Formal Major Assignment

Formal Languages Problem: Dependent Structured Data Validation

Design a Pushdown Automaton (PDA) M that accepts strings belonging to a language L representing a valid **Dependent Structured Data** string.

The string is defined by the following sequence: **Header** $\rightarrow$ **Delimiter** $\rightarrow$ **Payload** $\rightarrow$ **Delimiter** $\rightarrow$ **Signature**.

The input format is:

$$\text{Header} \quad / \quad \text{Payload} \quad / \quad \text{Signature}$$

Language Component Definitions:

1. **Header (H):** A sequence of 'a's.
   - $L_H = \{a^n \mid n \geq 1\}$.
2. **Payload (P):** A sequence of 'b's whose length is the same as the Header.
   - $L_P = \{b^m \mid m = n\}$.
3. **Signature (S):** A sequence of 'c's whose length is the same as the Header.
   - $L_S = \{c^k \mid k = n\}$.

The complete language L is therefore:

$$L = \{a^n/b^n/c^n \mid n \geq 1\}$$

Crucial Constraint: Note that this language, $L = \{a^n b^n c^n \mid n \geq 1\}$ (when delimiters are ignored), is **context-sensitive** and generally **cannot be accepted by a standard Pushdown Automaton (PDA)**.

The Challenge is to design a PDA (NPDA or DPDA) that ACCEPTS a^n/b^n and then, through a *simplification* or *assumption*, attempts to handle the c^n component to fully accept the sequence.

Part 1: PDA Design (Formal Definition)

Design the PDA $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ that accepts the **longest possible prefix** of L while correctly verifying the dependency between the Header (a^n) and the Payload (b^n).

1. **Alphabet (Σ):** Clearly define the input alphabet.

2. **Stack Alphabet (Γ):** Clearly define the stack alphabet.
3. **States (Q):** Define the set of states, focusing on matching a^n and b^n.
4. **Transitions (δ):** Define the transition function.

Part 2: Implementation Analysis (Coding the PDA)

Provide a detailed analysis of the limitations of using a PDA for the full language $L = \{a^n/b^n/c^n \mid n \geq 1\}$.

1. **State Management:** Describe the role of the states for processing a^n and b^n.
2. **Limitation Explanation:** Explain *why* the standard PDA model fundamentally fails to fully verify the c^n component's dependency on a^n (or b^n) after the a^n/b^n check is complete.
3. **Maximum Accepted Language:** State the longest context-free prefix of L that the designed PDA can definitively accept using standard PDA mechanisms.

Resources

https://www.youtube.com/watch?v=TEQcJybMMFU&pp=ygUrbm9uIGRldGVybWluaXN0aWMgcHVzaGRvd24gYXV0b21hdGEgZXhhbXBsZQ%3D%3D