

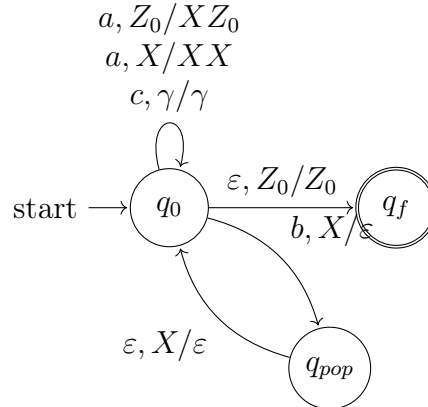
Solution - MC301 Theory of Computation Assignment-III

Problem 1: Construct PDAs

(a) $L = \{\omega : n_a(\omega) = 2n_b(\omega)\}$

Solution: We require a PDA that pushes one symbol for every a and pops two symbols for every b . To adhere to formal PDA restrictions (popping only one symbol at a time), we introduce an intermediate state q_{pop} .

$$\text{PDA} = (\{q_0, q_{pop}, q_f\}, \{a, b, c\}, \{X, Z_0\}, \delta, q_0, Z_0, \{q_f\})$$



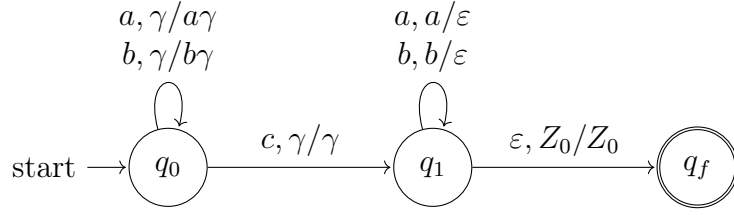
The transition function δ is defined as:

$$\begin{aligned} \delta(q_0, a, Z_0) &= \{(q_0, XZ_0)\} \\ \delta(q_0, a, X) &= \{(q_0, XX)\} \\ \delta(q_0, b, X) &= \{(q_{pop}, \varepsilon)\} \quad (\text{First pop}) \\ \delta(q_{pop}, \varepsilon, X) &= \{(q_0, \varepsilon)\} \quad (\text{Second pop}) \\ \delta(q_0, c, \gamma) &= \{(q_0, \gamma)\} \\ \delta(q_0, \varepsilon, Z_0) &= \{(q_f, Z_0)\} \end{aligned}$$

(b) $L = \{\omega c \omega^R : \omega \in \{a, b\}^*\}$

Solution: A standard palindrome recognizer with a center marker c .

$$\text{PDA} = (\{q_0, q_1, q_f\}, \{a, b, c\}, \{a, b, Z_0\}, \delta, q_0, Z_0, \{q_f\})$$

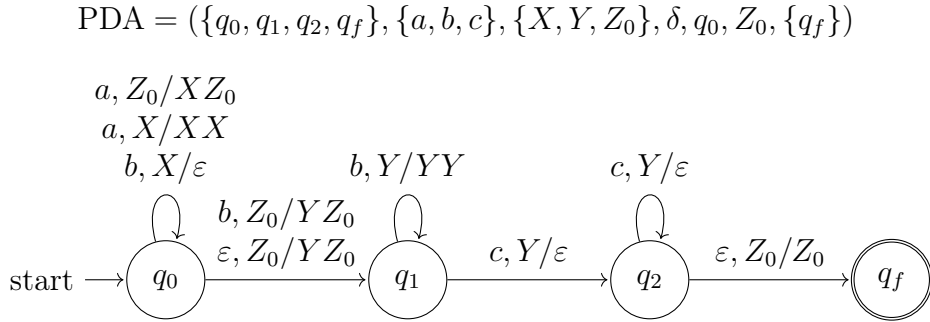


The transition function δ is defined as:

$$\begin{aligned}
 \delta(q_0, a, \gamma) &= \{(q_0, a\gamma)\} \\
 \delta(q_0, b, \gamma) &= \{(q_0, b\gamma)\} \\
 \delta(q_0, c, \gamma) &= \{(q_1, \gamma)\} \\
 \delta(q_1, a, a) &= \{(q_1, \varepsilon)\} \\
 \delta(q_1, b, b) &= \{(q_1, \varepsilon)\} \\
 \delta(q_1, \varepsilon, Z_0) &= \{(q_f, Z_0)\}
 \end{aligned}$$

(c) $L = \{a^n b^{n+m} c^m : n \geq 0, m \geq 1\}$

Solution: Phase 1 matches a^n with the first n b 's. Phase 2 counts the remaining m b 's to match with c^m .



The transition function δ is defined as:

$$\begin{aligned}
 \delta(q_0, a, Z_0) &= \{(q_0, XZ_0)\} \\
 \delta(q_0, a, X) &= \{(q_0, XX)\} \\
 \delta(q_0, b, X) &= \{(q_0, \varepsilon)\} \\
 \delta(q_0, b, Z_0) &= \{(q_1, YZ_0)\} \quad (\text{Start counting } m) \\
 \delta(q_0, \varepsilon, Z_0) &= \{(q_1, YZ_0)\} \quad (\text{Case } n = 0) \\
 \delta(q_1, b, Y) &= \{(q_1, YY)\} \\
 \delta(q_1, c, Y) &= \{(q_2, \varepsilon)\} \\
 \delta(q_2, c, Y) &= \{(q_2, \varepsilon)\} \\
 \delta(q_2, \varepsilon, Z_0) &= \{(q_f, Z_0)\}
 \end{aligned}$$

Problem 2: Instantaneous Description for $L = \{a^n b^n \mid n \geq 0\}$

Given PDA: $\text{PDA} = (\{q_0, q_1, q_f\}, \{a, b\}, \{X, Z_0\}, \delta, q_0, Z_0, \{q_f\})$

Trace for input string "abb":

1. $(q_0, abb, Z_0) \vdash (q_0, bb, XZ_0)$ (Read a , push X)
2. $(q_0, bb, XZ_0) \vdash (q_1, b, Z_0)$ (Read b , pop X , switch to q_1)
3. $(q_1, b, Z_0) \vdash \text{REJECT}$ (No transition for (q_1, b, Z_0))

Conclusion: The computation halts in a non-accepting state because the stack is empty of X 's, but there is still input b remaining. The string is rejected.

Problem 3: Final State vs Empty Stack Acceptance

Definitions

- **Acceptance by Final State:** The PDA accepts a string w if, after reading w , it enters a state $q_f \in F$. The stack content does not matter.

$$L(M) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q_f, \varepsilon, \gamma), q_f \in F, \gamma \in \Gamma^*\}$$

- **Acceptance by Empty Stack:** The PDA accepts a string w if, after reading w , the stack becomes completely empty. The final state does not matter.

$$N(M) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \vdash^* (q, \varepsilon, \varepsilon), q \in Q\}$$

Equivalence Proof

Theorem: The two modes of acceptance are equivalent.

Part 1: Final State \implies Empty Stack Given a PDA M_F that accepts by final state, we construct M_N that accepts by empty stack:

1. Create a new start state p_0 and push a new bottom marker X_0 .
2. Transition to the original start state q_0 : $\delta(p_0, \varepsilon, \varepsilon) = \{(q_0, X_0)\}$.
3. Simulate M_F . If M_F enters a final state, transition to a new state p_{erase} that pops everything.

Part 2: Empty Stack \implies Final State Given a PDA M_N that accepts by empty stack, we construct M_F :

1. Create a new start state p_0 and push a new bottom marker X_0 .
2. Transition to q_0 . Simulate M_N .
3. If the stack only contains X_0 (meaning M_N 's stack is effectively empty), transition to a new final state p_f .

Problem 4: PDA Processing

PDA A accepts $L = \{wcw^R\}$. Initial ID: $(q_0, aacaa, Z_0)$.

Trace for "aacaa":

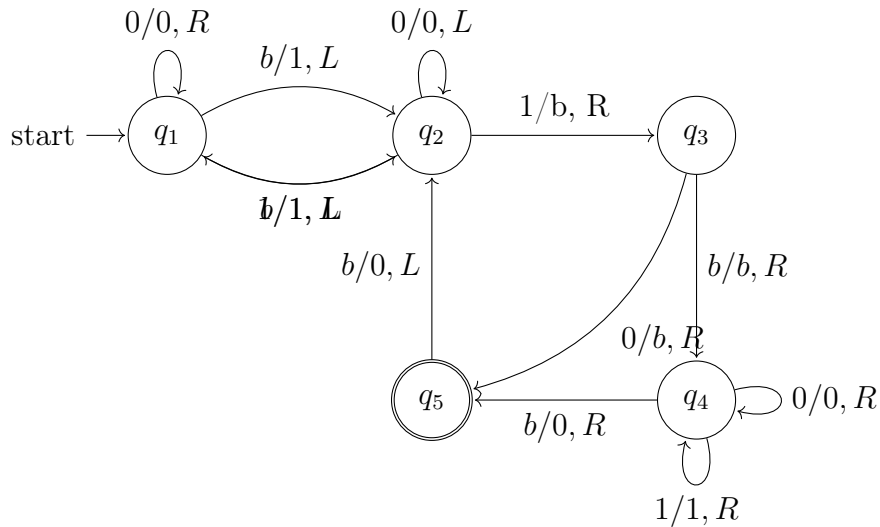
$$\begin{aligned} (q_0, aacaa, Z_0) &\vdash (q_0, acaa, aZ_0) \vdash (q_0, caa, aaZ_0) \vdash (q_1, aa, aaZ_0) \\ &\vdash (q_1, a, aZ_0) \vdash (q_1, \varepsilon, Z_0) \vdash (q_f, \varepsilon, Z_0) \quad (\text{Accept}) \end{aligned}$$

Processing of given strings:

- (i) **"abcba"**: Accepted. Final ID (q_f, ε, Z_0) .
- (ii) **"abcb"**: Rejected. Ends at (q_1, ε, aZ_0) . Stack not empty of input matches.
- (iii) **"acba"**: Rejected. Mismatch at (q_1, ba, aZ_0) .
- (iv) **"abac"**: Rejected. No transition for c in state q_1 .
- (v) **"abab"**: Rejected. Stuck in q_0 (no c found).

Problem 5: Turing Machine Transition Diagram

Based on the transition table provided:



Problem 6: CFG from PDA

The PDA given accepts $L = \{a^n b^n \mid n \geq 0\}$. **Context-Free Grammar G:**

$$S \rightarrow aSb \mid \varepsilon$$

Problem 7: Computation Sequences

Note: Notation bRq_2 means "Write b , Move Right, Go to State q_2 ".

1. Trace for "1213":

$q_1 \underline{1}213 \vdash bq_2 \underline{2}13$	$(q_1 : 1 \rightarrow b, R, q_2)$
$\vdash bbq_3 \underline{1}3$	$(q_2 : 2 \rightarrow b, R, q_3)$
$\vdash bq_6 bb \underline{3}$	$(q_3 : 1 \rightarrow 1, L, q_6 - \text{assuming table implied write 1})$
$\vdash bbq_1 \underline{1}3$	$(q_6 : b \rightarrow b, R, q_1)$
$\vdash bbbq_2 \underline{3}$	$(q_1 : 1 \rightarrow b, R, q_2)$
$\vdash \mathbf{HALT}$	(No transition for $(q_2, 3)$)

2. Trace for "2133":

$q_1 \underline{2}133 \vdash \mathbf{HALT}$ (No transition for $(q_1, 2)$)

3. Trace for "312":

$q_1 \underline{3}12 \vdash \mathbf{HALT}$ (No transition for $(q_1, 3)$)

Problem 8: Halting vs Rejecting States

- **Halting State:** Any state where the TM stops because no transition is defined.
- **Rejecting State:** A halting state that is NOT in the set of accepting states F .

Problem 9: Stack's Role in $a^n b^n$

The stack acts as infinite memory for counting:

1. **Push:** Every a adds a token to the stack (counting n).
2. **Pop:** Every b removes a token (matching n).
3. **Verify:** Acceptance requires the stack to be empty exactly when input is exhausted.

Problem 10: Input Head Movement

Feature	PDA	Turing Machine
Direction	Unidirectional (Left \rightarrow Right)	Bidirectional (Left or Right)
Access	Read-once	Random Access (can revisit)
Writing	Read-only input	Read/Write tape