Lokmanya Tilak Jankalyan Shikshan Sanstha's

Priyadarshini Bhagwati College of Engineering, Nagpur

An Autonomous Institution Affiliated to R.T.M. Nagpur University, Nagpur

Accredited by NAAC Grade 'A'

Harpur Nagar, Umred Road, Nagpur- 440024

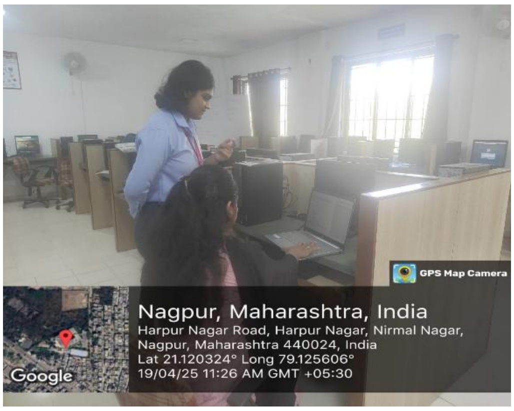
Department of Computer Science and Engineering

Session: 2024-25 (Even Sem)

Report on Seminar

Name of the Students	108-Hansika M. Kakpure
Date of Seminar	29/03/2025
Semester	Tech IV sem CSE
Subject	Theory of Computation
Topic of Seminar	Pushdown Automata
Contents of the Seminar	1. Introduction 2. Decoding the Formal Definition of a PDA 3. PDA in Action: Recognizing L = {0^n 1^n | n >= 0} 4. Instantaneous Description (ID): Tracing Computation 5. Transition diagram -DFA, NFA 6. Example 7. Difference 8. Conversion 9. Conclusion

Photos of the seminar



Signature of Students

Wrapping Up: PDA, CFGs, and Their Impact

Pushdown Automata use a stack for memory. They are closely related to Context-Free Grammars. Understanding PDAs unlocks key applications in computer science. Explore further learning resources to master PDAs.

Real-World Applications of Pushdown Automata

Compiler Design

Parsing context-free languages.

NLP

Syntax analysis in natural language processing.

Verification

Model checking of systems.

PDAs are used in programming language parsers and protocol analysis tools.

The Power Couple: PDA and Context-Free Grammars (CFG)

CFG to PDA

Convert a CFG to an equivalent PDA.

PDA to CFG

Convert a PDA to an equivalent CFG.

PDA and CFG are equivalent. For every CFG, there exists a PDA, and vice versa.

Acceptance Methods: Empty Stack vs. Final State

Empty Stack

PDA accepts if it empties its stack.

Final State

PDA accepts if it reaches a final state.

Both methods are equivalent. Any language accepted by one can be accepted by the other.

Instantaneous Description (ID): Tracing Computation

1. **State (q)**

   Current state of the PDA.

2. **Input (w)**

   Remaining input string.

3. **Stack (γ)**

   Current stack contents.

Represent PDA configuration as (q, w, γ). Use IDs to trace computation paths. For example: (q0, 0011, Z0) |- (q1, 011, 0Z0).

PDA in Action: Recognizing L = {0^n 1^n | n >= 0}

1. **Initial State**

   Read '0', push '0' onto stack.

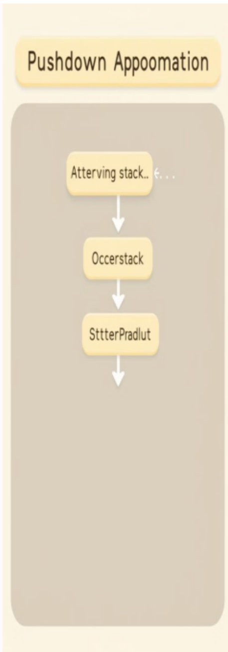
2. **Reading '1'**

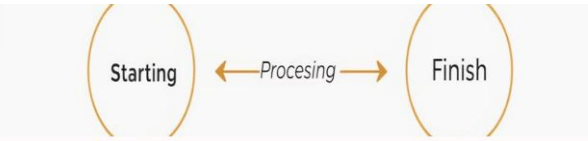
   Pop '0' from the stack.

3. **Accept**

   Reached when the stack is empty.

This example demonstrates PDA execution. The PDA accepts when the stack is empty upon reaching the end of the input.





Delving into the Transition Function (δ)

Current State (q)

The automaton's present state.

Input Symbol (a)

The current input being read.

Top of Stack (X)

Symbol at the top of the stack.

$\delta(q, a, X) = \{(p, Y), (r, Z), \ldots\}$ defines transitions. Epsilon transitions allow moves without consuming input. PDAs are inherently non-deterministic.

Mastering the Stack: LIFO Data Structure

1. **Push**
   Adding elements to the stack.
2. **Pop**
   Removing elements from the stack.

The stack uses LIFO (Last-In, First-Out). It is vital for storing and retrieving information during PDA execution.

Decoding the Formal Definition of a PDA

PDA Components

- **Q**: Finite set of states
- **Σ**: Input alphabet
- **Γ**: Stack alphabet

Key Functions

- **δ**: Transition function
- **q0**: Initial state
- **Z0**: Initial stack symbol
- **F**: Set of accepting states

Understanding each component is crucial. The transition function (δ) is particularly important for PDA operation.

Pushdown Automata

Pushdown Automata are a powerful computational model that extend the capabilities of Finite Automata by introducing a stack data structure. This additional memory allows PDAs to recognize a broader class of languages, including context-free languages, which cannot be recognized by Finite Automata alone.

In this deck, we will delve into the formal definition of a PDA, understanding the role of the stack and the transition function. We'll also explore how PDAs can be used to recognize specific languages, such as the language L = {0^n 1^n | n >= 0}, and trace the computation through the concept of Instantaneous Description (ID).