

# **Theory of Computation: Complete Exam Notes**

Definitions, Theorems, & Solved Problems

## **Contents**

## 1 1. The Standard Turing Machine (TM)

### 1.1 Formal Definition

A Turing Machine is defined as a 7-tuple:

$$M = (Q, \Sigma, \Gamma, \delta, q_0, \square, F)$$

Where:

- $Q$ : Finite set of internal states.
- $\Sigma$ : Input alphabet ( $\Sigma \subseteq \Gamma - \{\square\}$ ).
- $\Gamma$ : Finite tape alphabet (includes input symbols  $\Sigma$  and the Blank  $\square$ ).
- $\delta$ : Transition function.
- $q_0$ : Initial state ( $q_0 \in Q$ ).
- $\square$ : Special Blank symbol (initially fills the infinite tape beyond input).
- $F$ : Set of final states ( $F \subseteq Q$ ).

### 1.2 The Transition Function

The transition function is defined as:

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$$

**Interpretation:** If  $\delta(q, a) = (p, b, R)$ , the machine:

1. Reads symbol **a** while in state **q**.
2. Overwrites it with symbol **b**.
3. Changes state to **p**.
4. Moves the head one cell to the **Right** (*R*).

### 1.3 Standard Notations

- **Instantaneous Description (ID):** A snapshot of the machine is denoted  $x_1qx_2$ . The tape contains  $x_1x_2$ , and the head is scanning the first symbol of  $x_2$ .
- **Move Symbol:** A move from one configuration to another is denoted by  $\vdash$ . A sequence of moves is denoted by  $\vdash^*$ .
- **Never Halt (Infinite Loop):** If the machine enters an infinite loop from a configuration, it is denoted as:

$$x_1qx_2 \vdash \infty$$

## 2 2. The Chomsky Hierarchy

1. **Type 0 (Recursively Enumerable - RE):**
  - Recognized by Turing Machines.
  - **Behavior:** If  $w \in L$ , TM accepts. If  $w \notin L$ , TM may reject OR loop forever.
2. **Recursive Languages (Decidable):**
  - A subset of Type 0. Recognized by TMs that **always halt**.
3. **Type 1 (Context-Sensitive):** Recognized by Linear Bounded Automata (LBA). (Tape length is bounded by input length).
4. **Type 2 (Context-Free):** Recognized by Pushdown Automata (PDA).
5. **Type 3 (Regular):** Recognized by Finite Automata (FA).

## 3 3. Important Theorems

### 3.1 Rice's Theorem (Undecidability)

**Statement:** Any **non-trivial property** of the Recursively Enumerable (RE) languages is **undecidable**.

**Explanation:**

- A "property" is a set of languages.
- "Non-trivial" means the property is true for *some* RE languages but not *all*.
- **Consequence:** You cannot write an algorithm that inspects a TM and decides if its language is empty, finite, infinite, or regular.

### 3.2 Church-Turing Thesis

**Statement:** Any computation that can be performed by mechanical means (algorithm) can be performed by a Turing Machine.

This thesis establishes the Turing Machine as the standard mathematical model for "algorithm".

## 4 4. Combining Turing Machines

To solve complex tasks, we combine simple TMs:

1. **Transducers:** Viewing TMs as computers of functions  $f(w)$  rather than just accepters.
2. **Block Diagrams:** Visualizing complex machines as connected blocks (e.g., "Copy Machine" → "Shift Machine").
3. **Subroutines:** Using a portion of the tape as a workspace to run a sub-algorithm, then returning control to the main state.

## 5 5. Solved Examples (Basic to Advanced)

### 5.1 1. Language $L = \{a^n b^n : n \geq 1\}$

*Goal: Check if count of a's equals count of b's.*

- Replace leftmost  $a$  with  $X$ .
- Move Right to find the first  $b$ . Replace it with  $Y$ .
- Move Left to find the first  $X$ . Move one step Right to find the next  $a$ .
- Repeat. If all symbols are matched, Accept.

### 5.2 2. Language $L = \{a^n b^n c^n\}$

*Goal: Match three distinct symbols (Not Context-Free).*

- Mark  $a \rightarrow X$ .
- Move Right scanning past  $a$ 's and  $Y$ 's to find  $b$ . Mark  $b \rightarrow Y$ .
- Move Right scanning past  $b$ 's and  $Z$ 's to find  $c$ . Mark  $c \rightarrow Z$ .
- Move Left scanning past everything to find  $X$ .
- Repeat. If successful, Accept.

### 5.3 3. Unary Addition ( $f(x, y) = x + y$ )

*Input:  $1^m 0 1^n$  (e.g., 111011).*

- The goal is to merge the two blocks of 1s.
- Replace the '0' separator with '1'.
- Go to the end of the tape and erase the last '1' (to maintain the correct count).
- Result:  $1^{m+n}$ .

### 5.4 4. Unary Multiplication ( $f(x, y) = x \cdot y$ )

*Input:  $1^m 0 1^n$ .*

- **Logic:** Repeated addition. For every '1' in block  $m$ , copy block  $n$  to the output.
- Step 1: Change first '1' of  $m$  to  $X$ .
- Step 2: Copy block  $n$  to the end of the tape.
- Step 3: Return to  $X$ , find next '1' in  $m$ , change to  $X$ .
- Step 4: Repeat copy.
- Step 5: Erase original inputs  $m$  and  $n$ , leaving only result.

**5.5 5. String Reversal ( $f(w) = w^R$ )**

- Mark the first symbol (e.g.,  $a \rightarrow A$ ).
- Move to the first blank at the far Right. Write  $a$ .
- Move Left back to the marked symbol. Move one step Right to next symbol.
- Repeat until all symbols are copied in reverse.
- Erase original string markers.

**5.6 6. Language  $L = \{ww\}$  (Doubled String)**

*Input:*  $w \in \{a,b\}^+$ . *Goal:* Check if first half equals second half.

1. **Find Midpoint:** Mark Left ( $X$ ) and Right ( $Y$ ) ends iteratively moving inward. If markers meet, length is even. If not, Reject.
2. **Reset:** Change  $X, Y$  back to original symbols (or use temporary marks).
3. **Compare:** Mark first symbol of first half. Move to first symbol of second half (after midpoint). Check match.
4. Repeat for all symbols.

**5.7 7. Middle of Even String**

*Input:*  $w = a_1 \dots a_{2n}$ . *Output:*  $a_1 \dots a_n c a_{n+1} \dots a_{2n}$ .

1. Mark first symbol (Left) and last symbol (Right).
2. Move Right from Left marker to mark next symbol.
3. Move Left from Right marker to mark next symbol.
4. Repeat until Left head and Right head are on adjacent cells.
5. Shift the right half of the string one cell to the Right.
6. Write  $c$  in the newly created gap.

**5.8 8. Modulo 5 ( $x \bmod 5$ )**

*Input:* Unary string of 1s.

- Use cyclic states  $q_0, q_1, q_2, q_3, q_4$ .
- Start in  $q_0$ . For every '1' read, move to next state ( $q_0 \rightarrow q_1 \rightarrow q_2 \rightarrow q_3 \rightarrow q_4 \rightarrow q_0$ ).
- When Blank is found, the current state corresponds to the remainder.
- Write that number of 1s on the tape as output.

**5.9 9. Decimal Addition ( $x + y$ )**

*Input:* 15#7.

- Simulate manual addition starting from the rightmost digits (Least Significant Bit).
- Read last digit of  $y$ , replace with marker.
- Move Left to find last digit of  $x$ . Add them.
- If Sum  $> 9$ , write (Sum % 10) and switch to a "Carry" state.
- If Sum  $\leq 9$ , write Sum and remain in "No Carry" state.
- Move Right to process next digit of  $y$ .