COS210 - Theoretical Computer Science
Pushdown Automata

Pushdown Automata

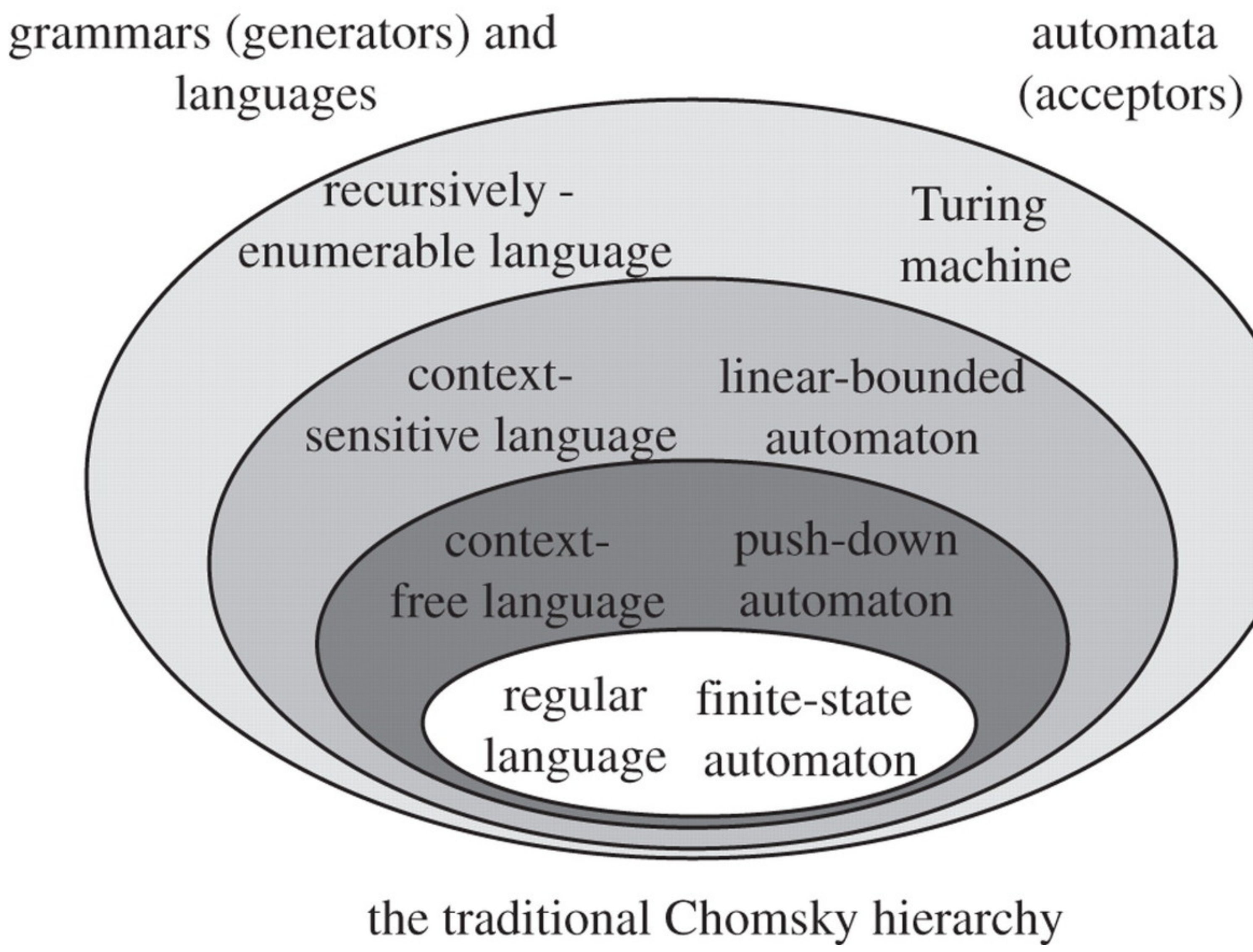
We will introduce the concept of **pushdown automata** (PDA)

- **Deterministic** pushdown automata
- **Non-deterministic** pushdown automata

Unlike DFAs and NFAs, non-deterministic pushdown automata can describe **more** languages than deterministic ones

We will prove: Every **context-free grammar** can be transformed into an **equivalent non-deterministic PDA**

Descriptive Power of Languages and Automata



Finite Automaton M_{FA} vs Pushdown Automaton M_{PDA}

Finite automaton $M_{FA} = (Q, \Sigma, \delta, q, F)$

- states Q, input alphabet Σ, transition function δ, initial state q, accepting states F

Pushdown Automaton $M_{PDA} = (Q, \Sigma, \Gamma, \delta, q)$

- Q set of states
- Σ input alphabet (also called tape alphabet)
- Γ **stack alphabet**
- δ transition function
- q initial state

no set of accepting states, acceptance is defined implicitly for PDAs

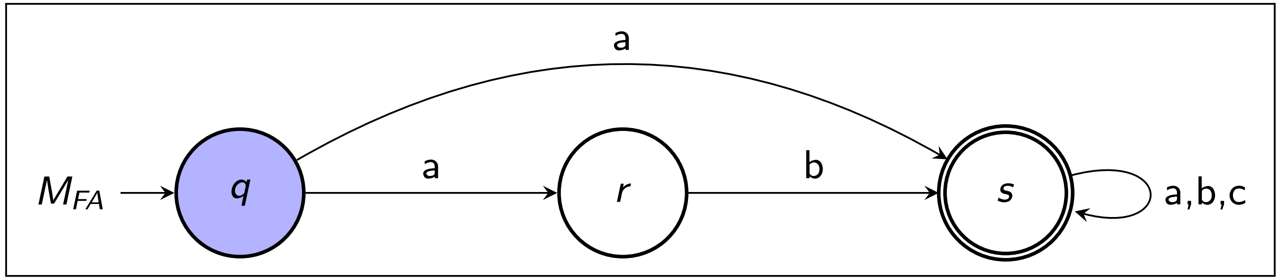
As a FA, a PDA can **process strings** over the input alphabet $\Sigma = \{a, b, c, \ldots\}$

Additionally, a PDA makes use of a **stack** where symbols from the stack alphabet $\Gamma = \{A, B, C, \ldots\}$ can be **pushed** and **popped**

Finite Automaton M_{FA} vs Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

state control

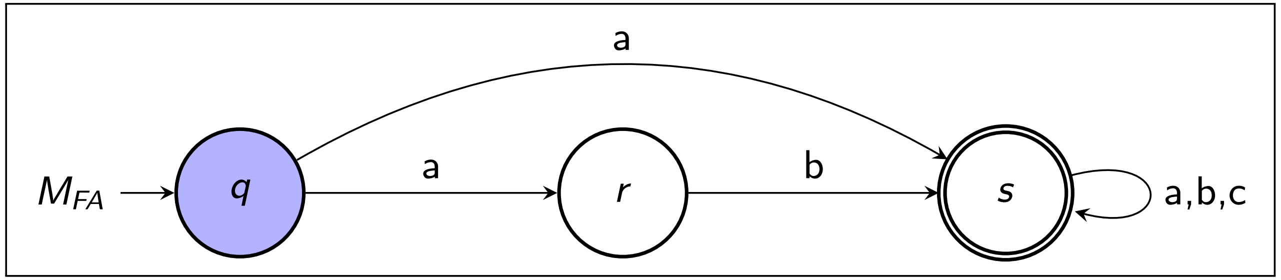


Finite Automaton M_{FA} vs Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

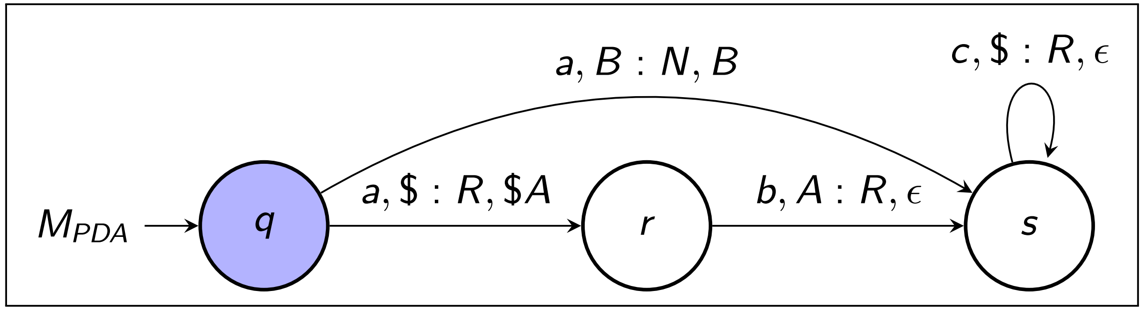
---

state control

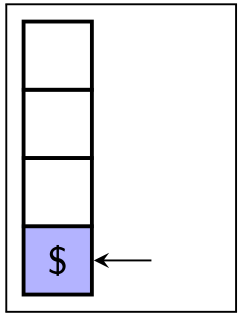


---

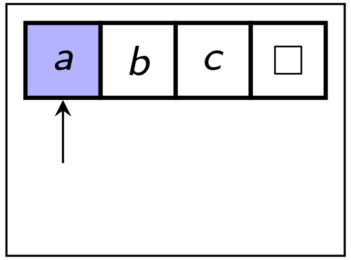
state control



stack



tape



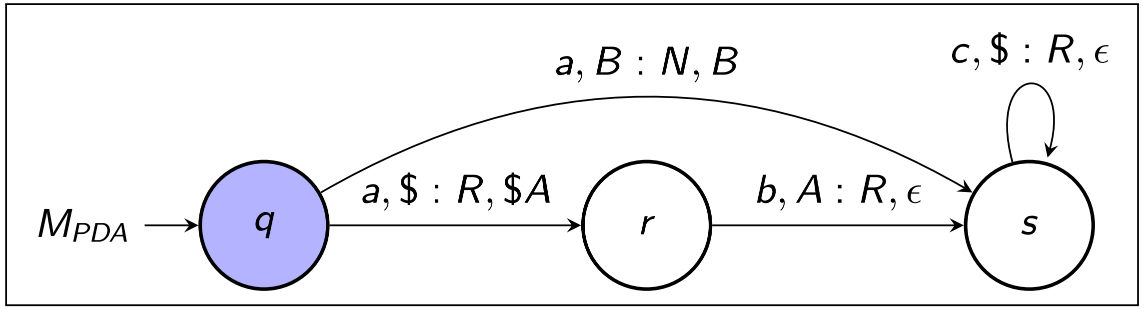
Configuration of a PDA:

- triple of state, symbol at tape head, and stack content
- current configuration: $(q, a, \$)$

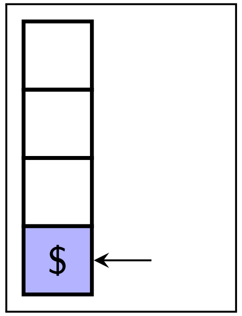
Run of a Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

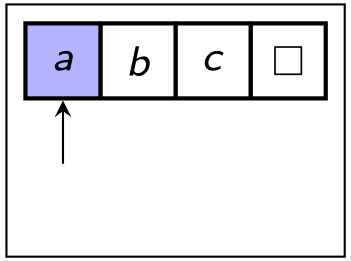
state control



stack



tape



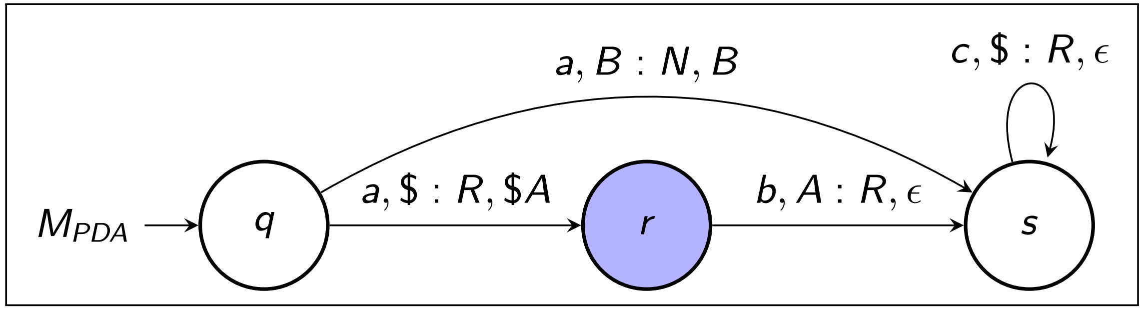
General form of PDA transitions: $r \xrightarrow{a,\, A \,:\, \tau,\, W} r'$

- $r, r' \in Q$ states
- $a \in \Sigma$ language/tape symbol
- $A \in \Gamma$ stack symbol
- $\tau \in \{R, N\}$ 'Right move' or 'No move'
- $W \in \Gamma^*$ string over stack symbols

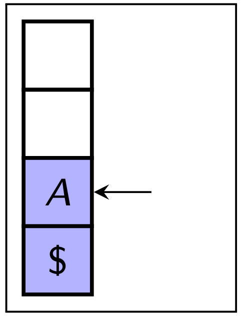
Run of a Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

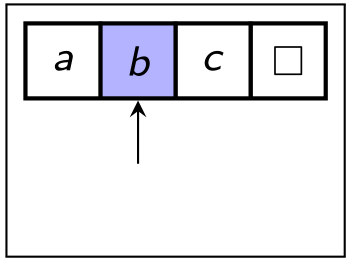
state control



stack



tape



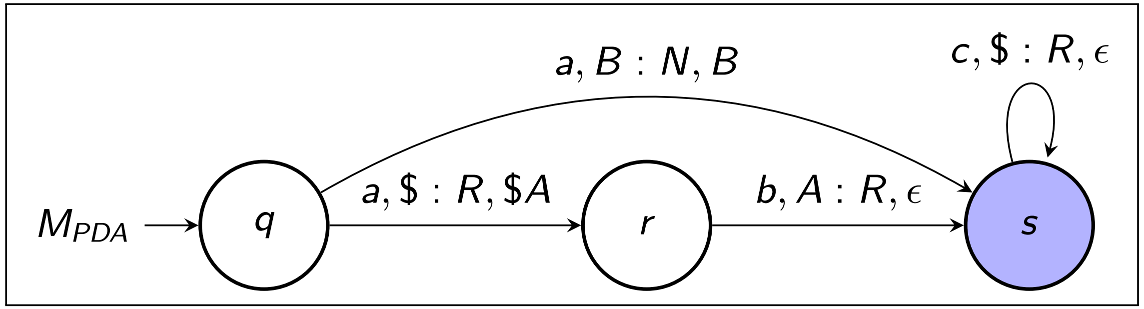
General form of PDA transitions: $r \xrightarrow{a,\ A\ :\ \tau,\ W} r'$

- $r, r' \in Q$ states
- $a \in \Sigma$ language/tape symbol
- $A \in \Gamma$ stack symbol
- $\tau \in \{R, N\}$ 'Right move' or 'No move'
- $W \in \Gamma^*$ string over stack symbols

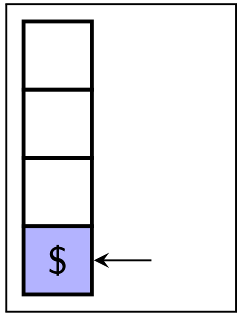
Run of a Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

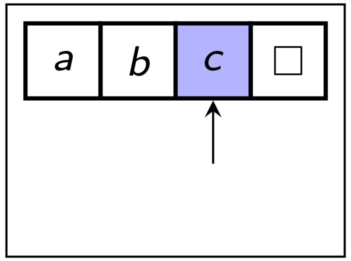
state control



stack



tape



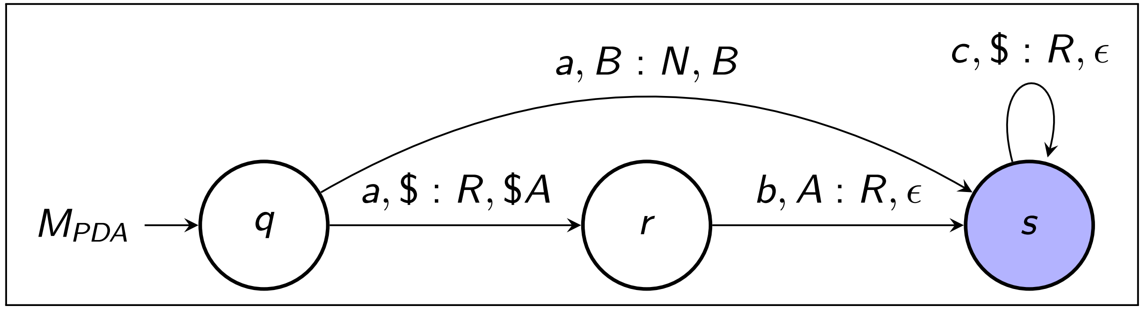
General form of PDA transitions: $r \xrightarrow{a,\, A\, :\, \tau,\, W} r'$

- $r, r' \in Q$ states
- $a \in \Sigma \cup \{\Box\}$ language/tape symbol
- $A \in \Gamma$ stack symbol
- $\tau \in \{R, N\}$ 'Right move' or 'No move'
- $W \in \Gamma^*$ string over stack symbols

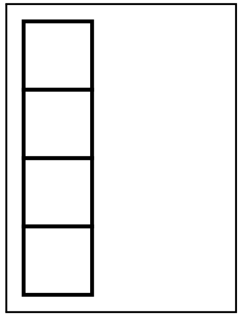
Run of a Pushdown Automaton M_{PDA}

string $w = abc$ accepted by automaton?

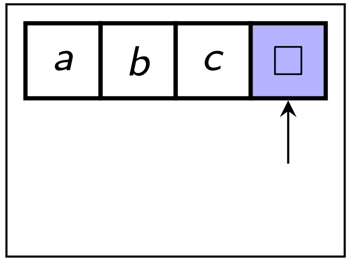
state control



stack



tape



Termination and acceptance of PDA:

- PDA **terminates** when stack becomes empty
- PDA **accepts** input string w if
  - PDA terminates for input w, and
  - at time of termination, the tape head points at $\square$

Pushdown Automata – Informal Summary

Tape and stack of a PDA

- Each cell of the tape stores one symbol from the tape alphabet Σ, or the blank symbol $\Box$ which indicates that the cell is empty
- An input string over Σ is stored on the tape followed by $\Box$, which indicates the end of the input string
- The stack contains symbols from the stack alphabet Γ
- The special symbol $\$ \in \Gamma$ indicates the bottom of the stack

Initial configuration of a PDA

- The tape head points at the left-most symbol of the input string
- The stack contains $\$$ only
- The state control is in the initial state q

Transitions of a PDA

1. PDA determines the current configuration
   - read current state r of state control
   - read symbol a at head of the tape
   - read symbol A at top of the stack
2. PDA selects a transition that is available in configuration (r, a, A)
   - in a deterministic PDA there will be exactly one available transition for each configuration
   - in a non-deterministic PDA there will be a set of available transitions
3. By taking the transition, PDA enters a new configuration
   - state control moves to successor state
   - tape head moves one position to the right or remains at current position
   - top of the stack is popped and a new string is pushed onto it

Deterministic Pushdown Automata – Definition

Definition

A **deterministic pushdown automaton** is a 5-tuple $M = (Q, \Sigma, \Gamma, \delta, q)$, where

- Q is a finite set, whose elements are called **states**
- Σ is a finite set, called the **tape alphabet**
  the blank symbol $\Box$ is not contained in Σ
- Γ is a finite set, called the **stack alphabet**
  Γ contains the special symbol \$
- q is an element of Q, called the **initial state**
- δ is called the **transition function**, which is a function

$$\delta : Q \times (\Sigma \cup \{\Box\}) \times \Gamma \rightarrow Q \times \{N, R\} \times \Gamma^*$$

Deterministic Pushdown Automata – Transition Function

$$\delta : Q \times (\Sigma \cup \{\Box\}) \times \Gamma \to Q \times \{N, R\} \times \Gamma^*$$

Given a current configuration, δ allows to determine the **unique successor configuration**

$$\delta(r, a, A) = (r', \tau, W)$$

$$r \in Q, a \in \Sigma \cup \{\Box\}, A \in \Gamma \qquad r' \in Q, \tau \in \{R, N\}, W \in \Gamma^*$$

Instruction notation:

$$raA \to r'\tau W$$

A deterministic PDA M accepts a string w if the unique run over w is accepting (M terminates, at time of termination tape head points at $\Box$)

Non-Deterministic Pushdown Automata – Definition

Definition

A **non-deterministic pushdown automaton** is a 5-tuple $M = (Q, \Sigma, \Gamma, \delta, q)$, where

- Q is a finite set, whose elements are called states
- Σ is a finite set, called the tape alphabet
  the blank symbol $\Box$ is not contained in Σ
- Γ is a finite set, called the stack alphabet
  Γ contains the special symbol \$
- q is an element of Q, called the initial state
- δ is called the **transition function**, which is a function

$$\delta : Q \times (\Sigma \cup \{\Box\}) \times \Gamma \to \mathcal{P}_f\left(Q \times \{N, R\} \times \Gamma^*\right)$$

where $\mathcal{P}_f(Q \times \{N, R\} \times \Gamma^*)$ is the set of all **finite** subsets of $Q \times \{N, R\} \times \Gamma^*$

Non-Deterministic PDA – Transition Function

$$\delta : Q \times (\Sigma \cup \{\square\}) \times \Gamma \to \mathcal{P}_f\left(Q \times \{N, R\} \times \Gamma^*\right)$$

Given a current configuration, δ allows to determine the **set successor configurations**

$$\delta(r, a, A) = \{(r'_1, \tau_1, W_1), \ldots, (r'_n, \tau_n, W_n)\}$$

$r \in Q, a \in \Sigma \cup \{\square\}, A \in \Gamma \qquad r'_i \in Q, \tau_i \in \{R, N\}, W_i \in \Gamma^*, \forall 1 \leq i \leq n$

Instruction notation:

$$raA \to r'_1\tau_1W_1$$
$$\ldots$$
$$raA \to r'_n\tau_nW_n$$

A non-deterministic PDA M accepts a string w if there exists an accepting run over w (M terminates, at time of termination tape head points at $\square$)

Deterministic Pushdown Automata – Example

Construct a DPDA M that accepts the language

$$L = \{w : w \text{ is a string of properly nested brackets}\}$$

over the alphabet $\Sigma = \{a, b\}$ where $a =$ (and $b =$)

Strings $w \in L$ can be equivalently defined as follows:

- in w the number of **a**'s is equal to the number of **b**'s
- in each prefix of w the number of **a**'s is greater than or equal to the number of **b**'s

a	a	b	a	b	b	a	b	$\square$

Construction idea for M:

- each time an **a** is read on the tape, **push** a symbol S onto the stack
- each time a **b** is read on the tape, **pop** a symbol from the stack

If there is a prefix with more **b**'s, then \$ will be popped before $\square$ is read

Deterministic Pushdown Automata – Example

DPDA $M = (Q, \Sigma, \Gamma, \delta, q)$ with $Q = \{q\}$, $\Sigma = \{a, b\}$, $\Gamma = \{\$, S\}$, and δ:

1. $qa\$ \rightarrow qR\$S$: **a** is read, **push S**
2. $qaS \rightarrow qRSS$: **a** is read, **push S**
3. $qbS \rightarrow qR\epsilon$: **b** is read and **S** is **on top** of stack, **pop S**
4. $qb\$ \rightarrow qN\epsilon$: **b** is read and \$ is **on top**, **pop** \$ and **no move**

   more **b**'s than **a**'s have been read,
   termination in **non-accepting** configuration

5. $q\Box\$ \rightarrow qN\epsilon$: $\Box$ is read and \$ is **on top**, **pop** \$ and **no move**

   entire string has been read when stack becomes empty,
   termination in **accepting** configuration

6. $q\Box S \rightarrow qNS$: $\Box$ is read and **S on top**, **no stack operation** and **no move**

   entire string has been read but stack still contains an **S**,
   PDA will loop forever in **non-accepting** configuration

Deterministic Pushdown Automata – Example 2

Construct a DPDA M that accepts the language

$$L = \{0^n 1^n \in \{0,1\}^* : n \geq 0\} \text{ over } \Sigma = \{0,1\}$$

Construction idea for M:

- M uses two states q_0 and q_1, where q_0 is the initial state
- for each **0** that is read, **push** a symbol S onto the stack and stay in state q_0
- when first **1** is read, **pop** a symbol from stack and **switch** to state q_1

  From then on:
  - for each **1** that is read, **pop** a symbol from stack and **stay** in state q_1
  - if a **0** is read **again**, loop forever in **non-accepting** configuration

0	0	0	0	1	1	1	1	□

Deterministic Pushdown Automata – Example 2

DPDA $M = (Q, \Sigma, \Gamma, \delta, q_0)$ with $Q = \{q_0, q_1\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{\$, S\}$, and δ:

1)	$q_0 0\$ \to q_0 R\$S$	push S onto the stack
2)	$q_0 0S \to q_0 RSS$	push S onto the stack
3)	$q_0 1\$ \to q_0 N\$$	first symbol in the input is 1; loop forever
4)	$q_0 1S \to q_1 R\epsilon$	first 1 is encountered
5)	$q_0 \square\$ \to q_0 N\epsilon$	input string is empty; accept
6)	$q_0 \square S \to q_0 NS$	input only consists of 0s; loop forever
7)	$q_1 0\$ \to q_1 N\$$	0 to the right of 1; loop forever
8)	$q_1 0S \to q_1 NS$	0 to the right of 1; loop forever
9)	$q_1 1\$ \to q_1 N\$$	too many 1s; loop forever
10)	$q_1 1S \to q_1 R\epsilon$	pop top symbol from the stack
11)	$q_1 \square\$ \to q_1 N\epsilon$	accept
12)	$q_1 \square S \to q_1 NS$	too many 0s; loop forever