

CS 483 - Introduction to the Theory of Computation

Lecture 8

Examples of Pushdown Automata

Example 3:

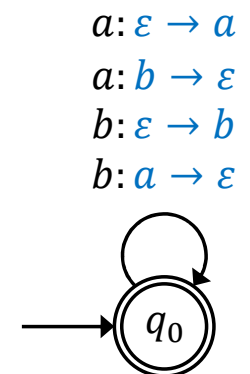
$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

- $Q = \{q_0\}, \Sigma = \{a, b\}, \Gamma = \{a, b\}, F = \{q_0\}$
- Non-empty values of δ :
 - $\delta(q_0, a, \varepsilon) = \{(q_0, a)\}$ (After reading a , you can push a ...
 - $\delta(q_0, a, b) = \{(q_0, \varepsilon)\}$... or pop b)
 - $\delta(q_0, b, \varepsilon) = \{(q_1, b)\}$
 - $\delta(q_0, b, a) = \{(q_1, \varepsilon)\}$



Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

A CFG for L_3 :

$$S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$$

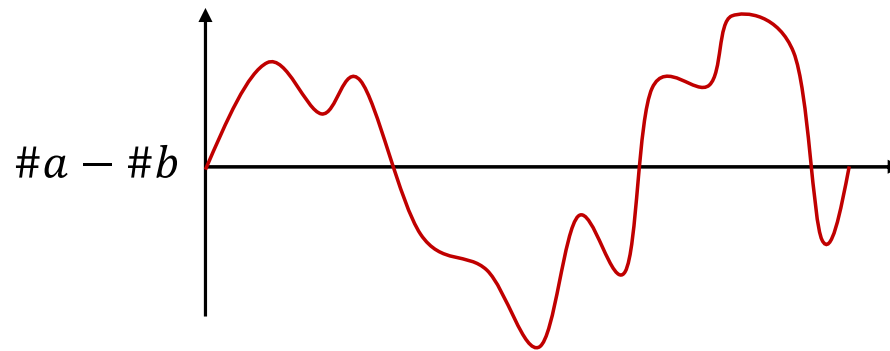
Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

A CFG for L_3 :

$$S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$$



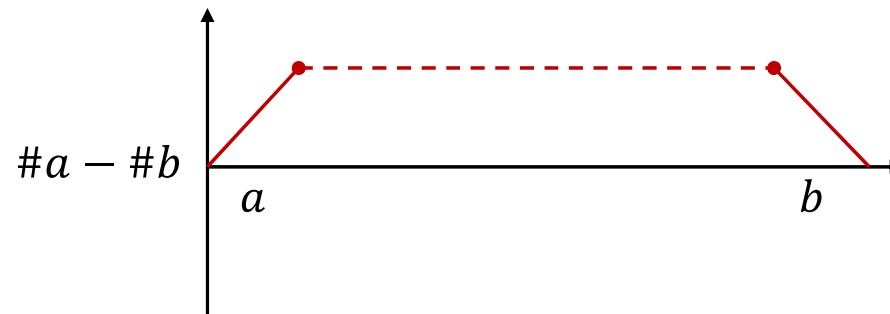
Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

A CFG for L_3 :

$$S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$$



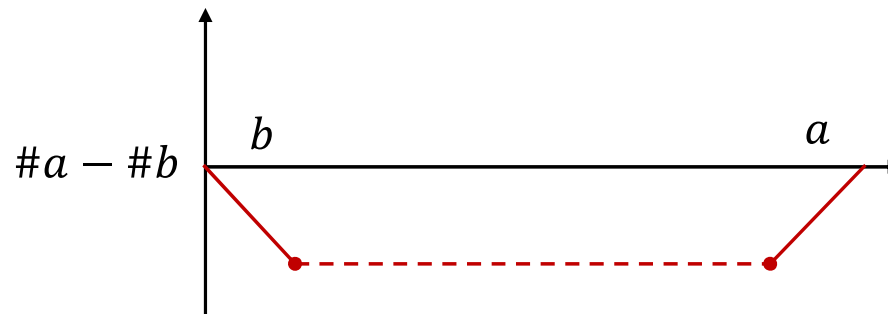
Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

A CFG for L_3 :

$$S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$$



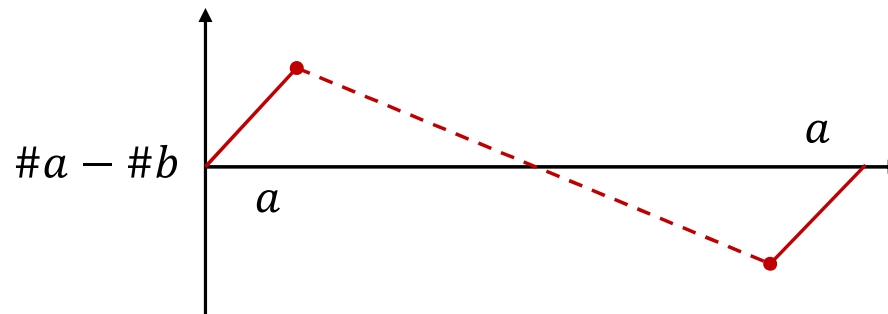
Examples of Pushdown Automata

Example 3:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$

A CFG for L_3 :

$$S \rightarrow \varepsilon \mid aSb \mid bSa \mid SS$$

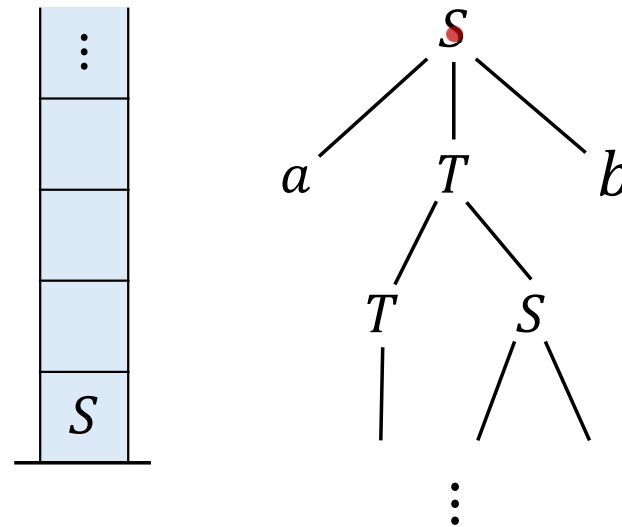


Equivalence between PDA and CFG

Theorem. A language can be computed by a PDA if and only if it is context-free.

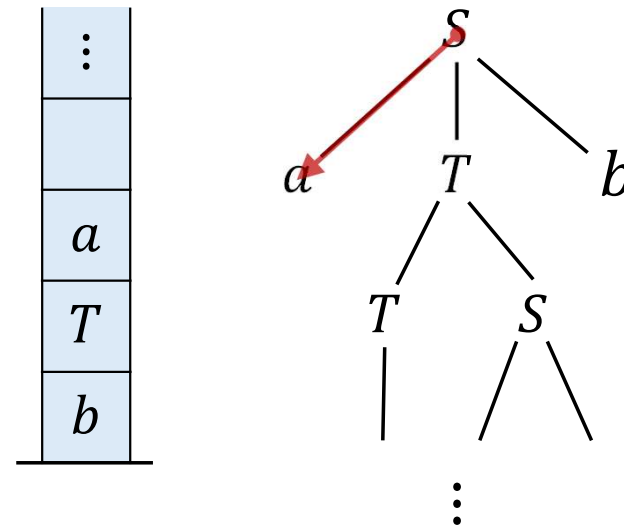
From CFG to PDA

Given a CFG $G = (\Sigma, N, P, S)$, we construct a PDA M that parses G by traversing the parsing tree.



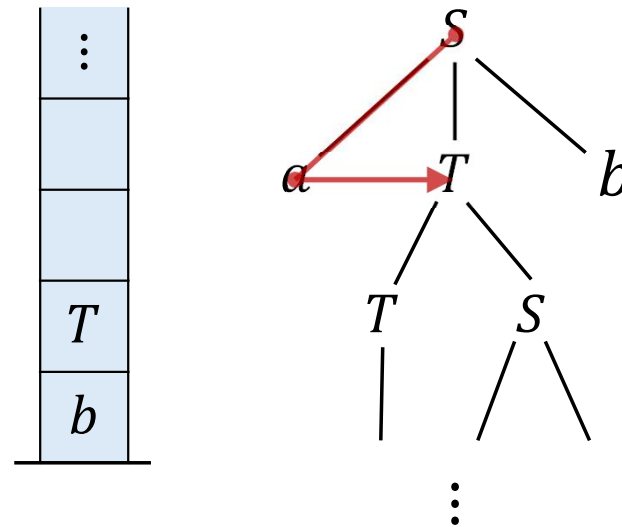
From CFG to PDA

Given a CFG $G = (\Sigma, N, P, S)$, we construct a PDA M that parses G by traversing the parsing tree.



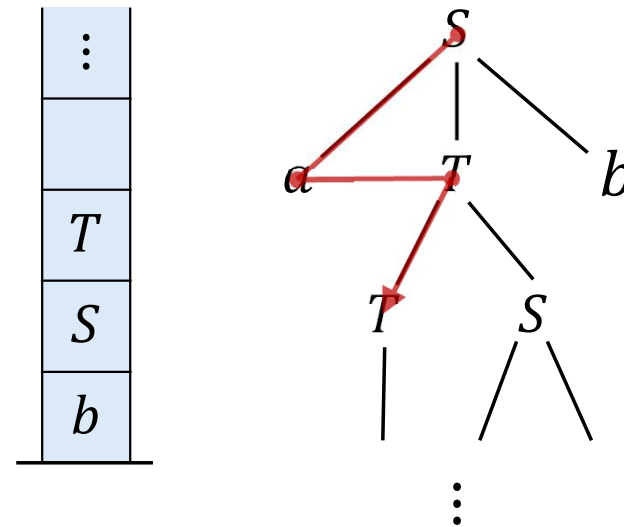
From CFG to PDA

Given a CFG $G = (\Sigma, N, P, S)$, we construct a PDA M that parses G by traversing the parsing tree.



From CFG to PDA

Given a CFG $G = (\Sigma, N, P, S)$, we construct a PDA M that parses G by traversing the parsing tree.

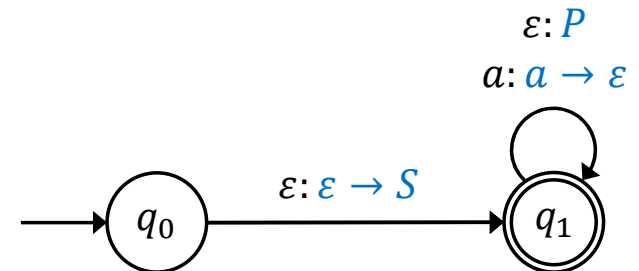


From CFG to PDA

Given a CFG $G = (\Sigma, N, P, S)$, we construct a PDA M that parses G by traversing the parsing tree.

$M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ where:

- $Q = \{q_0, q_1\}, F = \{q_1\}$
- $\Gamma = \Sigma \cup N$
- Non-empty values of δ :
 - $\delta(q_0, \varepsilon, \varepsilon) = \{(q_1, S)\}$
 - $\delta(q_1, \varepsilon, A) = \{(q_1, \alpha) \mid P \text{ has rule } A \rightarrow \alpha\}$
 - $\delta(q_1, a, a) = \{(q_1, \varepsilon)\}$



From CFG to PDA

Input w is in $L(M)$ if and only if there exists:

- $s_0, \dots, s_m \in (\Sigma \cup N)^*$, $s_0 = S$, $s_m = \varepsilon$
- $y_1 \cdots y_m = w$
- $s_{i-1} = y_i s_i$ (pop), or $s_{i-1} = \alpha t$, $s_i = t$ and $y_i = \varepsilon$ (replace)

From CFG to PDA

Input w is in $L(M)$ if and only if there exists:

- $s_0, \dots, s_m \in (\Sigma \cup N)^*$, $s_0 = S$, $s_m = \varepsilon$
- $y_1 \cdots y_m = w$
- $s_{i-1} = y_i s_i$ (pop), or $s_{i-1} = \alpha t$, $s_i = t$ and $y_i = \varepsilon$ (replace)

This exactly corresponds to a **derivation** in G :

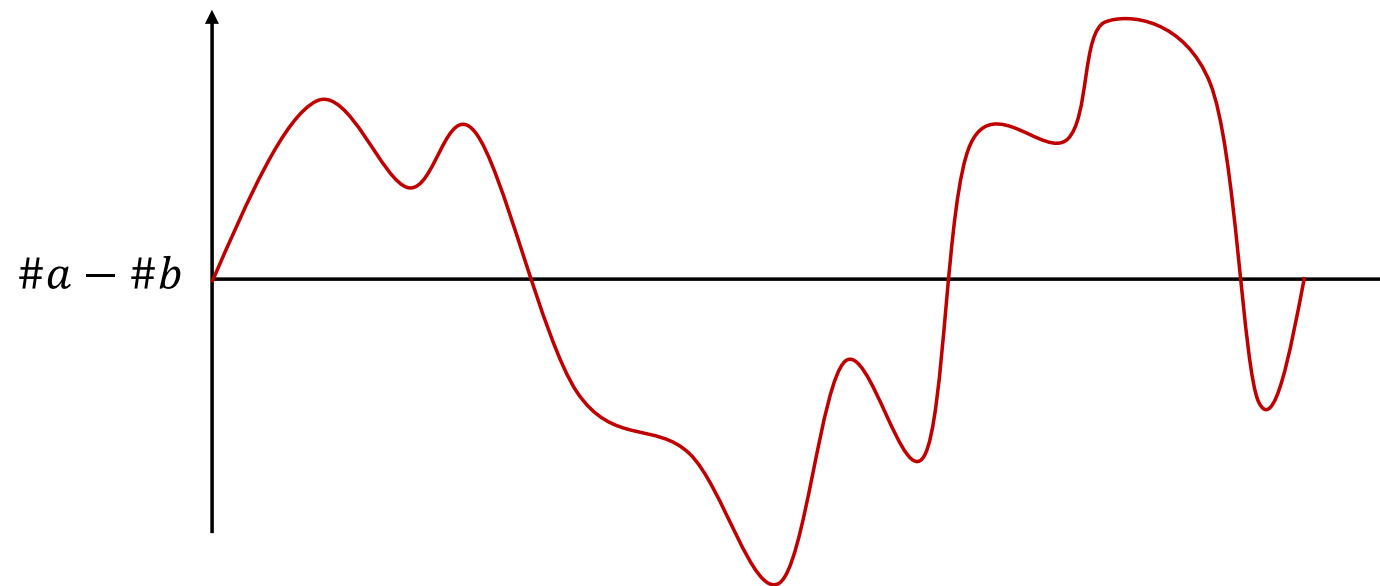
$$s_0 \Rightarrow y_1 s_1 \Rightarrow y_1 y_2 s_2 \Rightarrow \cdots \Rightarrow y_1 \cdots y_m s_m$$

which exists if and only if $w \in L(G)$.

From PDA to CFG

Recall the example:

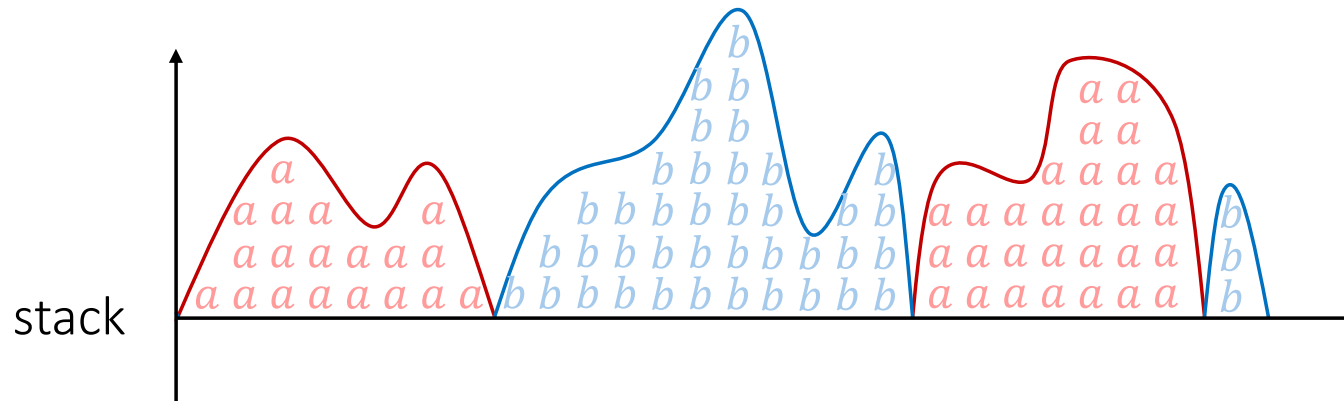
$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$



From PDA to CFG

Recall the example:

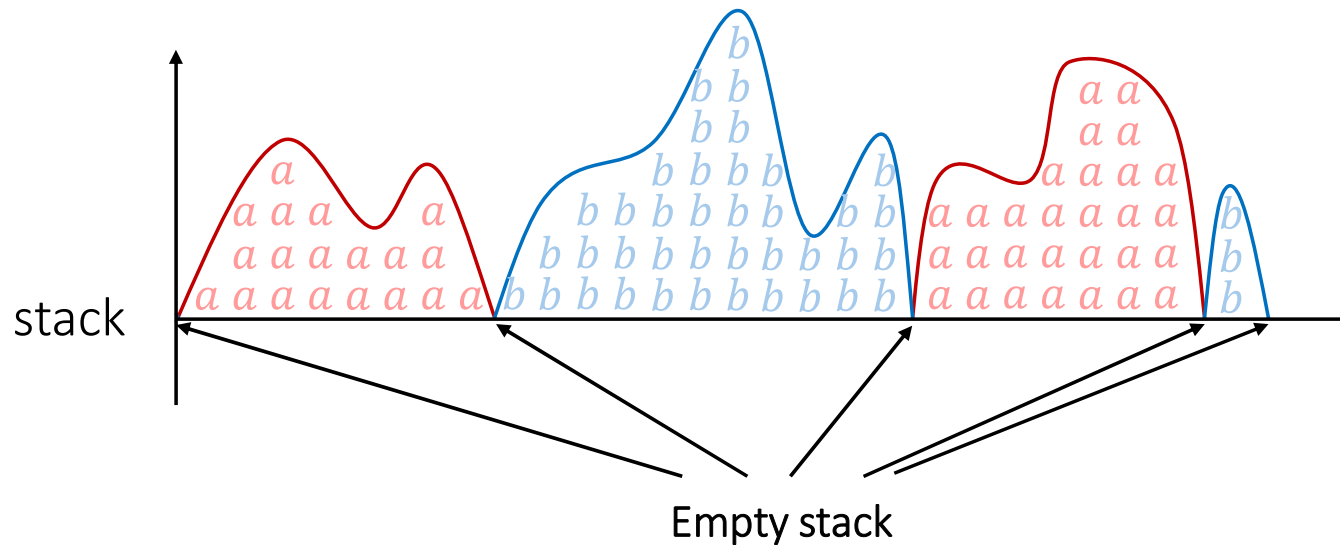
$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$



From PDA to CFG

Recall the example:

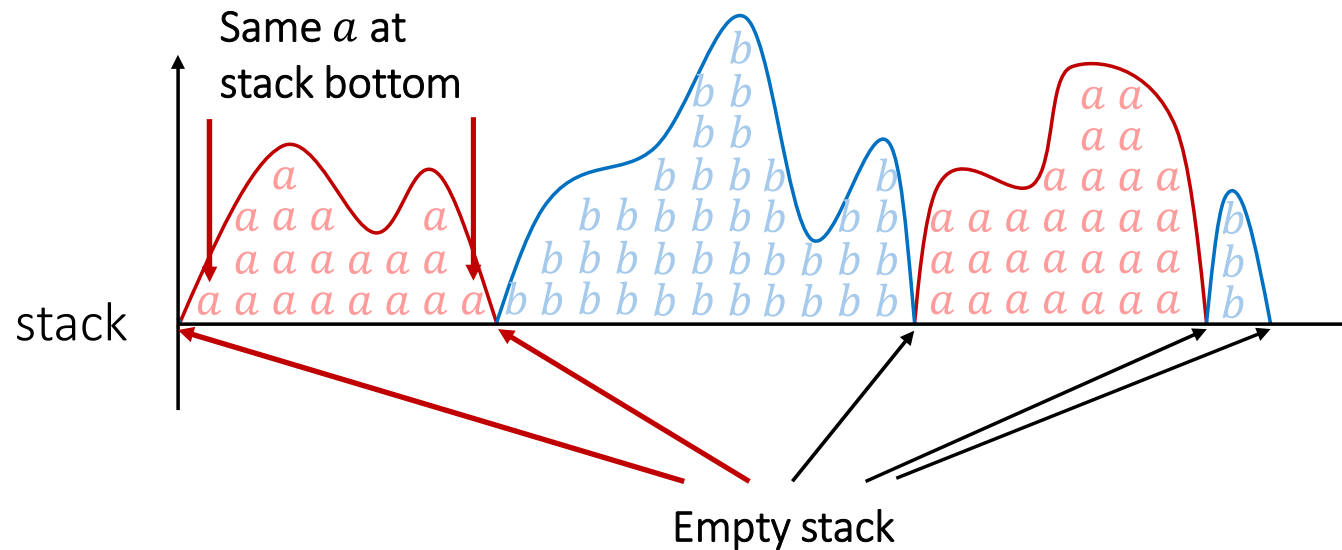
$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$



From PDA to CFG

Recall the example:

$$L_3 = \{w \in \{a, b\}^* \mid w \text{ contains same numbers of } a \text{ and } b\}$$



From PDA to CFG

Let A_{pq} be a non-terminal symbol that represents all strings that corresponds to the PDA transition:

state p and empty stack to state q and empty stack

From PDA to CFG

Let A_{pq} be a non-terminal symbol that represents all strings that corresponds to the PDA transition:

state p and empty stack to state q and empty stack

Then we have rules:

- $A_{pq} \rightarrow A_{pr}A_{rq}$ for every $r \in Q$

From PDA to CFG

Let A_{pq} be a non-terminal symbol that represents all strings that corresponds to the PDA transition:

state p and empty stack to state q and empty stack

Then we have rules:

- $A_{pq} \rightarrow A_{pr}A_{rq}$ for every $r \in Q$
- $A_{pq} \rightarrow aA_{p'q'}b$ for $p', q' \in Q$ and $a, b \in \Sigma$ such that for some $s \in \Gamma$:

$(p', s) \in \delta(p, a, \varepsilon)$	$(p \text{ to } p': \text{read } a \text{ and push } s)$
$(q, \varepsilon) \in \delta(q', b, s)$	$(q' \text{ to } q: \text{read } b \text{ and pop } s)$

From PDA to CFG

Lemma. For every PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$, there is an equivalent PDA $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$ such that every valid transition only uses at most one **single** operation on the stack:

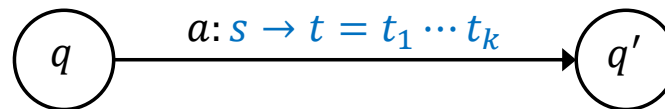
- Push: $(q', s) \in \delta(q, a, \varepsilon), \quad s \in \Gamma \cup \{\varepsilon\}$
- Or Pop: $(q', \varepsilon) \in \delta(q, a, s)$

From PDA to CFG

Lemma. For every PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$, there is an equivalent PDA $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$ such that every valid transition only uses at most one **single** operation on the stack:

- Push: $(q', s) \in \delta(q, a, \varepsilon), \quad s \in \Gamma \cup \{\varepsilon\}$
- Or Pop: $(q', \varepsilon) \in \delta(q, a, s)$

Proof. Decompose each transition $(q', t) \in \delta(q, a, s)$:

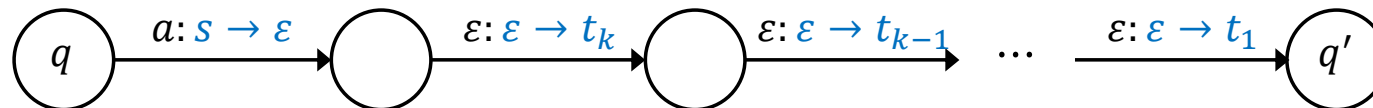


From PDA to CFG

Lemma. For every PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$, there is an equivalent PDA $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$ such that every valid transition only uses at most one **single** operation on the stack:

- Push: $(q', s) \in \delta(q, a, \varepsilon), \quad s \in \Gamma \cup \{\varepsilon\}$
- Or Pop: $(q', \varepsilon) \in \delta(q, a, s)$

Proof. Decompose each transition $(q', t) \in \delta(q, a, s)$:



From PDA to CFG

From $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$, construct CFG $G = (\Sigma, N, P, S)$ where:

- $N = \{A_{pq} \mid p, q \in Q'\} \cup \{S\}$
- P contains the following rules:
 $S \rightarrow A_{q_0q}$ for every $q \in F$

From PDA to CFG

From $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$, construct CFG $G = (\Sigma, N, P, S)$ where:

- $N = \{A_{pq} \mid p, q \in Q'\} \cup \{S\}$
- P contains the following rules:
 - $S \rightarrow A_{q_0q}$ for every $q \in F$
 - $A_{pq} \rightarrow A_{pr}A_{rq}$ for every $p, q, r \in Q'$

From PDA to CFG

From $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$, construct CFG $G = (\Sigma, N, P, S)$ where:

- $N = \{A_{pq} \mid p, q \in Q'\} \cup \{S\}$
- P contains the following rules:

$S \rightarrow A_{q_0q}$ for every $q \in F$

$A_{pq} \rightarrow A_{pr}A_{rq}$ for every $p, q, r \in Q'$

$A_{pq} \rightarrow aA_{p'q'}b$ for every $p, q, p', q' \in Q'$ and $a, b \in \Sigma \cup \{\varepsilon\}$,
such that $(p', s) \in \delta'(p, a, \varepsilon)$ and $(q, \varepsilon) \in \delta'(q', b, s)$

From PDA to CFG

From $M' = (Q', \Sigma, \Gamma, \delta', q_0, F)$, construct CFG $G = (\Sigma, N, P, S)$ where:

- $N = \{A_{pq} \mid p, q \in Q'\} \cup \{S\}$
- P contains the following rules:

$$S \rightarrow A_{q_0q} \text{ for every } q \in F$$

$$A_{pq} \rightarrow A_{pr}A_{rq} \text{ for every } p, q, r \in Q'$$

$$A_{pq} \rightarrow aA_{p'q'}b \text{ for every } p, q, p', q' \in Q' \text{ and } a, b \in \Sigma \cup \{\varepsilon\},$$

such that $(p', s) \in \delta'(p, a, \varepsilon)$ and $(q, \varepsilon) \in \delta'(q', b, s)$

$$A_{pp} \rightarrow \varepsilon \text{ for every } p \in Q'$$

Equivalence between PDA and CFG

Theorem. A language can be computed by a PDA if and only if it is context-free.

By-product. Every CFG can be equivalently written with rules of forms:

$$A \rightarrow B$$

$$A \rightarrow BC$$

$$A \rightarrow a$$

$$A \rightarrow \varepsilon$$

Equivalence between PDA and CFG

Theorem. A language can be computed by a PDA if and only if it is context-free.

By-product. Every CFG can be equivalently written with rules of forms:

$$A \rightarrow B$$

$$A \rightarrow BC$$

$$A \rightarrow a$$

$$\textcolor{red}{A \rightarrow \varepsilon} \quad (\text{if } A \xRightarrow{*} \varepsilon, \text{ replace rule } D \rightarrow AC \text{ with } D \rightarrow C)$$

Equivalence between PDA and CFG

Theorem. A language can be computed by a PDA if and only if it is context-free.

By-product. Every CFG can be equivalently written with rules of forms:

~~$A \rightarrow B$~~ (if $A \xRightarrow{*} B$, for rule $D \rightarrow AC$ add $D \rightarrow BC$)

$A \rightarrow BC$

$A \rightarrow a$

~~$A \rightarrow \epsilon$~~

Chomsky Normal Form

Theorem. Every CFG can be equivalently written with rules of forms:

$$A \rightarrow BC$$

$$A \rightarrow a$$

$$S \rightarrow \varepsilon \quad (\text{if } \varepsilon \in L(G))$$

Chomsky Normal Form

Theorem. Every CFG can be equivalently written with rules of forms:

$$A \rightarrow BC$$

$$A \rightarrow a$$

$$S \rightarrow \varepsilon \quad (\text{if } \varepsilon \in L(G))$$

Corollary. For every CFL L , deciding whether $w \in L$ can be solved by DP:

- $F[i, i, A] = \text{True}$ if there is rule $A \rightarrow w_i$
- $F[i, j, A] = \text{True}$ if there is rule $A \rightarrow BC$, and both $F[i, k, B]$ and $F[k + 1, j, C]$ are true for some $i \leq k < j$.