auxiliary result

$$A \xrightarrow[G_1]{*} w \text{ if } A \in V', A \xrightarrow[G_2]{*} w \quad \text{--- (2)}$$

(2) can be proved by induction on the number of steps in  $A \xrightarrow[G_2]{*} w$

Applying (1) to 2, we get

$$L(G_2) \subseteq L(G_1).$$

Example: Find a grammar in CNF equivalent to

$$S \rightarrow aAbB, A \rightarrow aA|a, B \rightarrow bB|b$$

Soln:

- No unit production, no null productions
- eliminate terminals on R.H.S.

Let  $G_1 = (V_1, \{a, b\}, P_1, S)$  where  $V_1$  and  $P_1$  are constructed as follows:

- i.  $A \rightarrow a, B \rightarrow b$  are added to  $P_1$ ,
- ii.  $S \rightarrow aAbB, A \rightarrow aA, B \rightarrow bB$  yields

$$S \rightarrow C_a A C_b B, A \rightarrow C_a A, B \rightarrow C_b B, C_a \rightarrow a, C_b \rightarrow b$$

which are added to  $P_1$

$$V_1 = \{S, A, B, C_a, C_b\}$$

- Restrict the number of variables on R.H.S.

$P_1$  consists of  $S \rightarrow C_a A C_b B, A \rightarrow C_a A, B \rightarrow C_b B, C_a \rightarrow a,$

$$C_b \rightarrow b, A \rightarrow a, B \rightarrow b$$

Let  $G_2 = (V_2, \{a, b\}, P_2, S)$  where  $P_2$  and  $V_2$  are constructed as:

- i.  $S \rightarrow C_a A C_b B$  is replaced by

$$S \rightarrow C_a C_1, C_1 \rightarrow A C_2, C_2 \rightarrow C_b B$$

which are added to  $P_2$ .

- ii. Remaining productions in  $P_1$  are added to  $P_2$

$$V_2 = \{S, A, B, C_a, C_b, C_1, C_2\}$$

Example: Find a grammar in CNF equivalent to the grammar  $s \rightarrow \sim s \mid [s \triangleright s] \mid p \mid q$   
 (  $s$  being the only variable )

Soln:

- No unit productions, no null productions
- Eliminate terminals on R.H.S.

Let  $G_1 = (V_1, T, P_1, S)$ , where  $P_1, V_1$  are constructed as follows:

- i.  $s \rightarrow p, s \rightarrow q$  are added to  $P_1$ ,
- ii.  $s \rightarrow \sim s$  induced  $s \rightarrow As, A \rightarrow \sim$ , which are added to  $P_1$ ,
- iii.  $s \rightarrow [s \triangleright s]$  induces

$s \rightarrow BSCSD, B \rightarrow [, C \rightarrow \triangleright, D \rightarrow ]$

which are added to  $P_1$ .

- Restrict the number of variables on R.H.S.

Let  $G_2 = (V_2, T, P_2, S)$  where  $P_2$  and  $V_2$  are constructed as

- $P_2$  consists of

i.  $s \rightarrow p \mid q \mid As, A \rightarrow \sim, B \rightarrow [, C \rightarrow \triangleright, D \rightarrow ]$

ii.  $s \rightarrow BSCSD$  is replaced by

$s \rightarrow BC_1, C_1 \rightarrow SC_2, C_2 \rightarrow CC_3, C_3 \rightarrow SD$  which are added to  $P_2$

$$V_2 = \{S, A, B, C, D, C_1, C_2, C_3\}$$

$\therefore G_2$  is in CNF and is equivalent to the given grammar

## Greibach Normal Form (GNF)

Useful in some proofs and constructions

(e.g. CFG generated by pushdown automata is in GNF)

### Definition:

A CFG  $G = (V, T, P, S)$  is in GNF if every production is of the form  $A \rightarrow a\alpha$ , where  $a \in T$  and  $\alpha \in V^*$  ( $\alpha$  may be  $\epsilon$ ) and  $S \rightarrow \epsilon$  is in  $G$  if  $\epsilon \in L(G)$

When  $\epsilon \in L(G)$ , we assume that  $S$  does not appear on the R.H.S. of any production.

e.g.  $S \rightarrow aAB | \epsilon, A \rightarrow bC, B \rightarrow b, C \rightarrow c$  is in GNF.

### Lemma 1

Let  $G = (V, T, P, S)$  be a CFG. Let  $A \rightarrow \alpha, B \alpha_2$  be an  $A$ -production in  $P$  and  $B \rightarrow \beta_1 | \beta_2 | \dots | \beta_r$  be the set of all  $B$ -productions.

Let  $G_1 = (V, T, P_1, S)$  be obtained from  $G$  by deleting the production  $A \rightarrow \alpha, B \alpha_2$  from  $P_1$  and adding the production

$$A \rightarrow \alpha, \beta_1 \alpha_2 | \alpha, \beta_2 \alpha_2 | \dots | \alpha, \beta_r \alpha_2$$

Then,

$$L(G) = L(G_1)$$

Proof:

To prove  $L(G) \subseteq L(G_1)$

Let  $w \in L(G)$

If we apply  $A \rightarrow \alpha, B\alpha_2$  in the derivation for  $w \in L(G)$ , then we have to apply  $B \rightarrow \beta_i$  for some  $i$  at a later step.

Also note that  $A \rightarrow \alpha, B\alpha_2$  is the only production in  $G$  not in  $G_1$ .

The effect of applying  $A \rightarrow \alpha, B\alpha_2$  and eliminating  $B$  in grammar  $G$  is the same as applying  $A \rightarrow \alpha_i \beta_i \alpha_2$  for some  $i$  in  $G_1$ ,

Hence  $w \in L(G_1)$

To prove  $L(G_1) \subseteq L(G)$

Instead of applying  $A \rightarrow \alpha, B\alpha_2$  in a derivation of  $G_1$ , we can apply  $A \rightarrow \alpha, B\alpha_2$  and  $B \rightarrow \beta_i$  in  $G$  to get

$$A \xrightarrow[G]{*} \alpha, B\beta_i\alpha_2$$

Thus if  $w \in L(G_1)$  and  $A \rightarrow \alpha, B\beta_i\alpha_2$  is used for the derivative of  $w$ , then by replacing the single  $A \rightarrow \alpha, B\beta_i\alpha_2$  by two steps  $A \rightarrow \alpha, B\alpha_2$  and  $B \rightarrow \beta_i$  in the derivative of  $w$ , we obtain  $w \in L(G)$

Hence  $L(G_1) \subseteq L(G)$

## Lemma 2

Let  $G = (V, T, P, S)$  be a CFG. Let the set of  $A$ -productions be

$$A \rightarrow A\alpha_1 | A\alpha_2 | \dots | A\alpha_r | B_1 | B_2 | \dots | B_s$$

Let  $z$  be a new variable.

Let  $G_1 = (V \cup \{z\}, T, P_1, S)$ , where  $P_1$  is defined as follows:

i. The set of  $A$ -productions in  $P_1$  are

$$A \rightarrow B_1 | B_2 | \dots | B_s$$

$$A \rightarrow B_1 z | B_2 z | \dots | B_s z$$

ii. The set of  $z$ -productions in  $P_1$  are

$$z \rightarrow \alpha_1 | \alpha_2 | \dots | \alpha_r$$

$$z \rightarrow \alpha_1 z | \alpha_2 z | \dots | \alpha_r z$$

iii. The productions for the other variables are as in  $G$ .

Then  $G_1$  is a CFG and equivalent to  $G$

$$\text{i.e. } L(G_1) = L(G)$$

Proof:

In a leftmost derivation, a sequence of productions of the form  $A \rightarrow A\alpha_i$  must eventually end with a production  $A \rightarrow \beta_i$

The sequence of replacements in  $G$

$$A \xrightarrow{G} A\alpha_{i_1} \xrightarrow{G} A\alpha_{i_2}\alpha_{i_1} \xrightarrow{G} \dots \xrightarrow{G} A\alpha_{i_p}\alpha_{i_{p-1}}\dots\alpha_{i_1}$$

$$\dots \xrightarrow{G} \beta_j \alpha_{i_p} \alpha_{i_{p-1}} \dots \alpha_{i_1}$$

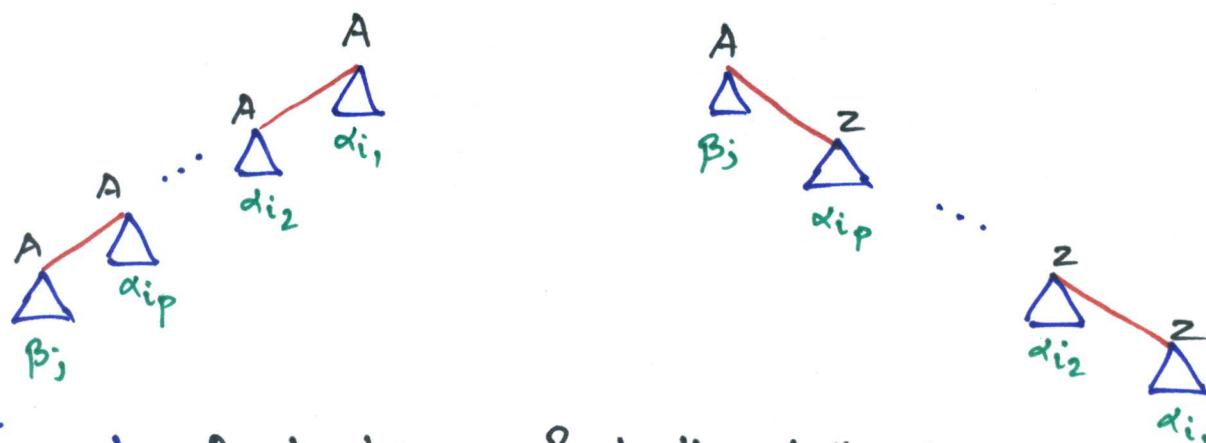
can be replaced in  $G_1$ , by

$$A \Rightarrow_{G_1} \beta_j z \Rightarrow_{G_1} \beta_j \alpha_i p z \Rightarrow_{G_1} \beta_j \alpha_i p \alpha_{i-1} z$$

$$\dots \Rightarrow_{G_1} \beta_j \alpha_i p \alpha_{i-1} \dots \alpha_{i_2} z \Rightarrow_{G_1} \beta_j \alpha_i p \alpha_{i-1} \dots \alpha_{i_2} \alpha_{i_1}$$

The reverse transformation can be also made.

Thus  $L(G) = L(G_1)$



Example: Apply Lemma 2 to the following

A-productivity in a CFG  $G$ .

$$A \rightarrow \underbrace{\alpha \beta D}_{{\beta}_1} \mid \underbrace{b D \beta}_{{\beta}_2} \mid \underbrace{c}_{{\beta}_3} \quad A \rightarrow \underbrace{A \beta}_{{\alpha}_1} \mid \underbrace{A D}_{{\alpha}_2}$$

Soln:

$$G = (V, T, P, S) \rightarrow G' = (V', T, P', S), V' = V \cup \{z\}$$

$P'$ :

$$\begin{aligned} i. \quad & A \rightarrow \alpha \beta D \mid b D \beta \mid c \\ & A \rightarrow \alpha \beta D z \mid b D \beta z \mid c z \end{aligned}$$

$$ii. \quad z \rightarrow \beta \mid D$$

$$z \rightarrow \beta z \mid D z$$

A-productions

$$A \rightarrow \beta_1 \mid \beta_2 \mid \beta_3$$

$$A \rightarrow \beta_{i,2} \mid \beta_{i,2} z \mid \beta_{i,3} z$$

z-productions

$$z \rightarrow \alpha_1 \mid \alpha_2$$

$$z \rightarrow \alpha_{i,1} z \mid \alpha_{i,2} z$$

## Theorem (Greibach Normal Form or GNF)

Every CFL  $L$  without can be generated by a CFG for which each production is of the form  $A \rightarrow \alpha\alpha$ , where  $A$  is a terminal and  $\alpha$  is a (possibly empty) string of variables.

Proof: (Construction of a CFG in GNF)

Step 1: Eliminate null productions and then construct a grammar  $G$  in CNF generating  $L$

We rename the variables as  $A_1, A_2, \dots, A_n$  with  $S = A_1$ ,

Let  $G = (\{A_1, A_2, \dots, A_n\}, T, P, A_1)$

Step 2: (Modify the productions so that if  $A_i \rightarrow A_j$  is a production, then  $j > i$ )

Starting with  $S$  and proceeding to  $A_k$  we do this as follows:

\* { Assume productions have been modified so that for  $1 \leq i \leq k$ ,  $A_i \rightarrow A_j$  is a production only if  $j > i$

We now modify the  $A_k$ -productions (Apply lemma 1)

- If  $A_k \rightarrow A_j$  is a production with  $j < k$ , then we generate a new set of productions by substituting for  $A_j$  the R.H.S. of each  $A_j$ -productions according to Lemma 1.

i.e. we remove  $A_k \rightarrow A_j$  and add productions  $A_k \rightarrow \beta$  for all  $A_j$ -productions  $A_j \rightarrow \beta$  as  $j < k$ ,  $A_j \rightarrow \beta$  satisfies \*

Repeating the process  $k-1$  times at most, we obtain productions of the form  $A_k \rightarrow A_k f$ ,  $j > k$

### Step 3: (Apply lemma 2)

- The production with  $j=k$  are then replaced according to Lemma 2, introducing a new variable  $z_k$

i.e. we remove each  $A_k \rightarrow A_k f$ , and productions

$$z_k \rightarrow f, z_k \rightarrow fz_k$$

and for each  $A_k \rightarrow \beta$ , where  $\beta$  does not start with  $A_k$ , we add production  $A_k \rightarrow \beta z_k$

- By repeating the above process for each original variable, we have only productions of the form:

i.  $A_i \rightarrow A_j f$ ,  $j > i$

ii.  $A_i \rightarrow af$ ,  $a \in T$

iii.  $Z_i \rightarrow f$ ,  $f \in (VU\{z_1, \dots, z_{i-1}\})^*$

Any  $A_n$ -production is of the form  $A_n \rightarrow a$

where  $a \in T$  as  $A_n$  is the highest numbered variable

### Step 4: (Modify $A_i$ -productions to the form $A_i \rightarrow af$ for $i = 1, 2, \dots, n-1$ using lemma 1.)

- $A_n$ -productions are of the form  $A_n \rightarrow af$
- $A_{n-1}$ -productions are of the form ' $A_{n-1} \rightarrow af'$ ' or  $A_{n-1} \rightarrow Any'$

Eliminate production  $A_{n-1} \rightarrow A_n f'$  by using lemma 1.

The resulting productions are in the required form.

- Repeat the construction by considering  $A_{n-2}, A_{n-3}, \dots, A_1$

Step 5: (Modify  $Z_i$ -productions using lemma 1)

- $Z_i$ -productions are of the form  $Z_i \rightarrow a\beta$  or  $Z_i \rightarrow A_K \gamma$  for some  $K$ .

Remark :

It is not necessary to convert all the productions to the form required for CNF in the 1<sup>st</sup> step.

We may allow productions of the form

- $A \rightarrow a\alpha$  where  $a \in T, \alpha \in V^*$
- $A \rightarrow \alpha$  where  $\alpha \in V^*$  with  $|\alpha| \geq 2$

Remark

A CFG in GNF generates a CFL without  $\epsilon$

Example:

Convert the grammar  $S \rightarrow AB, A \rightarrow BS|b, B \rightarrow SA|a$  into GNF.

Soln:

Step 1: No null productions, CNF

rename variable:  $A_1 = S, A_2 = A, A_3 = B$

productions are:  $A_1 \rightarrow A_2 A_3, A_2 \rightarrow A_3 A_1 | b, A_3 \rightarrow A_1 A_2 | a$

Step 2:

i.  $A_1$  productions  $A_1 \rightarrow A_2 A_3$  is in required form

ii.  $A_2$  productions  $A_2 \rightarrow A_3 A_1 | b$  are in the required form

iii.  $A_3 \rightarrow a$  is in required form

Apply lemma 1 to  $A_3 \rightarrow A_1 A_2$ :

replace  $A_3 \rightarrow A_1 A_2$  by the following productions

$A_3 \rightarrow A_2 A_3 A_2$  as  $A_1 \rightarrow A_2 A_3$  is an  $A_1$ -productions

Apply lemma 1 again to  $A_3 \rightarrow A_2 A_3 A_2$

Resulting productions are  $A_3 \rightarrow A_3 A_1 A_3 A_2 | b A_3 A_2$

Thus  $A_3$ -productions are

$A_3 \rightarrow a | A_3 A_1 A_3 A_2 | b A_3$

Step 3: Apply Lemma 2 to  $A_3 \rightarrow A_3 A_1 A_3 A_2$

introduce a new variable  $Z_3$

the resulting productions are:

$A_3 \rightarrow a | b A_3 A_2 | a Z_3 | b A_3, A_2 Z_3$

$Z_3 \rightarrow A_1 A_3 A_2 | A_1 A_3 A_2 Z_3$

$(A_3 \rightarrow \beta_1 | \beta_2 | \beta_3 | \beta_2 Z_3, Z_3 \rightarrow \alpha_1 | \alpha_1 Z_3)$

#### Step 4:

i.  $A_3$ -productions are

$$A_3 \rightarrow a|bA_3A_2|az_3|bA_3A_2z_3$$

ii.  $A_2$ -productions are

$$A_2 \rightarrow A_3A_1|b$$

• We retain  $A_2 \rightarrow b$

• We replace  $A_2 \rightarrow A_3A_1$  using lemma 1.

The resulting productions are

$$A_2 \rightarrow aA_1|bA_3A_2A_1|az_3A_1|bA_3A_2z_3A_1$$

The set of all modified  $A_2$ -productions is

$$A_2 \rightarrow b|aA_1|bA_3A_2A_1|az_3A_1|bA_3A_2z_3A_1$$

iii.  $A_1$ -productions are  $A_1 \rightarrow A_2A_3$

Apply lemma 1 to  $A_1 \rightarrow A_2A_3$  to get

$$A_1 \rightarrow bA_3|aA_1A_3|bA_3A_2A_1A_3|az_3A_1A_3|$$

$$bA_3A_2z_3A_1A_3$$

#### Step 5:

The  $z_3$ -productions are

$$z_3 \rightarrow A_1A_3z_2|A_1A_3A_2z_3$$

Apply Lemma 1 to modify  $z_3$ -productions

The resulting  $z_3$ -productions are

$$z_3 \rightarrow bA_3A_3A_2|aA_1A_3A_3A_2|bA_3A_2A_1A_3A_3A_2|$$

$$az_3A_1A_3A_3A_2|bA_3A_2z_3A_1A_3A_2$$

Example:

Construct a CFG in GNF equivalent to the CFG

$$S \rightarrow AA|a, A \rightarrow SS|b$$

Sol:

Step 1: No null productions, the CFG is in CNF.

Rename the variables:  $A_1 = S, A_2 = A$

So, the productions are:  $A_1 \rightarrow A_2 A_2 | a, A_2 \rightarrow A_1 A_1 | b$

Step 2:

1.  $A_1$ -productions are in the required form. They are  $A_1 \rightarrow A_2 A_2 | a$

2.  $A_2 \rightarrow b$  is in the required form.

Apply lemma 1 to  $A_2 \rightarrow A_1 A_1$

As  $A_1 \rightarrow A_2 A_2 | a$  are  $A_1$ -productions, we replace  $A_2 \rightarrow A_1 A_1$  by the following productions:

$$A_2 \rightarrow A_2 A_2 A_1 | a A_1$$

Thus  $A_2$ -productions are  $A_2 \rightarrow b | A_2 A_2 A_1 | a A_1$

Step 3:

Apply Lemma 2 to  $A_2 \rightarrow A_2 A_2 A_1$

• Introduce a new variable  $z_2$

•  $A_2 \rightarrow A_2 A_2 A_1$  is replaced by  $A_2 \rightarrow b z_2 | a A_1 z_2$

$$z_2 \rightarrow A_2 A_1 | A_2 A_1 z_2$$

#### Step 4:

•  $A_2$ -productions are

$$A_2 \rightarrow b | aA_1 | bz_2 | aA_1z_2$$

—①

$A_1$ -productions are

$$A_1 \rightarrow A_2A_2 | a$$

We retain  $A_1 \rightarrow a$

We replace  $A_1 \rightarrow A_2A_2$  using Lemma 1 and get the following productions

$$A_1 \rightarrow bA_2 | aA_1A_2 | bz_2A_2 | aA_1z_2A_2$$

The set of all modified  $A_1$ -productions are

$$A_1 \rightarrow a | bA_2 | aA_1A_2 | bz_2A_2 | aA_1z_2A_2$$

—②

#### Step 5: Modify $Z_2$ -productions using Lemma 1

$Z_2$ -productions  $Z_2 \rightarrow A_2A_1 | A_2A_1z_2$  are replaced by the following productions

$$Z_2 \rightarrow bA_1 | aA_1A_1 | bz_2A_1 | aA_1z_2A_1$$

$$Z_2 \rightarrow bA_1z_2 | aA_1A_1z_2 | bz_2A_1z_2 | aA_1z_2A_1z_2$$

—③

Hence the equivalent grammar is

$$G' = (\{A_1, A_2, Z_2\}, \{a, b\}, P, A_1)$$

where  $P$ , consists of  $A_1$ -productions,  $A_2$ -productions and  $Z_2$ -productions given by ①, ②, ③ respectively

## Pushdown Automata (PDA)

regular expression  $\leftrightarrow$  FA (DFA)

content-free grammar  $\leftrightarrow$  PDA (non-deterministic device)

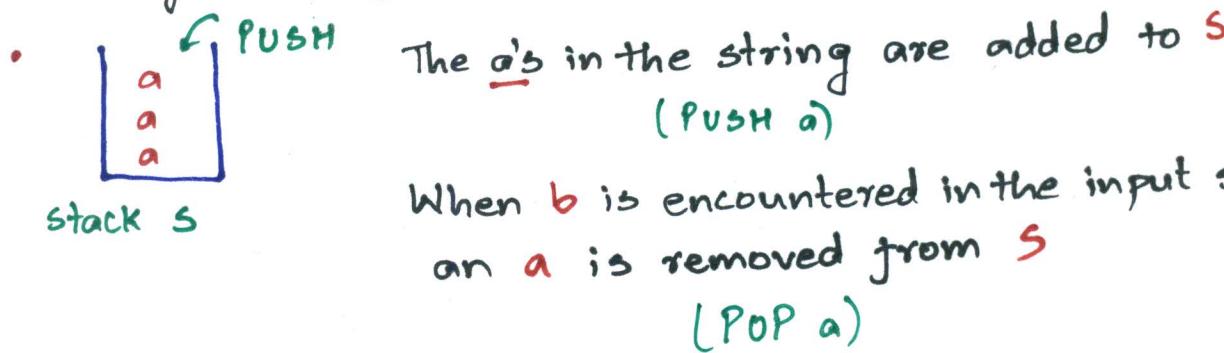
\* Deterministic version accepts only a subset of all CFLs

Example:

$L = \{a^n b^n | n \geq 1\}$  is not a regular (can be shown using Pumping Lemma), but is a CFL as the CFG

$S \rightarrow aSb \mid ab$  generates  $L$

- A FA cannot accept  $L = \{a^n b^n | n \geq 1\}$  as it has to remember number of  $a$ 's in the string and so an infinite number of states
- This difficulty can be avoided by adding an auxiliary memory in the form of a stack



Thus matching of the numbers of  $a$ 's and of  $b$ 's is accomplished

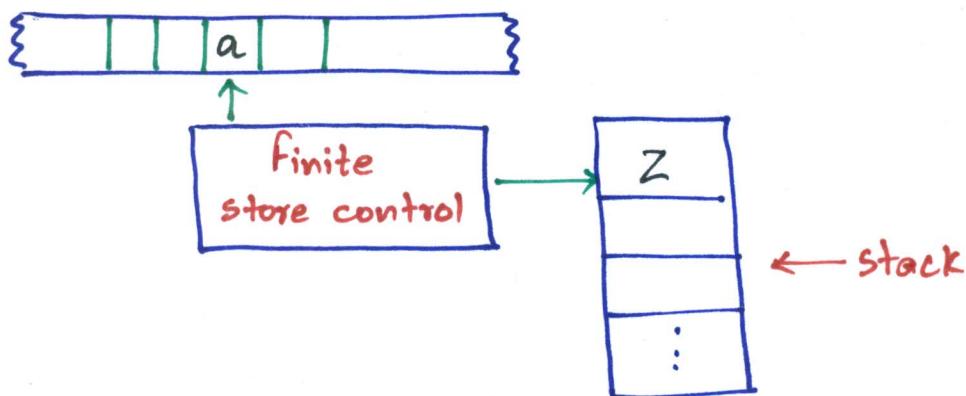
- PDA is essentially a FA with control of both an input tape and a stack.

## Definition (PDA)

A pushdown automata  $M$  is a system  $(Q, \Sigma, \Gamma, \delta, q_0, z_0, F)$  where

1.  $Q$  is a finite set of states
2.  $\Sigma$  is an alphabet, called the input alphabet
3.  $\Gamma$  is an alphabet, called the stack alphabet
4.  $q_0$  in  $Q$  is the initial state
5.  $z_0$  in  $\Gamma$  is a particular stack symbol called the start symbol.
6.  $F \subseteq Q$  is the set of final states
7.  $\delta$  is a mapping from  $Q \times (\Sigma \cup \{\epsilon\}) \times \Gamma$  to a finite subset of  $Q \times M^*$

## PDA:



Pushdown store (PDS)

- lower case letters near the front of alphabet
  - ↳ input symbol
- lower case letters near the end of alphabet
  - ↳ strings of input symbols
- Capital letters → stack symbol
- Greek letters → string of stack symbols

## Moves

- $\delta(q, a, z) = \{(p_1, r_1), (p_2, r_2) \dots (p_m, r_m)\}$

where  $q$  and  $p_i$ ,  $1 \leq i \leq m$  are states  $a \in \Sigma$ ,  $z$  is a stack symbol and  $r_i \in M^*$ ,  $1 \leq i \leq m$

- Interpretation of  $\delta(q, a, z)$

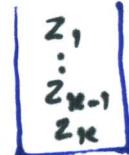
- PDA is state  $q$ , with input symbol  $a$  and  $z$  is the top symbol on the stack.

- For any  $r_i$ , enter state  $p_i$ ,  
replace symbol  $z$  by  $r_i$ ,  
and advance the input head one symbol

- Convention

- leftmost symbol of  $r_i$  will be placed highest on the stack and the rightmost symbol lowest on the stack

i.e. if  $r_i = z_1 z_2 \dots z_k$  then



- Interpretation of  $\delta(q, \epsilon, z) = \{(p_1, r_1) \dots (p_m, r_m)\}$

- PDA in state  $q$ , independent of the input symbol being scanned and with the top symbol on the stack,

can enter state  $p_i$  and

replace  $z$  by  $r_i$  for any

- In this case input head is not advanced.

### Example:

Let  $M = (\mathcal{Q}, \Sigma, \Gamma, \delta, q_0, z_0, F)$  where

$$\mathcal{Q} = \{q_0, q_1, q_f\}, \quad \Sigma = \{a, b\}, \quad \Gamma = \{a, z_0\}$$

$$F = \{q_f\}$$

and  $\delta$  is given by

$$\delta(q_0, a, z_0) = \{(q_0, a, z_0)\}$$

$$\delta(q_1, b, a) = \{(q_1, \epsilon)\}$$

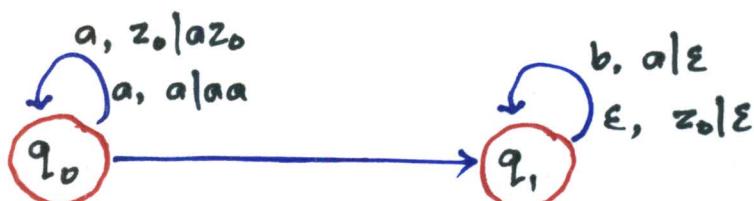
$$\delta(q_0, a, a) = \{(q_0, aa)\}$$

$$\delta(q_1, \epsilon, z_0) = \{(q_1, \epsilon)\}$$

$$\delta(q_0, b, a) = \{(q_1, \epsilon)\}$$

- PDA cannot take transition when PDS is empty.  
In this case the PDA halts.

- The behaviour of PDA is non-deterministic as the transition given by any element of  $\delta(q, a, z)$



## Instantaneous Description (ID)

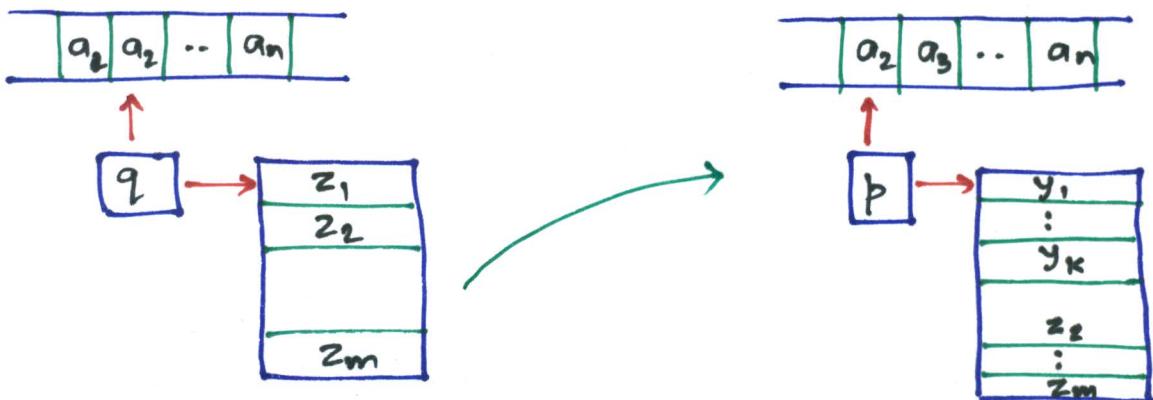
Let  $M = (\Delta, \Sigma, \Gamma, \delta, q_0, z_0, F)$  be a PDA

An instantaneous description (ID) is a tuple  $(q, w, r)$  where  $q$  is a state,  $w$  is a string of input symbols and  $r$  is a string of stack symbols.

Move relation  $\xrightarrow{M}$

$(q, aw, za) \xrightarrow{M} (p, wp\beta) \text{ if } \delta(q, a, z) \text{ contains } (p, \beta)$   
 $(a \text{ may be } \epsilon \text{ or an input symbol})$

- if  $w = a_1 a_2 \dots a_n$ ,  $\alpha = z_1 \dots z_m$ ,  $\beta = y_1 y_2 \dots y_k$



An illustration of the move relation

Relation  $\xrightarrow{M}^*$  (Reflexive and transitive closure of  $\xrightarrow{M}$ )

- $I \xrightarrow{M}^* I$  for each ID  $I$
- $I \xrightarrow{M}^* J$  and  $J \xrightarrow{M}^* K$  imply  $I \xrightarrow{M}^* K$

Note:  $I \xrightarrow{i} K$  if ID  $I$  can become  $K$  after exactly  $i$  steps

Note:  $\xrightarrow{M}$ ,  $\xrightarrow{M}^*$ ,  $\xrightarrow{i} M$  subscript  $M$  is dropped if PDA  $M$  is understood

## Accepted Languages

- For a PDA  $M = (\Delta, \Sigma, \Gamma, \delta, q_0, z_0, F)$  we define  $L(M)$  the language accepted by the final state as

$$\{ w \mid (q_0, w, z_0) \xrightarrow{*} (p, \epsilon, r) \text{ for some } p \text{ in } F \text{ and } r \text{ in } \Gamma^* \}$$

- We define  $N(M)$ , the language accepted by empty stack (null stack) to be

$$\{ w \mid (q_0, w, z_0) \xrightarrow{*} (p, \epsilon, \epsilon) \text{ for some } p \text{ in } \Delta \}$$

- When acceptance is by null stack, the set of final states is irrelevant, and in this case, we usually let the set of final states be the empty set, i.e.  $F = \emptyset$

- $(q_1, x, \alpha) \xrightarrow{*} (q_2, \epsilon, \beta)$  iff for every  $y \in \Sigma^*$

$$(q_1, xy, \alpha) \xrightarrow{*} (q_2, y, \beta)$$

- If  $(q, x, \alpha) \xrightarrow{*} (q', \epsilon, r)$  then for every

$$\beta \in \Gamma^*$$

$$(q, x, \alpha\beta) \xrightarrow{*} (q', \epsilon, r\beta)$$

- Converse is not true

### Example:

The following PDA accepts  $\{ww^R \mid w \text{ in } (0+1)^*\}$

$$M = (\{q_1, q_2\}, \{0, 1\}, \{R, B, G\}, \delta, q_1, R, \epsilon)$$

$$R_1: \delta(q_1, 0, R) = \{(q_1, BR)\}$$

$$R_2: \delta(q_1, 1, R) = \{(q_1, GR)\}$$

$$R_3: \delta(q_1, 0, B) = \{(q_1, BD), (q_2, \epsilon)\}$$

$$R_4: \delta(q_1, 0, G) = \{(q_1, BG)\}$$

$$R_5: \delta(q_1, 1, B) = \{(q_1, GB)\}$$

$$R_6: \delta(q_1, 1, G) = \{(q_1, G, G), (q_2, \epsilon)\}$$

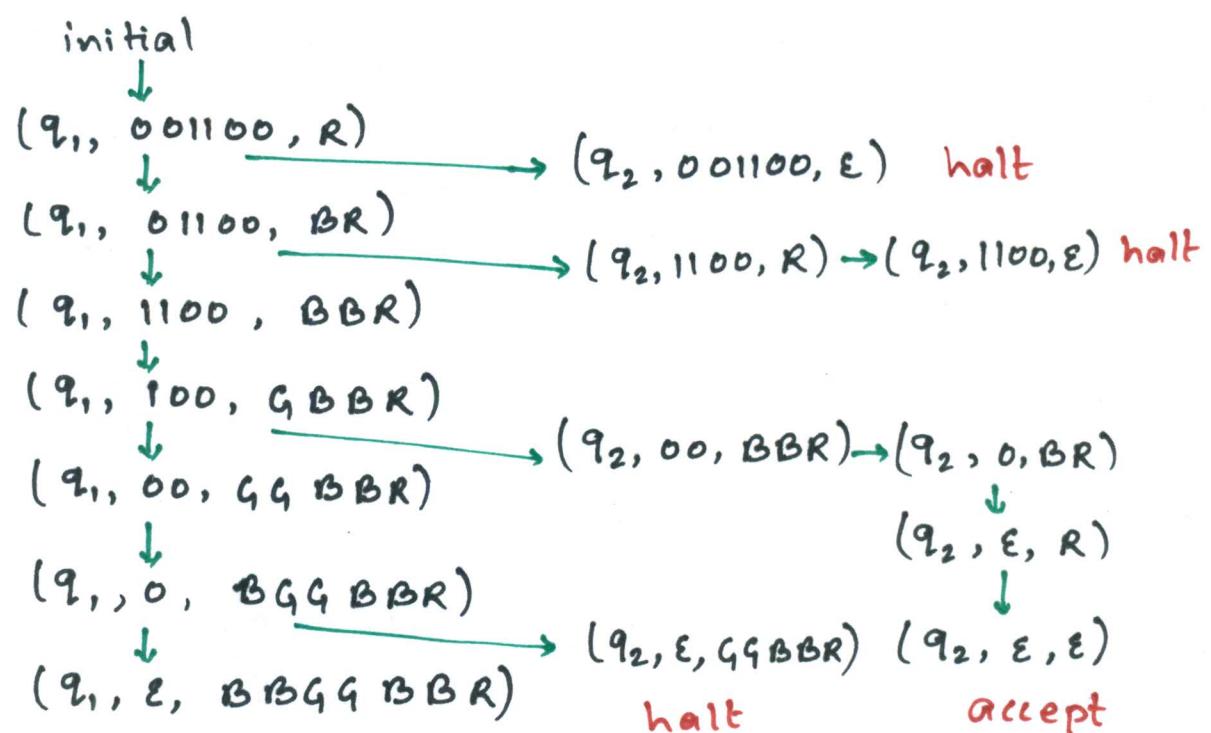
$$R_7: \delta(q_2, 0, B) = \{(q_2, \epsilon)\}$$

$$R_8: \delta(q_2, 1, G) = \{(q_2, \epsilon)\}$$

$$R_9: \delta(q_2, \epsilon, R) = \{(q_2, \epsilon)\}$$

$$R_{10}: \delta(q_2, \epsilon, R) = \{(q_2, \epsilon)\}$$

- $R_1 - R_6$  allow to store the input on the stack



Example:

Construct a PDA M accepting the set of all strings over  $\{a, b\}$  with equal number of a's and b's

Soln:

Let  $M = (\{q\}, \{a, b\}, \{z_0, ab\}, S, q, z_0, \phi)$

where  $\delta$  is defined by the following rules

$$\delta(q, a, z_0) = \{(q, az_0)\}$$

$$\delta(q, b, z_0) = \{(q, bz_0)\}$$

$$\delta(q, a, a) = \{(q, aa)\}$$

$$\delta(q, b, b) = \{(q, bb)\}$$

$$\delta(q, \overset{\text{new incoming symbol}}{a}, \overset{\text{stack top symbol}}{b}) = \{(q, \epsilon)\} \quad (b \text{ is erased in PDS})$$

$$\delta(q, b, a) = \{(q, \epsilon)\} \quad (a \text{ is erased in PDS})$$

$$\delta(q, \epsilon, z_0) = \{(q, \epsilon)\}$$

$$(q, w, z_0) \xrightarrow{*} (q, \epsilon, z_0) \xrightarrow{} (q, \epsilon, \epsilon)$$

if  $w$  has equal number of a's and b's

Hence  $w \in \underline{N(M)}$

### Example 1:

$$M = (\{q_0\}, \{a, b\}, \{z_0\}, \delta, q_0, z_0, \phi)$$

where  $\delta(q_0, a, z_0) = \{(q_0, \epsilon)\}$ ,  $\delta(q_0, b, z_0) = \{(q_0, z_0 z_0)\}$

$$(q_0, aab, z_0 z_0 z_0 z_0) \xrightarrow{} (q_0, ab, z_0 z_0 z_0)$$

$$\xrightarrow{} (q_0, b, z_0 z_0)$$

$$\xrightarrow{} (q_0, \epsilon, z_0 z_0 z_0)$$

But

$$(q_0, aab, z_0) \xrightarrow{} (q_0, ab, \epsilon)$$

Thus,

$$(q_0, aab, z_0 z_0 z_0 z_0) \xrightarrow{*} (q_0, \epsilon, z_0 z_0 z_0)$$

$$\text{but } (q_0, aab, z_0 z_0 z_0) \not\xrightarrow{*} (q_0, \epsilon, z_0 z_0)$$

$$\xrightarrow{} (q_0, ab, z_0 z_0)$$

~~↑~~  
~~ε~~

### Example 2:

$$M = (\{q_0, q_1, q_f\}, \{a, b, c\}, \{a, b, z_0\}, \delta, q_0, z_0, \{q_f\})$$

is a PDA, where  $\delta$  is defined as :

$$\delta(q_0, a, z_0) = \{(q_0, az_0)\}, \quad \delta(q_0, b, z_0) = \{(q_0, bz_0)\}$$

$$\delta(q_0, a, a) = \{(q_0, aa)\} \quad \delta(q_0, b, a) = \{(q_0, ba)\}$$

$$\delta(q_0, a, b) = \{(q_0, ab)\} \quad \delta(q_0, b, b) = \{(q_0, bb)\}$$

$$\delta(q_0, c, a) = \{(q_1, a)\} \quad \delta(q_0, c, b) = \{(q_1, b)\}$$

$$\delta(q_0, c, z_0) = \{(q_1, z_0)\} \quad \delta(q_1, a, a) = \delta(q_1, b, b)$$

$$\delta(q_1, \epsilon, z_0) = \{(q_f, z_0)\} \quad = \{(q_1, \epsilon)\}$$

M accepts  $L = \{w c w^R \mid w \in \{a, b\}^*\}$  by final state  
i.e.  $L(M) = L$

- $(q_0, bacab, z_0) \xrightarrow{} (q_0, acab, bz_0)$   
 $\xrightarrow{} (q_0, cab, abz_0)$   
 $\xrightarrow{} (q_1, ab, abz_0)$   
 $\xrightarrow{} (q_1, b, bz_0)$   
 $\xrightarrow{} (q_1, \epsilon, z_0)$   
 $\xrightarrow{} (q_f, \epsilon, z_0)$

i.e.  $(q_0, bacab, z_0) \xrightarrow{*} (q_f, \epsilon, z_0)$

Proceeding in a similar way; we can show that

$(q_0, w\omega^R, z_0) \xrightarrow{*} (q_f, \epsilon, z_0) \nvdash w \in \{a,b\}^*$

i.e.  $L = \{w\omega^R \mid w \in \{a,b\}^*\} \subseteq L(M)$

Reverse inclusion is also true.

- $(q_0, abcbb, z_0) \xrightarrow{} (q_0, bcb, a z_0)$   
 $\xrightarrow{} (q_0, cbb, ba z_0)$   
 $\xrightarrow{} (q_1, bb, ba z_0)$   
 $\xrightarrow{} (q_1, b, a z_0)$

• PDA halts when it is in  $ID(q_1, b, a z_0)$  as

$$\delta(q_1, b, a) = \phi$$

- As  $\delta(q_0, c, z_0) = \phi$ , the PDA cannot make any transition if it starts with an  $ID$  of the form  $(q_0, cw, z_0)$