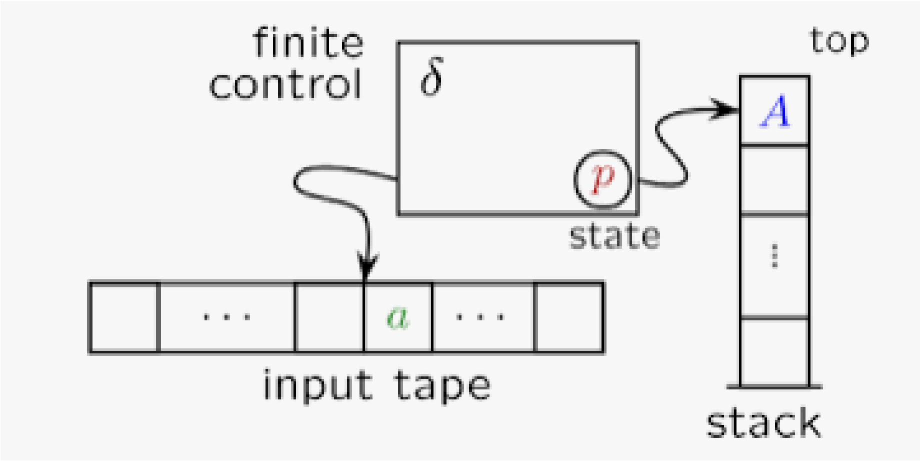
Chapter 7: Pushdown Automata

Plan

1. Definition
2. Configuration, Transition, and Computation
3. Acceptance Criteria
4. Deterministic Pushdown Automata

Automates à piles

Pushdown Automata (PDA) are abstract machines that determine whether or not a word belongs to a language. The languages recognized by PDA are context-free languages (Type 2). In addition to the components of finite state automata (FSA), PDA have a stack for storage



Definition

A **pushdown automaton** is a 7-tuple:

A = <X, Y, S, S_0, F, I, #>

- #: Empty stack symbol
- **X**: Input alphabet
- **Y**: Stack alphabet
- **S**: Set of states
- **S_0**: Initial state
- **F**: Set of final states, with **F ⊆ S**
- **I**: Set of instructions (transitions), defined as:

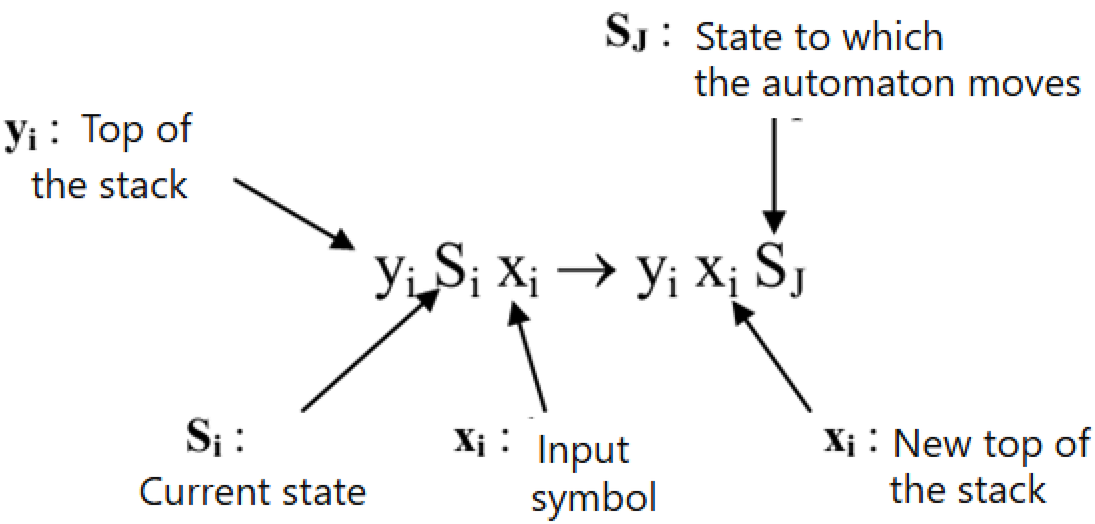
$$I : S \times (X \cup \varepsilon) \times Y \rightarrow S \times Y^*$$

stack operations

- **Push (Empilement)**
- **Pop (Dépilement)**
- **Do nothing (Neither push nor pop)**

stack operations

- **Push (Empilement)**



$S_i, S_J \in S$

$y_i \in Y \cup \{\#\}$

$x_i \in X \cup \{\#\}$

- If we are in state Si , and the top of the stack is yi , and the symbol under the read head is xi , then we push xi onto yi (i.e., xi becomes the new top of the stack), and the automaton moves to state SJ .

stack operations

- **Pop Operation**

$y_i\ S_i\ x_i \rightarrow S_J$ Stack top is changed $S_i, S_J \in S$

- **Neither Push nor Pop**

$\mathbf{y_i\ S_i\ x_i \rightarrow y_i\ S_J} :$ No push, no pop.

Example

- $L = \{a^i b^i, i \geq 1\}$

The steps for recognizing the language $\{a^i b^i, i \geq 1\}$ by a PDA could be as follows:

- ✓ Read the a's, store them in the stack, and do not change state;
- ✓ Upon encountering the first b, pop an a and change state;
- ✓ Pop one a for each b encountered;
- ✓ If the a's in the stack are exhausted at the same time as the b's are finished being read, then the word belongs to the language.

Example 1 :

$$L_1 = \{a^i b^i, i > 0\}$$

A<X,Y,S,S_0,F,I,#>

X={a,b}, S={S_0,S_1,S_f}

$\# S_0 a \rightarrow \# a S_0$

$a S_0 a \rightarrow a a S_0$

$a S_0 b \rightarrow S_1$

$a S_1 b \rightarrow S_1$

$\# S_1 \rightarrow \# S_f$

Exemple 1(suite)

$\# \, S_0 \, a \rightarrow \# \, a \, S_0$

$a \, S_0 \, a \rightarrow a \, a \, S_0$

$a \, S_0 \, b \rightarrow S_1$

$a \, S_1 \, b \rightarrow S_1$

$\# \, S_1 \rightarrow \# \, S_f$

Vérification :

$\#S_0aaabbb \vdash \#aS_0aabbb$

$\vdash \#aaS_0abbb$

$\vdash \#aaaS_0bbb$

$\vdash \#aaS_1bb$

$\vdash \#aS_1b$

$\vdash \#S_1$

$\vdash \#S_f$

Input word	Current state	Stack state
aaabbb	S_0	#
aabbb	S_0	#a
abbb	S_0	#aa
bbb	S_0	#aaa
bb	S_1	#aa
b	S_1	#a
ε	S_1	# empty stack

Example 2 :

For $L_2 = \{a^i b^i, i \geq 0\}$, we add $\# S_0 \rightarrow \# S_f$ to recognize the empty word.

Example 3 :

$$L_3 = \{w \ / \ |w|_a = |w|_b\}$$

$\# \ S_0 \ a \rightarrow \# \ a \ S_0$

$\# \ S_0 \ b \rightarrow \# \ b \ S_0$

$a \ S_0 \ a \rightarrow a \ a \ S_0$

$a \ S_0 \ b \rightarrow S_0$

$b \ S_0 \ a \rightarrow S_0$

$b \ S_0 \ b \rightarrow b \ b \ S_0$

$\# \ S_0 \rightarrow \# \ S_f$

Exemple 4 :

$L_4 = \{a^i b^i, i > j \; et \; j \geq 0\}$

$\# S_0 a \rightarrow \# a S_0$

$a S_0 a \rightarrow a a S_0$

$a S_0 b \rightarrow S_1$

$a S_1 b \rightarrow S_1$

$a S_1 \rightarrow S_0$

$a S_1 \rightarrow S_f$

$a S_0 \rightarrow S_f$

Verification :

$\#S_0aabbb \vdash \#aS_0abbb$

$\vdash \#aaS_0bbb$

$\vdash \#aS_1bb$

$\vdash \#S_1b$ **Blocked** $\Rightarrow \mathbf{w} \notin \mathbf{L_4}$

Definition of Configuration

A configuration of the PDA, at a given moment, is defined by the content of the stack, the current state of the PDA, and the remaining input word to be read.

A **configuration** is a triplet (y, S_J, w') where:

- y is the content of the stack
- S_J is the current state of the stack
- w' is the remaining input word

Initial Configuration:

(# , S_0 , w)

S_0: the initial state

w: the word to be recognized, $w \in L(A_p)$

Final Configuration:

(y , S_f , ε)

$S_f \in F$ (a final state)

ε: the empty word

Word recognized by a pushdown automaton

PDAs (Pushdown Automata) can recognize languages in two different modes:

- **Recognition by final state**
- **Recognition by empty stack**.

w is recognized by a pushdown automaton if and only if $\#S_0 w \vdash_{A_p}^{*} yS_f$

$$L(A_p) = \{w \in X^* \text{ such that } \#S_0 w \vdash_{A_p}^{*} yS_f,\ S_f \in F\}$$

Example 5 :

$\boldsymbol{L_5 = \{a^i b^i\ , i} < j\}$

$\# \, S_0 \, a \rightarrow \# \, a \, S_0$

$a \, S_0 \, a \rightarrow a \, a \, S_0$

$a \, S_0 \, b \rightarrow S_1$

$a \, S_1 \, b \rightarrow S_1$

$\# \, S_1 \, b \rightarrow \# \, S_2$

$\# \, S_2 \, b \rightarrow \# \, S_2$

$\# \, S_0 \, b \rightarrow \# \, S_1$

$\# \, S_2 \rightarrow \# \, S_f$

Definition – Empty Stack Automaton–

- a final configuration where the stack is empty. The pushdown automaton is then said to be an empty stack automaton.

Recognition by Empty Stack

The notion of a final state does **not exist** in this type of recognition. Thus, a pushdown automaton (PDA) that recognizes by **empty stack** is defined by a **sextuple** (i.e., **without a set of final states**).

The language recognized by empty stack by an automaton A is defined as:

$$L(A) = \{\omega \in X^* \mid (\#, q_0, \omega) \vdash^* (\varepsilon, q, \varepsilon)\}$$

Example 6

- $L = \{wcw^R \text{ , } w \in \{a, b\}^*\}$

$\# S_0 a \to \# a S_0$
$\# S_0 b \to \# b S_0$
$a S_0 a \to a a S_0$
$b S_0 a \to b a S_0$
$a S_0 b \to a b S_0$
$b S_0 b \to b b S_0$
$a S_0 c \to a S_1$
$b S_0 c \to b S_1$
$a S_1 a \to S_1$
$b S_1 b \to S_1$
$\# S_1 \to \# S_f$
$\# S_0 c \to \# S_1$

Deterministic and Non-Deterministic Pushdown Automata (PDAs)

- There are **two cases of non-determinism** for PDAs:

1. For the **same stack top**, **same state**, and **same input symbol**, there exist **at least two transitions**;

2. For the **same stack top** and **same state**, the automaton has the **option to read or not read** from the input tape.

Definition

A **PDA** (Pushdown Automaton) is said to be **deterministic** if and only if for every triple (u, q, a) defined in $X \times Q \times \Gamma$, the transition function δ assigns **at most one pair** (α, p), and if $\delta(y_i, q, x_i)$ is defined, then **there is no transition** $\delta(y_i, q, \varepsilon)$.

Remark: There exist **context-free languages** for which **no deterministic PDA** can recognize them.

Theorem: If a language L is recognized by a **deterministic pushdown automaton**, then there exists a **non- ambiguous context-free grammar** that generates L.