Automata Theory - Lecture 8

Models of Computation – Pushdown Stack Machines

Lecturer: Martha Gichuki

Lecture learning outcomes

At the end of the lecture you will be able to:

(i) Describe Pushdown Stack Machines

(ii) Describe Pushdown Stack Machines using State Diagrams

(iii) Formally describe various Pushdown Stack Machines

Extensions of Finite Automata

The family of languages accepted by finite automata is called the family of regular languages.

More powerful automata can accept more complicated languages. Such automata include: -

Pushdown Automata (PDA)

Such machines are identical to Deterministic Finite Automatons (DFAs) or Non-deterministic Finite Automatons (NFAs), except that they additionally carry memory in form of a **stack.**

The transition function will now also depend on the symbol(s) on top of the stack, and will specify how the stack is to be changed at each transition.

- **Push** action is equivalent to **writing a symbol** into the stack
- **Pop action is equivalent to** reading a symbol from the stack

Why is the symbol on top of the stack important?

- The symbol on top of the stack will specify how the stack is to be changed at each transition.
- The symbol is replaced with another symbol meaning we pop it out and push in another symbol to take its place.

What language does a PDA recognize?

- Non-deterministic PDAs accept languages known as the Context-Free Languages (CFL).
- A Context Free Grammar (CFG) defines a Context Free Language (CFL) which is recognized by a Push Down Automata (PDA)

FORMAL DEFINITION OF A PUSHDOWN AUTOMATON

The formal definition of a PDA is similar to that of a finite automaton, except for the stack. At the heart of any formal definition of an automaton is the transition function, for that describes its behaviour.

A Pushdown Automaton is a 6-tuple $(Q, \sum, \Gamma, \delta, q_0, F)$, where Q, $\sum$, Γ and F are all **finite sets** and: -

Q is the set of states

$\sum$ is the input alphabet

Γ is the stack alphabet

$\delta: Q \times \sum_\epsilon \times \Gamma_\epsilon \rightarrow P(Q \times \Gamma_\epsilon)$ is the transition function

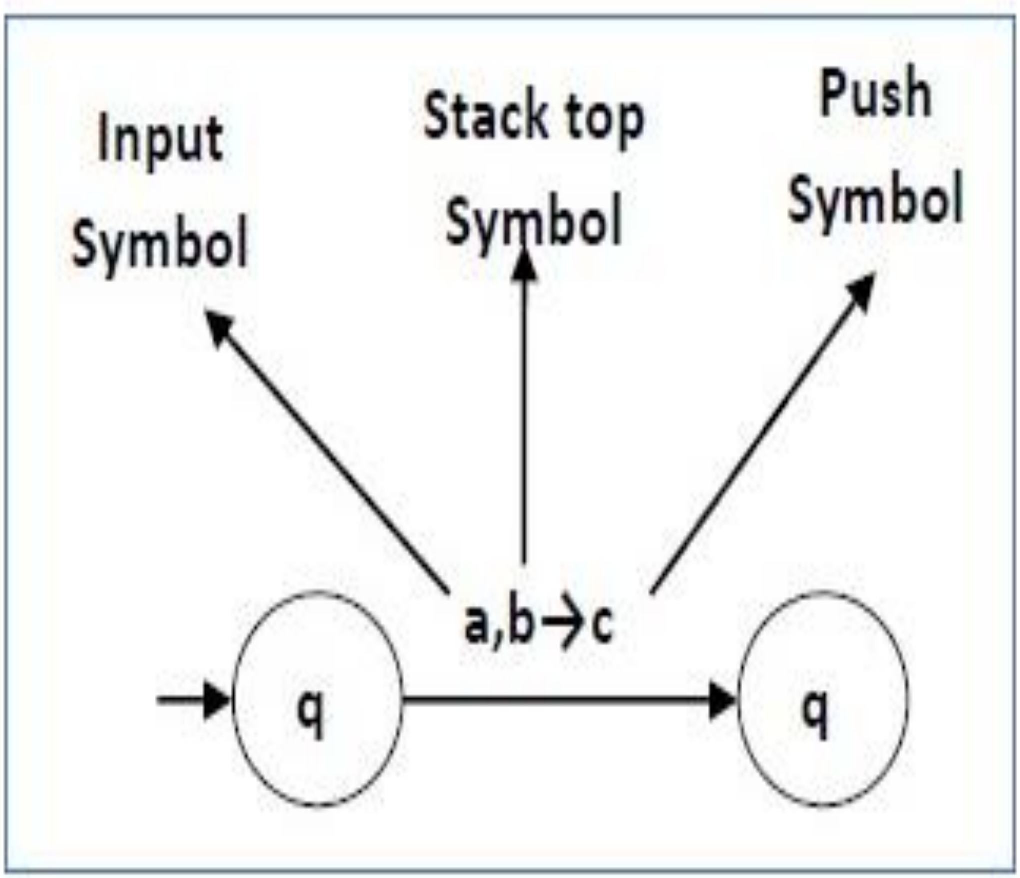
$q_0 \in Q$ is the start state,

and $F \subseteq Q$ is the set of accept states

How does the PDA Compute?

A Pushdown Automaton M = (Q, ∑, Γ, δ, q_0, F), computes as follows: -

- ✓ Generally, given the function **a, b** → **c**;
- ✓ It means that "a" is from the input alphabet and that when the machine **reads** an "a" from the input, **it *may* replace the symbol "b" on top of the stack with a symbol "c".**
- ✓ *Any of a, b, c may be empty (∈).*
- ✓ If **a is ∈**, the machine may **make this transition without reading any symbol from the input**.
- ✓ If **b is ∈**, the machine may **make this transition without reading and popping any symbol from the stack**.
- ✓ If **c is ∈**, the machine **does not write any symbol on the stack when going along this transition.**



It accepts input w if w can be written as $w = w_1 w_2 \ldots w_m$ where each $w_i \in \sum_\epsilon$ and sequences of states $r_0\, r_1, \ldots. r_m \in Q$ and strings $s_0\, s_1 \ldots s_m \in \Gamma^*$ exist that satisfy the next three conditions.

The strings s_1 represent the sequence of sack contents that M has on the accepting branch of the computation.

(i.) $r_0 = q_0$ and $s_0 = \epsilon$. This condition signifies that M starts out properly, in the start state and with an empty stack.

(ii.) For i = 0…m – 1, we have $(r_{i+1,} b) \in \delta\, (r_{i,}. W_{i+1,}\, a)$, where $s_i = a_i$ and $s_{i+1,} = b_t$ for some a, $b \in \Gamma_\epsilon$ and $t \in \Gamma^*$. This condition states that M moves properly according to the state, stack and next input symbol.

(iii.) $r_m \in F$. This condition states that an accept state occurs at the input end.

Characteristic of PDA

Given the transition: - $(q_j, B) \in \delta\, (q_i, a, A)$:

This means that the Push Down Automata (PDA) when in state q_i, reads symbol a, and having symbol A on top of the stack, is allowed to do the following: -

a) Consume a

b) POP A

c) PUSH B

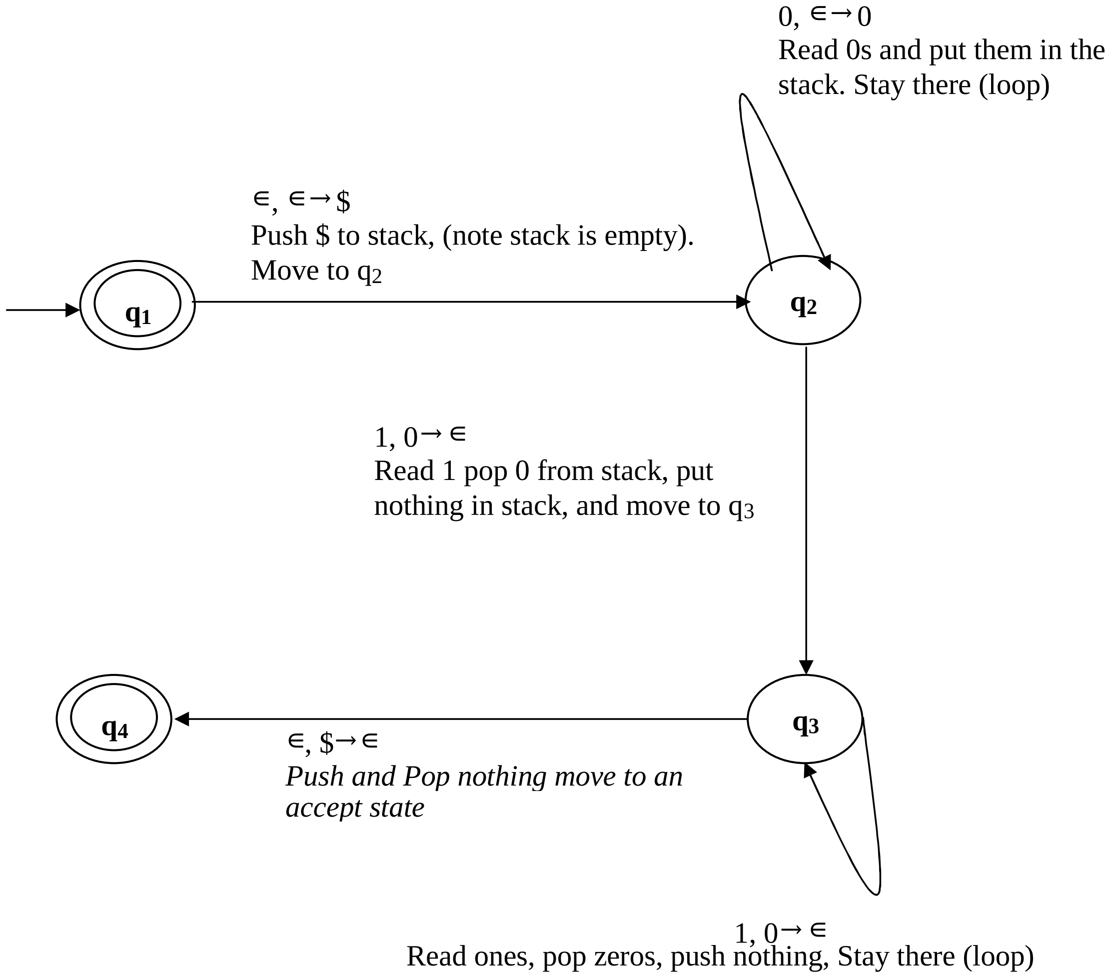
d) Change to state q_j

Example One

Consider the Push Down Automata that recognizes the language {o^n, 1^n |n≥0}

- ✓ Recall the general function **a, b → c** ;
- ✓ "a" is from the input alphabet
- ✓ when the machine **reads** an "a" from the input, **it *may* replace the symbol "b" on top of the stack with a symbol "c".**
- ✓ *Any of a, b, c may be ∈.*

We design the following state diagram for the Push Down Automata.



Source: Introduction to the theory of computation (3rd ed.), Michael, S. Boston, Cengage Learning. ISBN-13: 978-1133187790, (2012). Page 115.

Note that $ is a symbol / Mechanism to help us know that the stack is empty. Now, from the above characteristic of PDA, we can conclude that: -

- $(q_2, 0) \in \delta(q_2, 0, \in)$
- $(q_2, \$) \in \delta(q_1, \in, \in)$
  - $(q_3, \in) \in \delta(q_2, 1, 0)$

- $(q_3, \in) \in \delta(q_3, 1, 0)$
- $(q_4, \in) \in \delta(q_3, \in, \$)$

etc.

The following is the formal description of this PDA that recognizes the language {on, 1^n | $n \geq 0$}, Let M_1 be $(Q, \sum, \Gamma, \delta, q_0, F)$;

Where;

$Q = \{q_1, q_2, q_3, q_4\}$, are the states

$\sum = \{0,1\}$

$\Gamma = \{0, \$\}$

$\delta: Q \times \sum_\in \times \Gamma_\in \rightarrow P(Q \times \Gamma_\in)$ is the transition function given by the following table: -

Input	0			1			∈		
Stack	0	$	∈	0	$	∈	0	$	∈
q_1									({q_2, $})
q_2			({q_2, 0})	({q_3, ∈})					
q_3				({q_3, ∈})				({q_4, ∈})	
q_4									

$q_0 \in Q = q_1$ is the start state,

and $F \subseteq Q = \{q_1, q_4\}$ is the set of accept states

Question: Indicate the set of steps followed by the PDA upon reading string 00011

- $(q_2, \$) \in \delta (q_1, \in, \in)$
- $(q_2, 0) \in \delta (q_2, 0, \in)$
- $(q_2, 0) \in \delta (q_2, 0, \in)$
- $(q_2, 0) \in \delta (q_2, 0, \in)$
- $(q_3, \in) \in \delta (q_2, 1, 0)$
- $(q_3, \in) \in \delta (q_3, 1, 0)$
- $(q_3, \in) \in \delta (q_3, 1, 0)$
- $(q_4, \in) \in \delta (q_3, \in, \$)$

Does the PDA recognize the string 000111?

- Yes it does because the a symbol in the equation a, b → c ; is ∈, the machine while at state $q_3$may make a transition landing at state q_4 without reading any symbol from the input.
- State q_4 being the accept state means that the machine accepts string 000111

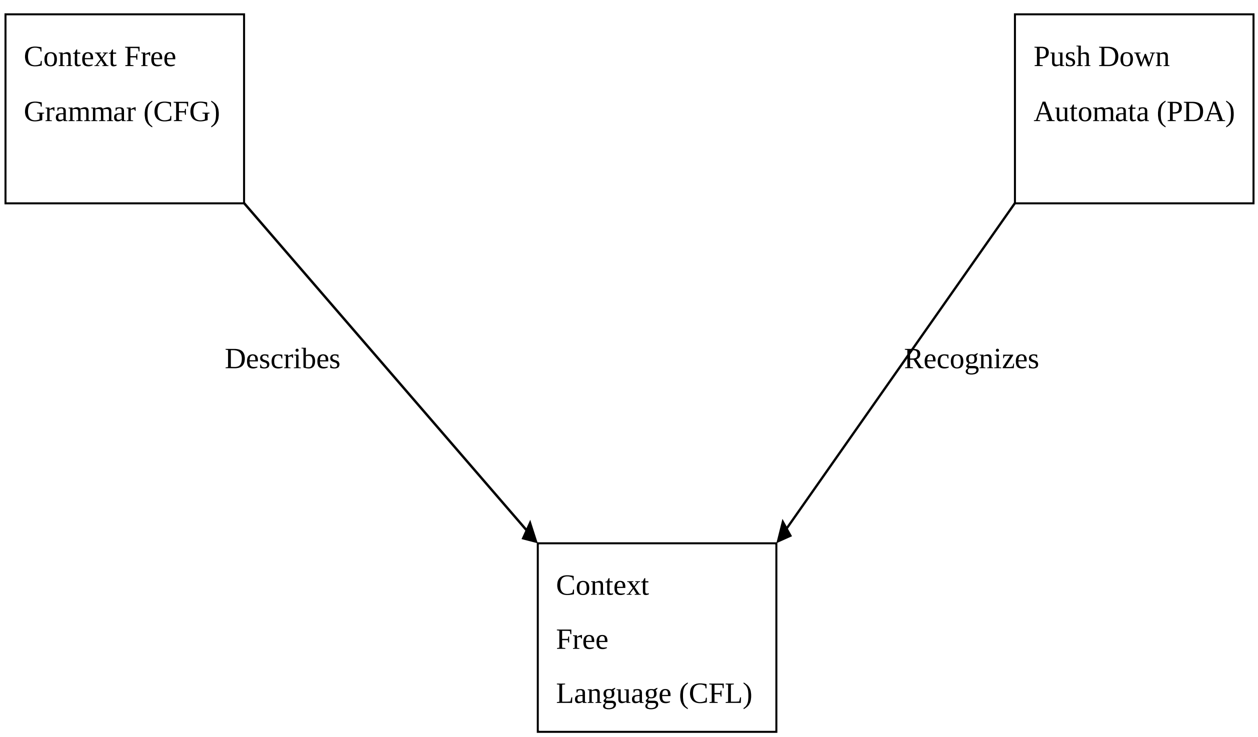
Summary

At the heart of every machine definition is the transition Function

Finite Automata can either be DFA or NFA.

There are other extensions of Finite Automata including the Push Down Automata (PDA) and the Turing Machine (TM) among others.

A Context Free Grammar (CFG) defines a Context Free Language (CFL) which is recognized by a Push Down Automata (PDA)



References

1 Rowan G. & John T., (2009), *Discrete Mathematics: Proofs, Structures and Applications*, CRC Press, ISBN: 9781439812808.

2 W. D. Wallis (2003), *A Beginners Guide to Discrete Mathematics*, Springer Science & Business Media, ISBN: 978-0817642693.

3 Introduction to the theory of computation (3rd ed.), Michael, S. Boston, Cengage Learning. ISBN-13: 978-1133187790, (2012).

4 Introduction to languages and the theory of computation (3rd ed.), Martin, J., New York: McGraw-Hill. ISBN-13: 978-0072322002, (2002)