Parsing

Push-Down-Automata (PDA)

Laura Kallmeyer

Heinrich-Heine-Universität Düsseldorf

Summer 2020

Table of contents

1 Intuition

2 Definition

3 PDA and CFG

PDA: Intuition (1)

A push-down automaton (Hopcroft and Ullman, 1979, 1994) is a FSA with an additional stack.
The moves of the automaton depend on

- the current state
- the next input symbol
- the topmost stack symbol

Each move consists of

- changing state
- popping the topmost symbol from the stack
- pushing a new sequence of symbols on the stack

PDA: Intuition (2)

Sample PDA

Automaton that

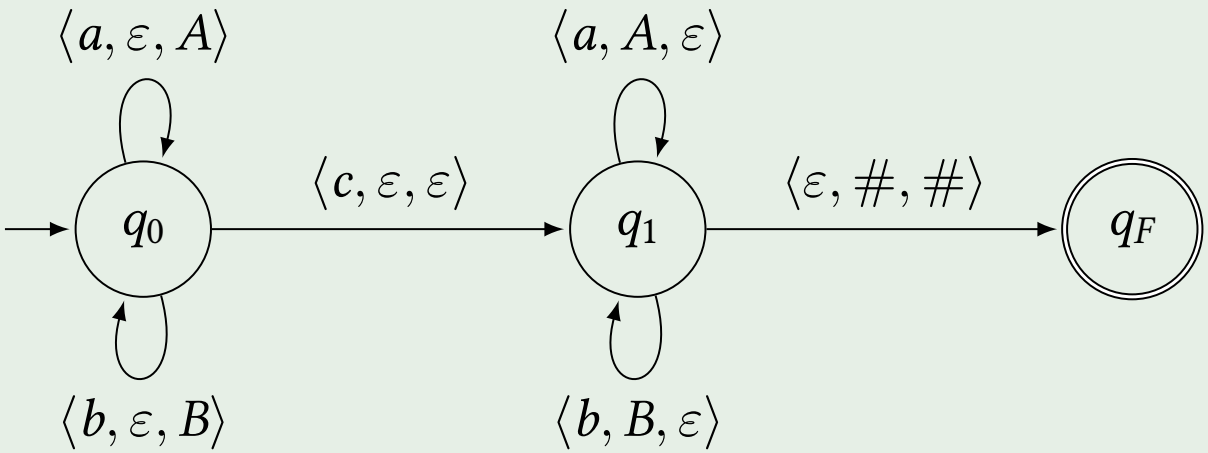
- starts with q_0 and stack $\#$
- in q_0: pushes A on the stack for an input symbol a
- in q_0: pushes B on the stack for an input symbol b
- in q_0: leaves stack unchanged and switches to q_1 for an input symbol c
- in q_1: pops an A from the stack for input symbol a
- in q_1: pops a B from the stack for input symbol b
- in q_1: moves to q_F if the top of the stack is $\#$
- accepts all words that allow to end up in q_F

Which (string) language does this automaton accept?

PDA: Intuition (3)

PDAs can also be drawn as graphs:

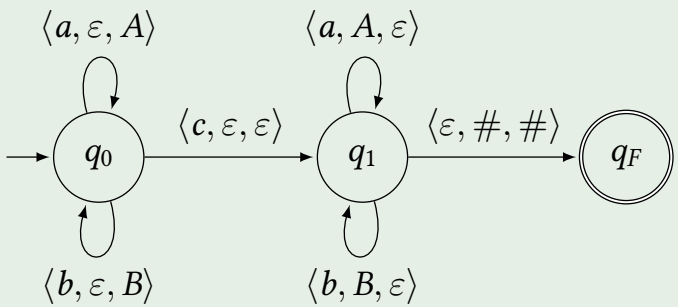
PDA represented as a graph



An edge label $\langle a, A, B \rangle$ signifies that a is read, A is popped from the stack and B is pushed on the stack.

PDA: Intuition (4)

Run of an automaton



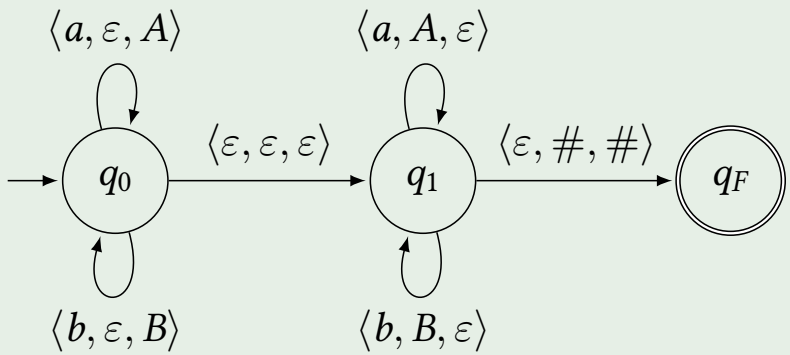
Run of this automaton, input aabcbaa:

state	remaining input	stack (top on the left)
q_0	aabcbaa	$\#$
q_0	abcbaa	$A\#$
q_0	bcbaa	$AA\#$
q_0	cbaa	$BAA\#$
q_1	baa	$BAA\#$
q_1	aa	$AA\#$
q_1	a	$A\#$
q_1	–	$\#$
q_F	–	$\#$

PDA: Intuition (5)

In general, PDAs are non-deterministic, since a given state, input symbol and topmost stack symbol can allow for more than one move.

Non-deterministic PDA



In contrast to FSA, the deterministic version of the automaton is not equivalent to the non-deterministic one: There are languages that are accepted by a non-deterministic PDA but not by any deterministic PDA.

CFLs are the languages accepted by (non-deterministic) PDAs.

PDA: Definition (1)

Push-down automaton

A **push-down automaton** (PDA) M is a tuple $\langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$:

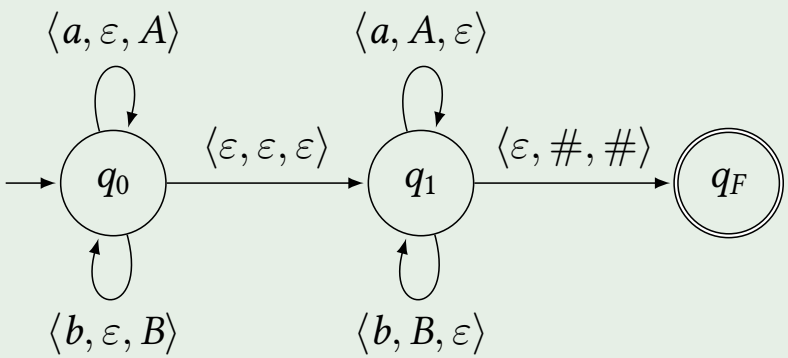
- Q is a finite set of states
- Σ is a finite set, the input alphabet
- Γ is a finite set, the stack alphabet
- $q_0 \in Q$ is the initial state
- $Z_0 \in \Gamma$ is the initial stack symbol
- $F \subseteq Q$ is the set of final states
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathcal{P}_{fin}(Q \times \Gamma^*)$ is the transition function[a]

[a] $\mathcal{P}_{fin}(X)$ is the set of finite subsets of X.

Equivalently, one can even define δ as
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \to \mathcal{P}_{fin}(Q \times \Gamma^*)$.

PDA: Definition (3)

Non-deterministic PDA



$Q = \{q_0, q_1, q_F\}$, $\Sigma = \{a, b\}$, $\Gamma = \{\#, A, B\}$, q_0 start state, $F = \{q_F\}$ and

$$\delta(q_0, a, \varepsilon) = \{\langle q_0, A\rangle\} \qquad \delta(q_0, b, \varepsilon) = \{\langle q_0, B\rangle\}$$
$$\delta(q_0, \varepsilon, \varepsilon) = \{\langle q_1, \varepsilon\rangle\}$$
$$\delta(q_1, a, B) = \{\langle q_1, \varepsilon\rangle\} \qquad \delta(q_1, b, B) = \{\langle q_1, \varepsilon\rangle\}$$
$$\delta(q_1, \varepsilon, \#) = \{\langle q_F, \#\rangle\}$$

In all other cases the value is $\emptyset$.

PDA: Definition (4)

An **instantaneous description** of a PDA is a triple (q, w, γ):

- $q \in Q$ is the current state of the automaton
- $w \in \Sigma^*$ is the remaining part of the input string
- $\gamma \in \Gamma^*$ is the current stack

For all $q, q' \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $w \in \Sigma^*$, $\alpha, \beta, \gamma \in \Gamma^*$:

$$(q, aw, \gamma\alpha) \vdash (q', w, \beta\alpha) \text{ iff } \langle q', \beta \rangle \in \delta(q, a, \gamma)$$

$\vdash^*$ is the reflexive transitive closure of $\vdash$

PDA: Definition (5)

There are two alternatives for the definition of the language accepted by a PDA $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$:

Language of an PDA

- The **language accepted by M with a final state** is

$$L(M) := \{w \mid (q_0, w, Z_0) \vdash^* (q_f, \varepsilon, \gamma) \text{ for a } q_f \in F \text{ and a } \gamma \in \Gamma^*\}$$

- The **language accepted by M with an empty stack** is

$$N(M) := \{w \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon) \text{ for a } q \in Q\}$$

The two modes of acceptance are equivalent, i.e., for each language L there is a PDA M_1 with $L = L(M_1)$ iff there is a PDA M_2 with $L = N(M_2)$.

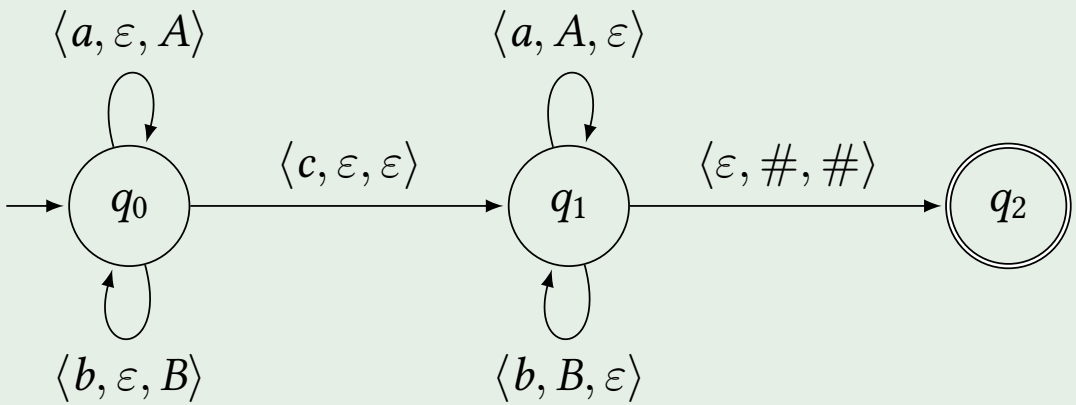
PDA: Definition (6)

Example

PDA $M_1 = \langle Q, \Sigma, \Gamma, \delta, q_0, \#, F \rangle$ for $L(M_1) = \{wcw^R \mid w \in \{a, b\}^*\}$

- $Q = \{q_0, q_1, q_2\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{\#, A, B\}$, $F = \{q_2\}$.
- Transitions:

$$\delta(q_0, a, \varepsilon) = \{\langle q_0, A \rangle\} \qquad \delta(q_0, b, \varepsilon) = \{\langle q_0, B \rangle\}$$
$$\delta(q_0, c, \varepsilon) = \{\langle q_1, \varepsilon \rangle\} \qquad \delta(q_1, a, A) = \{\langle q_1, \varepsilon \rangle\}$$
$$\delta(q_1, b, B) = \{\langle q_1, \varepsilon \rangle\} \qquad \delta(q_1, \varepsilon, \#) = \{\langle q_2, \# \rangle\}$$



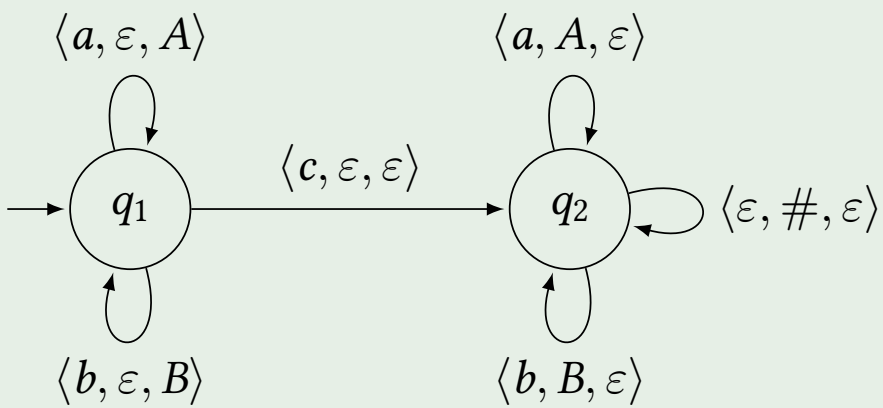
PDA: Definition (7)

Example

PDA $M_2 = \langle Q, \Sigma, \Gamma, \delta, q_0, \#, F \rangle$ for $N(M_2) = \{wcw^R \mid w \in \{a, b\}^*\}$

- $Q = \{q_0, q_1\}$, $\Sigma = \{a, b, c\}$, $\Gamma = \{\#, A, B\}$, $F = \emptyset$.
- Transitions:

$$\delta(q_0, a, \varepsilon) = \{\langle q_0, A \rangle\} \qquad \delta(q_0, b, \varepsilon) = \{\langle q_0, B \rangle\}$$
$$\delta(q_0, c, \varepsilon) = \{\langle q_1, \varepsilon \rangle\} \qquad \delta(q_1, a, A) = \{\langle q_1, \varepsilon \rangle\}$$
$$\delta(q_1, b, B) = \{\langle q_1, \varepsilon \rangle\} \qquad \delta(q_1, \varepsilon, \#) = \{\langle q_1, \varepsilon \rangle\}$$



PDA: Definition (8)

Deterministic PDA

A PDA $M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$ is a **deterministic PDA** (DPDA) iff

- for all $q \in Q, Z \in \Gamma, a \in \Sigma \cup \{\varepsilon\}$: $|\delta(q, a, Z)| \leq 1$

and

- for all $q \in Q, Z \in \Gamma$: if $\delta(q, \varepsilon, Z) \neq \emptyset$, then $\delta(q, a, Z) = \emptyset$ for all $a \in \Sigma$

(Note that this assumes the definition where
$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathcal{P}_{fin}(Q \times \Gamma^*)$,
i.e., ε as a third argument is not allowed.)

Examples for DPDA: M_1 and M_2 from the previous slides.

The class of languages accepted by DPDAs is smaller than the class accepted by (non-deterministic) PDAs.

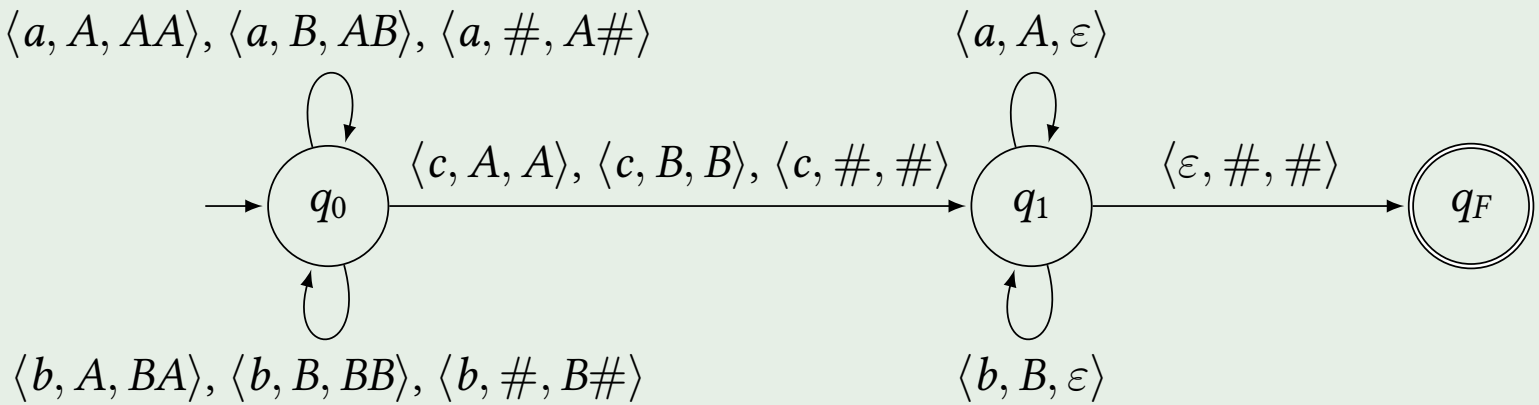
Example of a language that requires a non-determinstic PDA:

$$\{ww^R \mid w \in \{a, b\}^*\}$$

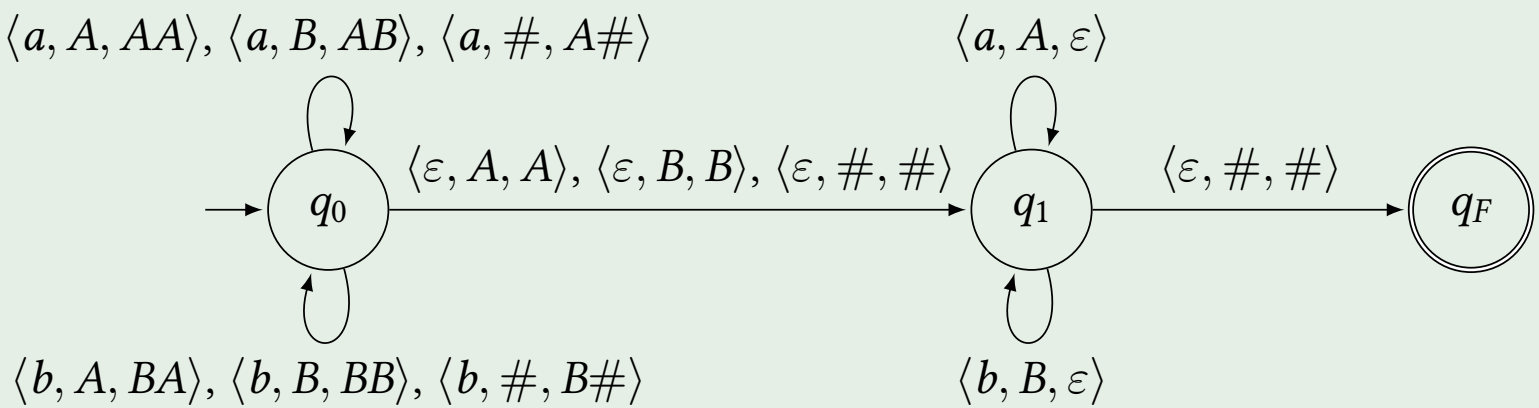
PDA: Definition (9)

Deterministic and non-deterministic PDA

Deterministic PDA for $\{wcw^R \mid w \in \{a, b\}^*\}$:



Non-deterministic PDA for $\{ww^R \mid w \in \{a, b\}^*\}$:



PDA and CFG (1)

For each CFL L, there is a PDA M with $L = N(M)$:

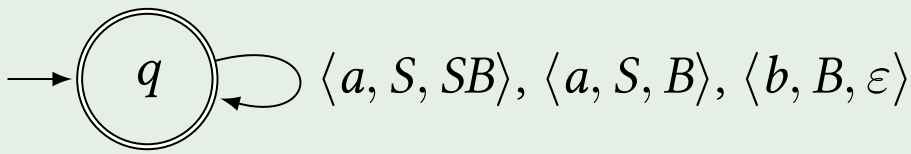
- Assume that $\varepsilon \notin L$
- $L = L(G)$ for a CFG $G = \langle N, T, P, S \rangle$ in GNF
- $M = \langle \{q\}, T, N, \delta, q, S, \emptyset \rangle$ with $\langle q, \gamma \rangle \in \delta(q, a, A)$ iff $A \to a\gamma \in P$
- The automaton simulates leftmost derivations in G

PDA and CFG (2)

From CFG in GNF to PDA

CFG: $S \to aSB \mid aB, B \to b$

Corresponding PDA with S initial stack symbol:



Run of this automaton:

state	remaining input	stack	(top on the left)
q	aabb	S	
q	abb	SB	
q	bb	BB	
q	b	B	
q	–	–	

PDA and CFG (3)

Two other possibilities to construct a PDA for a CFG $\langle N, T, P, S\rangle$:

Top-down, LL-PDA: We assume that $\langle N, T, P, S\rangle$ is not left-recursive. Then the following PDA accepts the language generated by $\langle N, T, P, S\rangle$:

- start with stack $\#$ and q_0
- $\delta(q_0, \varepsilon, \#) = \{\langle q_1, S\#\rangle\}$
- $\langle q_1, \alpha\rangle \in \delta(q_1, \varepsilon, A) \qquad \forall A \to \alpha \in P$
- $\langle q_1, \varepsilon\rangle \in \delta(q_1, a, a) \qquad \forall a \in T$
- $\delta(q_1, \varepsilon, \#) = \{\langle q_F, \varepsilon\rangle\}$

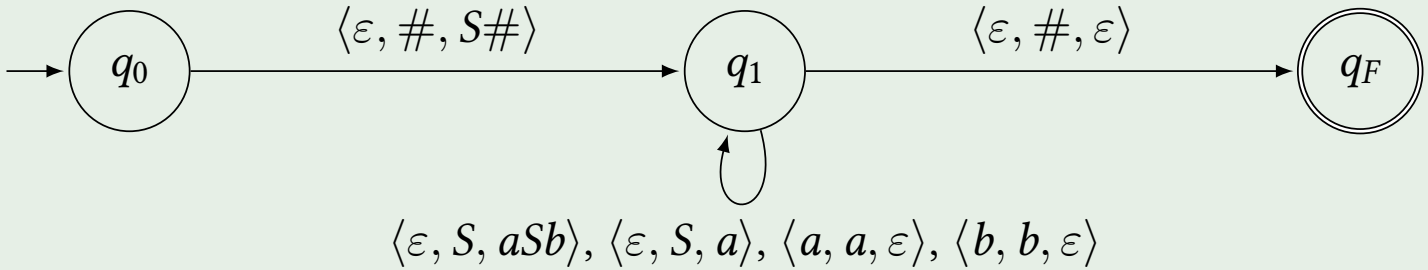
This PDA creates a leftmost derivation when keeping track of the predicted productions in a successful run.

PDA and CFG (4)

LL-PDA

CFG: $S \to aSb \mid a$

Corresponding PDA with # initial stack symbol:



Run of this automaton:

state	remaining input	stack (top on the left)
q_0	aab	#
q_1	aab	$S\#$
q_1	aab	$aSb\#$
q_1	ab	$Sb\#$
q_1	ab	$ab\#$
q_1	b	$b\#$
q_1	–	#
q_F	–	–

PDA and CFG (5)

Bottom-up, LR-PDA: We assume that $\langle N, T, P, S\rangle$ does not allow for any loops $A \overset{+}{\Rightarrow} A$. Then the following PDA accepts the language generated by $\langle N, T, P, S\rangle$:

- start with stack $\#$ and q_0
- $\langle q_0, aZ\rangle \in \delta(q_0, a, Z) \qquad \forall a \in T, Z \in N \cup T \cup \{\#\}$
- $\langle q_0, A\rangle \in \delta(q_0, \varepsilon, \alpha^R) \qquad \forall A \to \alpha \in P$

  (α^R denotes RHS of production $A \to \alpha$ in reverse order)
- $\langle q_1, \varepsilon\rangle \in \delta(q_0, \varepsilon, S)$
- $\langle q_F, \varepsilon\rangle \in \delta(q_1, \varepsilon, \#)$

For this, we use a slightly modified extension of PDA which allows also to pop more than one symbol from the stack in a single transition step. This form of PDA is equivalent to the one we defined above.

A run of an LR-PDA for a CFG creates a rightmost derivation (in reverse order)

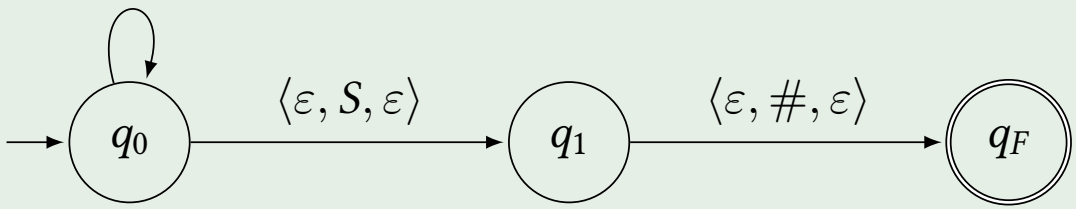
PDA and CFG (6)

LR-PDA

CFG: $S \to aSb \mid a$

Corresponding PDA with # initial stack symbol:

$\langle a, \varepsilon, a\rangle$, $\langle b, \varepsilon, b\rangle$, $\langle \varepsilon, a, S\rangle$, $\langle \varepsilon, bSa, S\rangle$



Run of this automaton:

state	remaining input	stack (top on the left)
q_0	aab	$\#$
q_0	ab	$a\#$
q_0	b	$aa\#$
q_0	b	$Sa\#$
q_0	–	$bSa\#$
q_0	–	$S\#$
q_1	–	$\#$
q_F	–	–

PDA and CFG (7)

For each PDA M with $L = N(M)$: L is a context-free language.

Construction of equivalent CFG for given PDA
$M = \langle Q, \Sigma, \Gamma, \delta, q_0, Z_0, F \rangle$:

- Terminals: Σ
- Nonterminals: S and all $[q_1, Z, q_2]$ with $q_1, q_2 \in Q$, $Z \in \Gamma$
- Start symbol: S
- Productions:
  - $S \to [q_0, Z_0, q]$ for every $q \in Q$ and
  - $[q, A, q_{m+1}] \to a[q_1, B_1, q_2][q_2, B_2, q_3] \ldots [q_m, B_m, q_{m+1}]$ for $q, q_1, \ldots, q_{m+1} \in Q$, $a \in \Sigma \cup \{\varepsilon\}$, $A, B_1, \ldots, B_m \in \Gamma$ such that $\langle q_1, B_1 \ldots B_m \rangle \in \delta(q, a, A)$
  - $[q, A, q_1] \to a$ if $\langle q_1, \varepsilon \rangle \in \delta(q, a, A)$

It holds: $[q_1, A, q_2] \overset{*}{\Rightarrow} w$ iff $(q_1, w, A) \vdash^* (q_2, \varepsilon, \varepsilon)$

Hopcroft, J. E. and Ullman, J. D. (1979). *Introduction to Automata Theory, Languages and Computation.* Addison Wesley.

Hopcroft, J. E. and Ullman, J. D. (1994). *Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie.* Addison Wesley, 3. edition.