3.3. A Pumping Lemma for Context-Free Languages

Theorem 3.14 (Pumping Lemma: Bar-Hillel, Perles, Shamir 1961)

Let L be a context-free language on Σ. Then there exists a constant k that depends on L such that each word $z \in L$, $|z| \geq k$, has a factorization of the form $z = uvwxy$ that satisfies all of the following conditions:

(1) $|vx| \geq 1$,

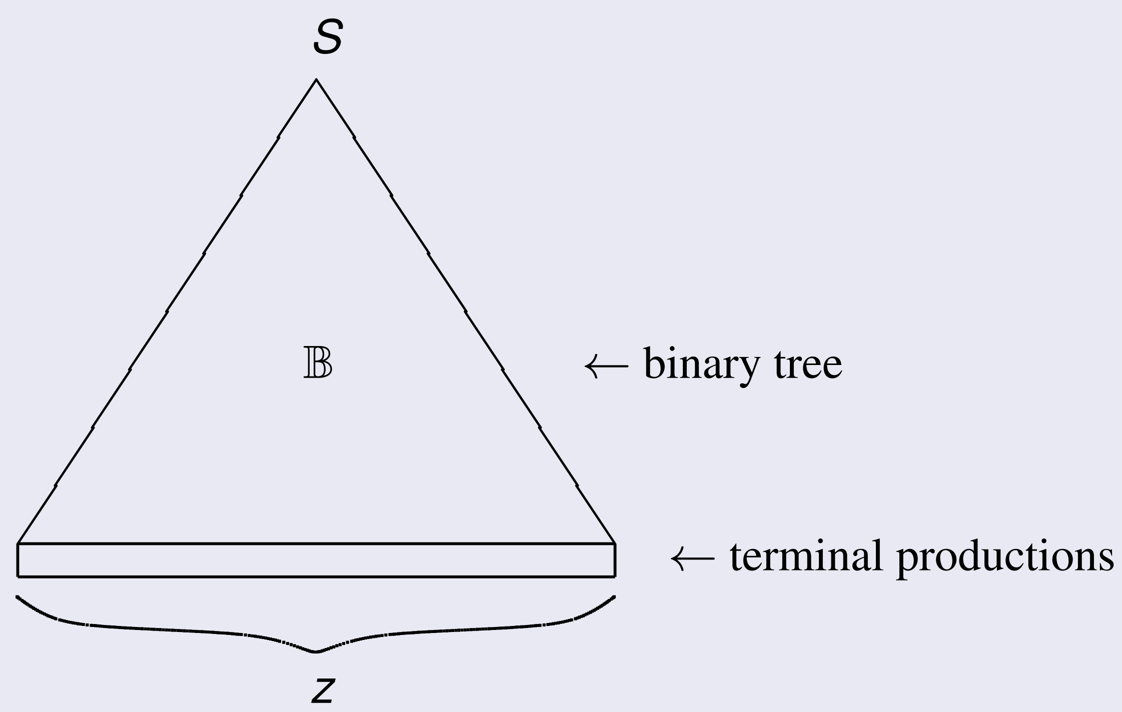
(2) $|vwx| \leq k$,

(3) $uv^iwx^iy \in L$ *for all* $i \geq 0$.

Proof of Theorem 3.14.

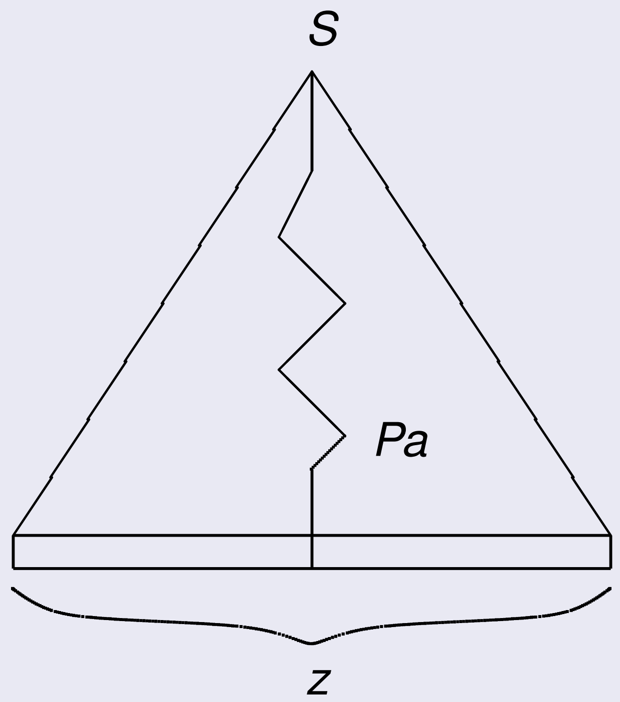
Let $G = (N, \Sigma, P, S)$ be a context-free grammar in CNF for $L - \{\varepsilon\}$, and let $n = |N|$.
We choose $k := 2^n$.
Now let $z \in L$ such that $|z| \geq k$.

We consider a syntax tree for z:



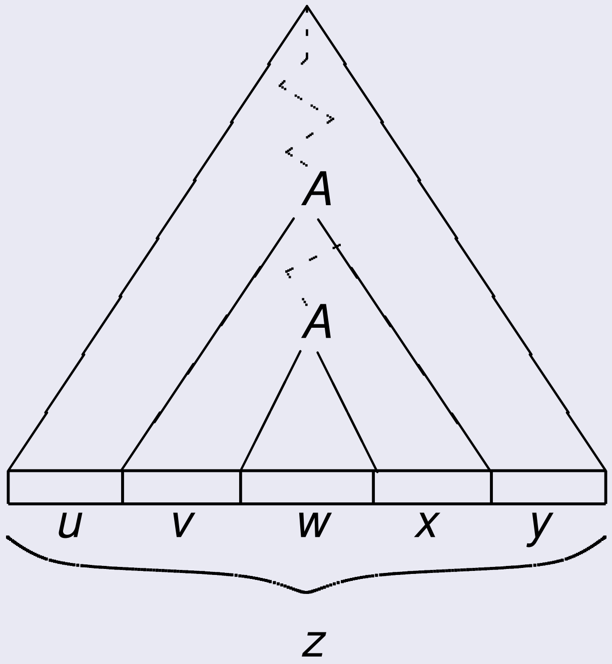
Proof of Theorem 3.14 (cont.)

$\mathbb{B}$ has $|z| \geq k = 2^n$ leaves. Thus, $\mathbb{B}$ contains a path of length $\ell \geq n$.
We consider a path Pa of maximal length ℓ:



Proof of Theorem 3.14 (cont.)

Pa contains $\ell + 1 \geq n + 1$ nodes labelled with nonterminals, that is, at least one nonterminal occurs twice at the nodes of *Pa*.



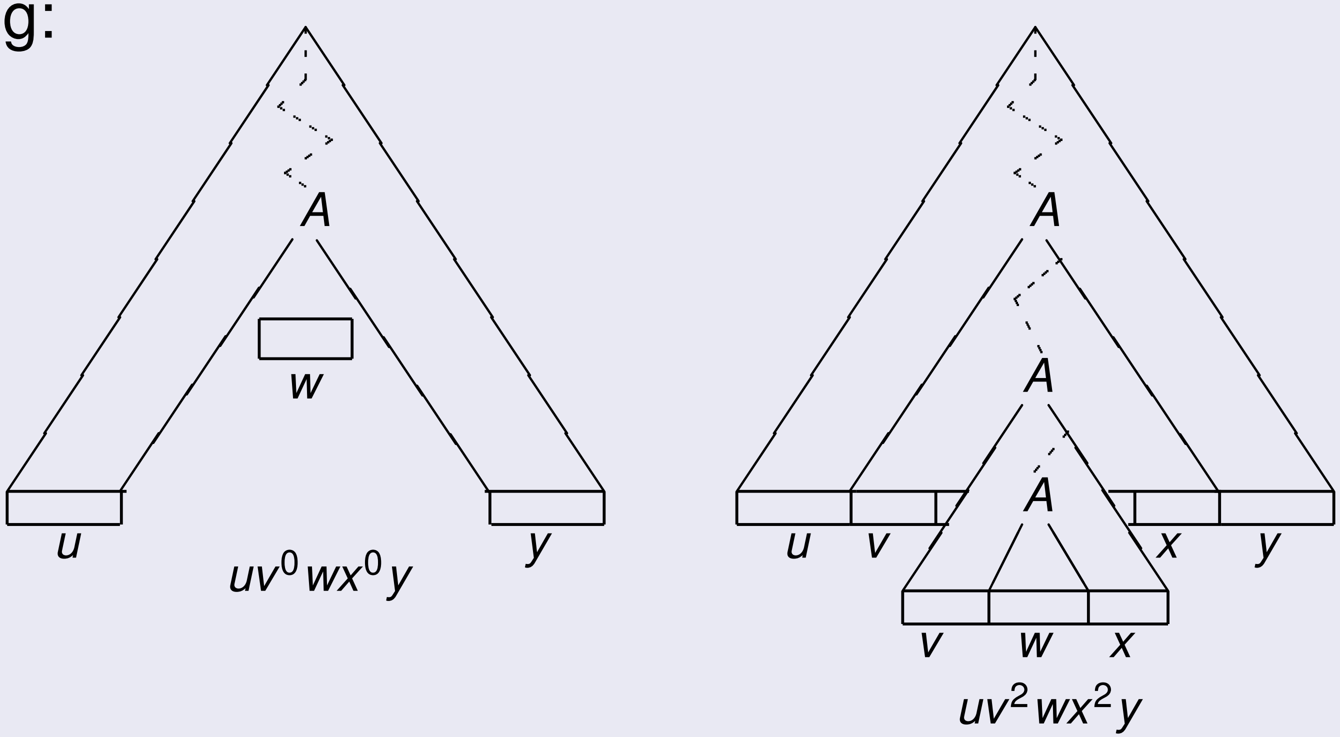
We choose the first such repetition in *Pa* starting from the leaf.

Proof of Theorem 3.14 (cont.)

$S \to^* uAy$
$A \to BC \to^* vAx$
$A \to^* w$

Chomsky Normal Form : $|vx| \geq 1$
Height of upper node with label $A \leq n$: $|vwx| \leq 2^n = k$

Pumping:



□

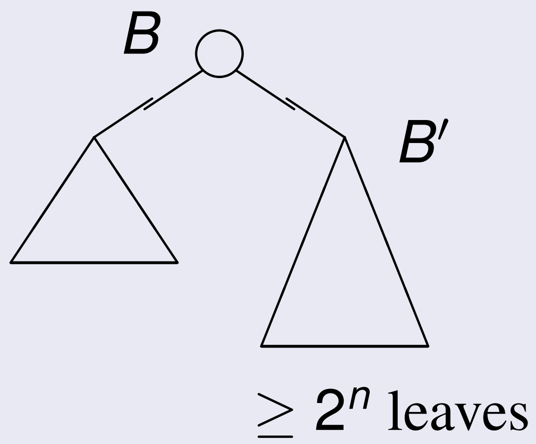
Lemma 3.15

Let B be a binary tree, each inner node of which has two sons. If B has at least 2^n leaves, then B contains a path of length at least n.

Proof by induction on n:

$n = 0$: Number of leaves is $\geq 2^0 = 1$.
B contains a path of length ≥ 0.

$n \rightsquigarrow n+1$: B has $\geq 2^{n+1}$ leaves:



I.H.: B' contains a path of length $\geq n$.

Hence: B contains a path of length $\geq n+1$. □

Example:

Claim: $L = \{ a^m b^m c^m \mid m \geq 1 \}$ is not context-free.

Proof (indirect).

Assume that L is context-free. Then L satisfies the Pumping Lemma, that is, $\exists k \in \mathbb{N}_+ \, \forall z \in L : |z| \geq k \leadsto \exists z = uvwxy$:
$|vx| \geq 1$, $|vwx| \leq k$, and $uv^i wx^i y \in L$ for all $i \geq 0$.

Consider the word $z := a^k b^k c^k$: $z \in L$ and $|z| = 3k \geq k$.

Hence:

$\exists z = uvwxy$ s.t. $vx \neq \varepsilon$, $|vwx| \leq k$, and $uv^i wx^i y \in L$ for all $i \geq 0$.

$|vwx| \leq k$: $|vx|_a = 0$ or $|vx|_c = 0$

$\leadsto uv^0 wx^0 y = uwy \notin L$. Contradiction!

Thus, L is not context-free. □

Example:

The language $L := \{ a^n b^m c^n d^m \mid n, m \geq 1 \}$ is not context-free, as for $z = a^k b^k c^k d^k$, no factor vwx satisfying $|vwx| \leq k$ can possibly contain a's and c's or b's and d's. □

Example:

Let $L := \{ a^i b^j c^k d^\ell \mid i, j, k, \ell \geq 0, \text{ and } i > 0 \text{ implies } j = k = \ell \}$, and let $n > 0$ be a constant.

If $z = b^j c^k d^\ell$, $|z| \geq n$, then we choose vwx as a factor of b^j, c^k or d^ℓ. For all $m \geq 0$, $uv^m wx^m y \in L$.

If $z = a^i b^j c^j d^j$, $|z| \geq n$ and $i > 0$, then we choose vwx as a factor a^i, and it follows that $uv^m wx^m y \in L$ for all $m \geq 0$.

Thus, L satisfies the Pumping Lemma, but we will see later that L is not context-free. □

Theorem 3.16

Each context-free language over a one-letter alphabet is regular.

Proof.

Let L be a context-free language over $\{a\}$, and let k be the constant from the Pumping Lemma for L:

$$\forall z \in L : |z| \geq k \rightsquigarrow \exists z = uvwxy : vx \neq \varepsilon, |vwx| \leq k, \text{ and } uv^iwx^iy \in L \text{ f.a. } i \geq 0.$$

However: $L \subseteq \{a\}^* : |z| = m \rightsquigarrow z = a^m$.

$\forall m \geq k : \exists n, \ell \geq 0 : m = n + \ell, 1 \leq \ell \leq k,$ and $a^n a^{i \cdot \ell} \in L$ f.a. $i \geq 0$.

Choose $q := k!$: Each ℓ_i divides q.
Choose $q' \geq q$ such that the following condition is met:

$\forall m \geq q : a^m \in L \rightsquigarrow \exists q \leq p \leq q' : m \equiv p \bmod q$ and $a^p \in L$.

Then $L := \{\, x \in L \mid |x| < q \,\} \cup \{\, a^r a^{iq} \mid q \leq r \leq q', a^r \in L, i \in \mathbb{N} \,\}$, which shows that L is regular. □

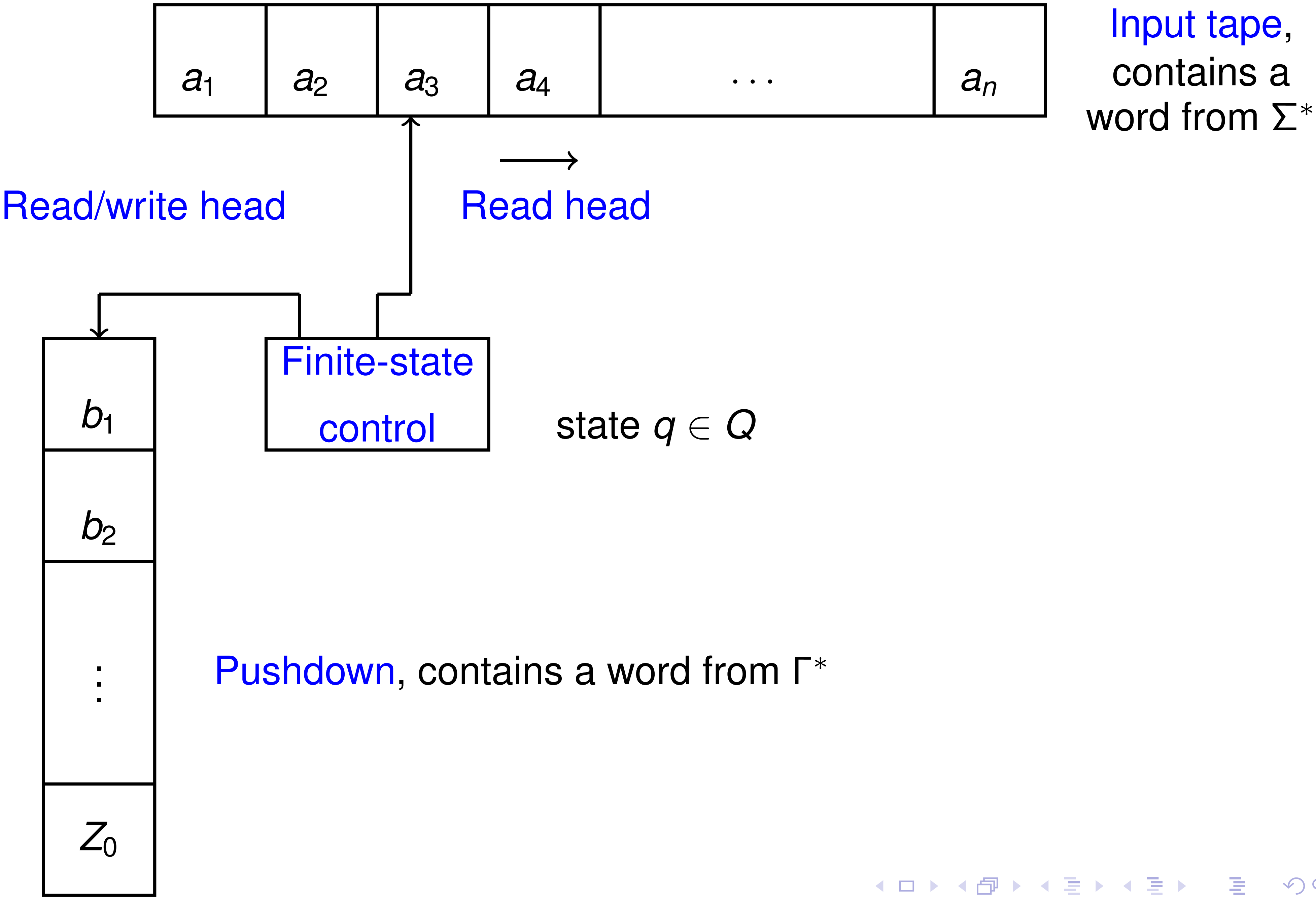
3.4. Pushdown Automata

A pushdown automaton (PDA) is an ε-NFA that has an additional external memory in the form of a pushdown.

A PDA M is defined through a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where

- Q is a finite set of (internal) states,
- Σ is a finite input alphabet,
- Γ is a finite pushdown alphabet,
- $q_0 \in Q$ is the initial state,
- $Z_0 \in \Gamma$ is the bottom marker of the pushdown,
- $F \subseteq Q$ is the set of accepting states, and
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$ is the transition relation. For each $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, and $b \in \Gamma$, $\delta(q, a, b)$ is a finite subset of $Q \times \Gamma^*$.

A PDA can be picturered as follows:



A configuration of M is a triple $(q, \gamma, w) \in Q \times \Gamma^* \times \Sigma^*$, where
q is the current state,
γ is the current content of the pushdown, and
w is the remaining input.
Here the last symbol of γ is the topmost symbol on the pushdown.

The PDA M induces a computation relation $\vdash_M^*$ on the set $\mathrm{CONF} := Q \times \Gamma^* \times \Sigma^*$ of configurations, which is the reflexive and transitive closure of the following single-step relation $\vdash_M$:

$$\begin{array}{lcll} (q, \gamma Z, aw) & \vdash_M & (p, \gamma\beta, w), & \text{if } (p, \beta) \in \delta(q, a, Z) \text{ , and} \\ (q, \gamma Z, w) & \vdash_M & (p, \gamma\beta, w), & \text{if } (p, \beta) \in \delta(q, \varepsilon, Z). \end{array}$$

Example:

Let $M = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, where $Q = \{q_0, q_1, q_2, q_3\}$, $F = \{q_3\}$, $\Sigma = \{a, b\}$, $\Gamma = \{A, B, Z_0\}$, and let δ be given by the following table:

$$
\begin{array}{rcl}
\delta(q_0, a, Z_0) & = & \{(q_1, Z_0A)\}, \\
\delta(q_0, b, Z_0) & = & \{(q_1, Z_0B)\}, \\
\delta(q_0, \varepsilon, Z_0) & = & \{(q_3, \varepsilon)\}, \\
\delta(q_1, a, Z) & = & \{(q_1, ZA)\}, \\
\delta(q_1, b, Z) & = & \{(q_1, ZB)\}, \\
\delta(q_1, \varepsilon, Z) & = & \{(q_2, Z)\}, \\
\delta(q_2, a, A) & = & \{(q_2, \varepsilon)\}, \\
\delta(q_2, b, B) & = & \{(q_2, \varepsilon)\}, \\
\delta(q_2, \varepsilon, Z_0) & = & \{(q_3, \varepsilon)\}.
\end{array}
$$

On input *abba*, the PDA M can execute the following computation:

$$
\begin{array}{ccccccc}
(q_0, Z_0, abba) & \vdash & (q_1, Z_0A, bba) & \vdash & (q_1, Z_0AB, ba) & \vdash & (q_2, Z_0AB, ba) \\
 & \vdash & (q_2, Z_0A, a) & \vdash & (q_2, Z_0, \varepsilon) & \vdash & (q_3, \varepsilon, \varepsilon).
\end{array}
$$

Depending on its mode of operation, a PDA M accepts one of two possible languages:

$L(M)$ denotes the language

$$L(M) := \{ w \in \Sigma^* \mid (q_0, Z_0, w) \vdash_M^* (p, \gamma, \varepsilon) \text{ for some } p \in F \text{ and } \gamma \in \Gamma^* \},$$

that is, $L(M)$ is the language that M accepts with final states,

and $N(M)$ denotes the language

$$N(M) := \{ w \in \Sigma^* \mid (q_0, Z_0, w) \vdash_M^* (p, \varepsilon, \varepsilon) \text{ for some } p \in Q \},$$

that is, $N(M)$ is the language that M accepts with empty pushdown.

Example (cont.):

$L(M) = N(M) = \{ uu^R \mid u \in \{a, b\}^* \}$.

Theorem 3.17

For each PDA M_1, there exists a PDA M_2 such that $L(M_1) = N(M_2)$.

Proof.

Let $M_1 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$, and let $L = L(M_1)$, that is,

$$L = \{ w \in \Sigma^* \mid (q_0, Z_0, w) \vdash^*_{M_1} (p, \gamma, \varepsilon) \text{ for some } p \in F \text{ and } \gamma \in \Gamma^* \}.$$

The PDA M_2 will simulate the PDA M_1 step by step.

Essentially M_2 must solve the following two problems:

- M_2 must be able to recognize when M_1 empties its pushdown without being in a final state, as in this situation, M_2 must not accept.
- M_2 must empty its pushdown when M_1 accepts.

Proof of Theorem 3.17 (cont.)

Let $M_2 = (Q \cup \{q_\ell, q_0'\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0, \emptyset)$ be defined as follows:

(1) $\delta'(q_0', \varepsilon, X_0) = \{(q_0, X_0 Z_0)\}$,

(2) $\delta'(q, a, Z) \supseteq \delta(q, a, Z)$ for all $q \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, and $Z \in \Gamma$,

(3) $\delta'(q, \varepsilon, Z) \ni (q_\ell, Z)$ for all $q \in F$ and $Z \in \Gamma \cup \{X_0\}$,

(4) $\delta'(q_\ell, \varepsilon, Z) = \{(q_\ell, \varepsilon)\}$ for all $Z \in \Gamma \cup \{X_0\}$.

By (1) M_2 enters the initial configuration of M_1, with the symbol X_0 below the bottom marker of M_1.

By (2) M_2 simulates the computation of M_1 step by step.

If and when M_1 reaches a final state, then M_2 empties its pushdown using (3) and (4). It follows that $L(M_1) \subseteq N(M_2)$.

If M_1 empties its pushdown without being in a final state, then M_2 gets stuck in the corresponding configuration (q, X_0, aw).

It can now be shown that $N(M_2) = L(M_1)$. □

Theorem 3.18

For each PDA M_2, there exists a PDA M_1 such that $L(M_1) = N(M_2)$.

Theorem 3.19

From a given context-free grammar G, one can effectively construct a PDA M such that $N(M) = L(G)$.

Proof.

Let $G = (N, \Sigma, S, P)$ be a context-free grammar. By Theorem 3.10 we can assume that G is in Greibach Normal form. To simplify the discussion we assume that $\varepsilon \notin L(G)$.

We define a PDA $M = (\{q\}, \Sigma, N, \delta, q, S, \emptyset)$ by taking $\delta(q, a, A) := \{\, (q, \gamma^R) \mid (A \to a\gamma) \in P \,\}$ for all $a \in \Sigma$ and $A \in N$.

Proof of Theorem 3.19 (cont.)

Claim:

$\forall x \in \Sigma^* \, \forall \alpha \in N^* : S \to_{\text{lm}}^* x\alpha$ iff $(q, S, x) \vdash_M^* (q, \alpha^R, \varepsilon)$.

Proof.

By induction on i, we will prove the following:

$(*)$ If $(q, S, x) \vdash_M^i (q, \alpha^R, \varepsilon)$, then $S \to_{\text{lm}}^i x\alpha$.

If $i = 0$, then $\alpha = S$ and $x = \varepsilon$.

Assume that $(*)$ has already been shown for some integer $i - 1$, and assume that $(q, S, x) \vdash_M^{i-1} (q, \beta^R, x') \vdash_M (q, \alpha^R, \varepsilon)$.

As M has no ε-transitions, $x = ya$ and $x' = a$ for some $y \in \Sigma^{i-1}$ and $a \in \Sigma$, that is, the computation above has the form

$$(q, S, ya) \vdash_M^{i-1} (q, \beta^R, a) \vdash_M (q, \alpha^R, \varepsilon).$$

Proof of Theorem 3.19 (cont.)

Proof of Claim (cont.)

Thus, on input y, M can execute the following computation:

$$(q, S, y) \vdash_M^{i-1} (q, \beta^R, \varepsilon).$$

By our I.H. this gives a derivation $S \to_{\mathrm{lm}}^{i-1} y\beta$.
As $(q, \beta^R, a) \vdash_M (q, \alpha^R, \varepsilon)$, $\beta^R = \gamma^R A$ for some $A \in N$ and $\alpha^R = \gamma^R \eta^R$ for a production $(A \to a\eta) \in P$.
Hence, we obtain the following derivation in G:

$$S \to_{\mathrm{lm}}^{i-1} y\beta = yA\gamma \to_{\mathrm{lm}} ya\eta\gamma = x\alpha.$$

This proves the implication from right to left.

Proof of Theorem 3.19 (cont.)

Proof of Claim (cont.)

Now we establish the converse implication:

$(**)$ If $S \to_{\text{lm}}^{i} x\alpha,$ then $(q, S, x) \vdash_M^i (q, \alpha^R, \varepsilon)$,

again proceeding by induction on i.

If $i = 0$, then $x = \varepsilon$ and $\alpha = S$.
As $(q, S, \varepsilon) \vdash_M^0 (q, S, \varepsilon)$, the claim holds in this situation.

Assume that the implication has already been proved for some integer $i - 1$, and let $S \to_{\text{lm}}^{i-1} yA\gamma \to_{\text{lm}} ya\eta\gamma$ be a derivation in G, where $(A \to a\eta) \in P$.

Then $x = ya$ and $\alpha = \eta\gamma$. By our I.H. $(q, S, y) \vdash_M^{i-1} (q, \gamma^R A, \varepsilon)$, which yields $(q, S, x) = (q, S, ya) \vdash_M^{i-1} (q, \gamma^R A, a)$.

Proof of Theorem 3.19 (cont.)

Proof of Claim (cont.)

Because of the production $(q, \eta^R) \in \delta(q, a, A)$, we obtain the following computation of M:

$$(q, S, x) = (q, S, ya) \vdash_M^{i-1} (q, \gamma^R A, a) \vdash_M (q, \gamma^R \eta^R, \varepsilon) = (q, \alpha^R, \varepsilon).$$

□

For all $x \in \Sigma^+$, we have the following sequence of equivalent statements:

$$\begin{array}{lll} x \in L(G) & \text{iff} & S \to_{\mathrm{lm}}^i x \text{ for some } i \geq 1 \\ & \text{iff} & (q, S, x) \vdash_M^i (q, \varepsilon, \varepsilon) \text{ for some } i \geq 1 \\ & \text{iff} & x \in N(M). \end{array}$$

Thus, $N(M) = L(G)$ follows. □

Example:

$(S \to aBC), (B \to bC), (C \to cDD), (D \to a) \in P$:

$S \to aBC \to abCC \to abcDDC \to^2 abcaaC \to abcaacDD \to^2$ $abcaacaa$.

Computation of M:

$$
\begin{aligned}
(q, S, abcaacaa) \quad & \vdash \quad (q, CB, bcaacaa) \vdash (q, CC, caacaa) \\
& \vdash \quad (q, CDD, aacaa) \vdash (q, CD, acaa) \\
& \vdash \quad (q, C, caa) \vdash (q, DD, aa) \\
& \vdash \quad (q, D, a) \vdash (q, \varepsilon, \varepsilon).
\end{aligned}
$$

Thus, each context-free language is accepted by a PDA that has only a single state and that uses no ε-transitions.