3 Turing Machines and Time Complexity

Material for this chapter is drawn from [6, Chapter 14.3 and recap Chapters 8.1, 8.2].

3.1 Definition (Turing Machine)
A standard (*single tape, deterministic*) *Turing machine* (TM) is a quintuple $M = (Q, \Sigma, \Gamma, \delta, q_0)$ where:

- Q a finite set of *states*;
- Γ a finite set called *tape alphabet* containing a *blank* B;
- $\Sigma \subseteq \Gamma \setminus \{B\}$ the *input alphabet*;
- $\delta : Q \times \Gamma \overset{\text{partial}}{\to} Q \times \Gamma \times \{L, R\}$, the *transition function*;
- $q_0 \in Q$ the *start state*.

3.2 Remark (Turing Machine Properties)
The following hold for every Turing Machine:

- The tape has a left boundary and is infinite to the right.
- The tape has positions numbered starting with 0.
- Each tape position contains an element from Γ.
- The Turing Machine starts in state q_0 and at position 0.
- The input is written on the tape beginning at 1.
- The rest of tape is blank.
- A transition consists of exactly three steps that happen simultaneously.
    1. Change to the appropriate next state based on the symbol written on the tape and the transition function.
    2. Write a new symbol at tape position. Note: the symbol does not need to actually change (i.e., a/a).
    3. Move the head left or right exactly one position determined by the transition function.
- Computation halts if no transition is defined.
- Computation terminates abnormally if it moves left of position 0. (Recall that there is a left boundary).
- TMs can be represented by state diagrams.

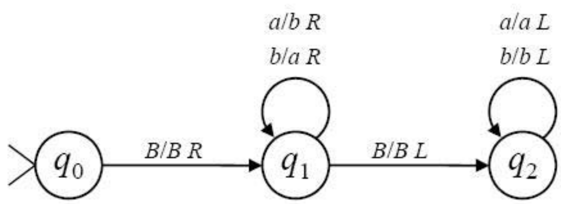
3.3 Example (Transition in Natural Language)
A transition x/yD where $x, y \in \Gamma$ and $D \in \{L, R\}$ is read as "if the symbol on the tape is x, write y in that position, and move the tape head to the direction D."

3.4 Example (State & Tape as a String)
The current state and tape of a Turing Machine can be specified as a string. The first symbol $\vdash$ simply indicates that the following string is such an encoding. We use the current state to indcate where the *head* of the Turing Machine is. That is, the symbol following the state marker q_i is currently being read by the head.

$$\vdash q_0 BababB \tag{1}$$
$$\vdash Bq_1 ababB \tag{2}$$
$$\vdash Bbaq_2 baB \tag{3}$$

1. The Turing Machine is in state q_0 and is reading the symbol B.
2. The Turing Machine is in state q_1 and is reading the symbol a.
3. The Turing Machine is in state q_2 and is reading the symbold b.

3.5 Example (Switching Example)
Swap all a's to b's and all b's to a's in a string of a's and b's.



Computation example:

$$\begin{array}{llll} \vdash q_0BababB & \vdash Bbaq_1abB & \vdash Bbabq_2aB & \vdash Bq_2babaB \\ \vdash Bq_1ababB & \vdash Bbabq_1bB & \vdash Bbaq_2baB & \vdash q_2BbabaB \\ \vdash Bbq_1babB & \vdash Bbabaq_1B & \vdash Bbq_2abaB & \end{array}$$

Number of steps required with input string size n: $1 + n + 1 + n = 2n + 2$

3.6 Example (Copy-String Example)
Copying a string: BuB becomes $BuBuB$ (u is a string of a's and b's)
1. State Diagram (Figure 1.)
2. Complexity: $O(n^2)$, where n is length of input string.

3.7 Definition (Language)
A language is a set of strings that adhere to some criteria. This may be specified for a language L,

$$L = \{x_j^i \mid x_j \in \Sigma, i, j \in \mathbb{N}\}$$

where i is the number of consecutive instances of x.

3.8 Notation (Star-notation)
The * may be used to indicate a value $i \geq 0 \in \mathbb{N}$.

3.9 Notation (Star-notation)
A language $L = A^*B^3$. Thus, *"BBB"* $\in L$ and *"AAAABBB"* $\in L$ and so on.

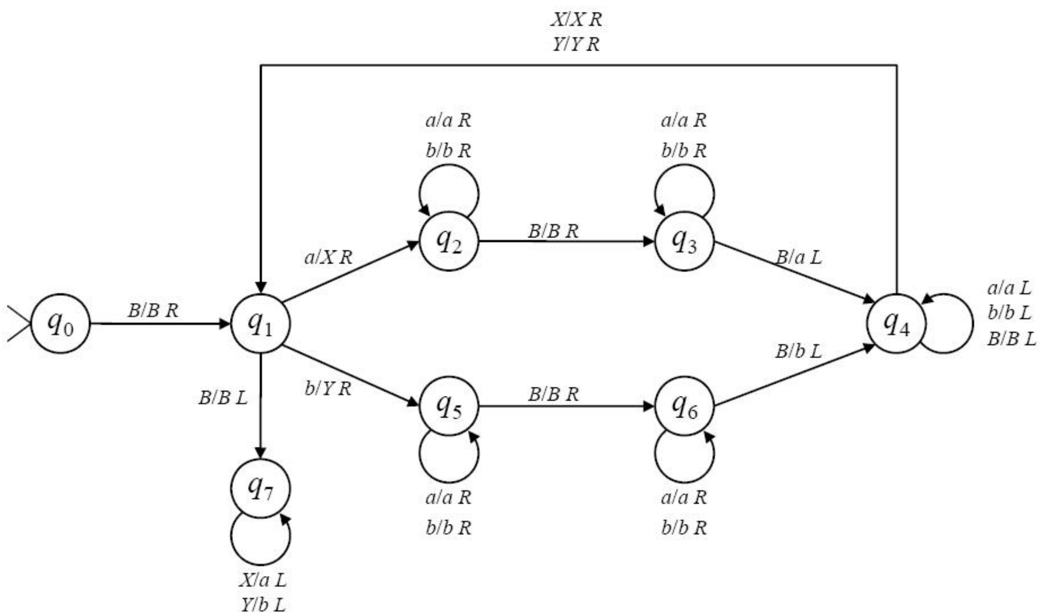


Figure 1: Turing Machine for Example 3.6.

3.10 Notation (Or-notation)
When specifying a language, we can use the $\cup$ symbol, borrowed from set theory, to indicate "or." That is, when specifying a string, a certain symbol maybe $x \cup y$ for $x, y \in \Sigma$. For example, a language of strings that begins and ends with an "a" symbol, but may have any numbers of "a" or "b" in between, specified as:

$$L = \{a(a \cup b)^* a\}$$

3.11 Definition (Language Accepting TM)
We say that a TM is *language accepting* if computation halts *normally* in a *final* state. A language accepting TM is defined as a sextuple $M_{LA} = (Q, \Sigma, \Gamma, \delta, q_0, F)$, where $F \subseteq Q$ of *final* states.

3.12 Example (Or-notation Example)
A TM for $(a \cup b)^* aa(a \cup b)^*$. Let $\Sigma = \{a, b\}$.
1. State Diagram: Figure 2.
2. Complexity: $O(n)$, where n is the length of the input string.

3.13 Example (Set-notation Example)
A TM for $\{a^i b^i c^i \mid i \geq 0\}$.
1. State Diagram: Figure 3.
2. $\Sigma = \{a, b, c\}$
3. Number of steps required with input string size $n = 3i$: $O(i \cdot 4i) = O(i^2) = O(n^2)$ (worst case)

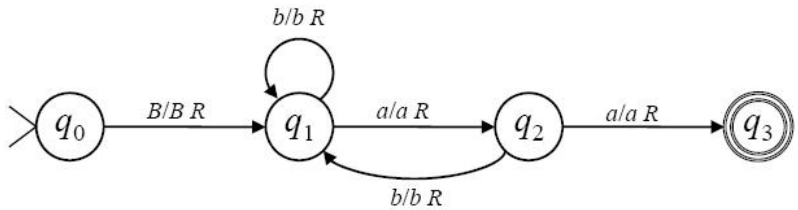


Figure 2: Turing Machine for Example 3.12.

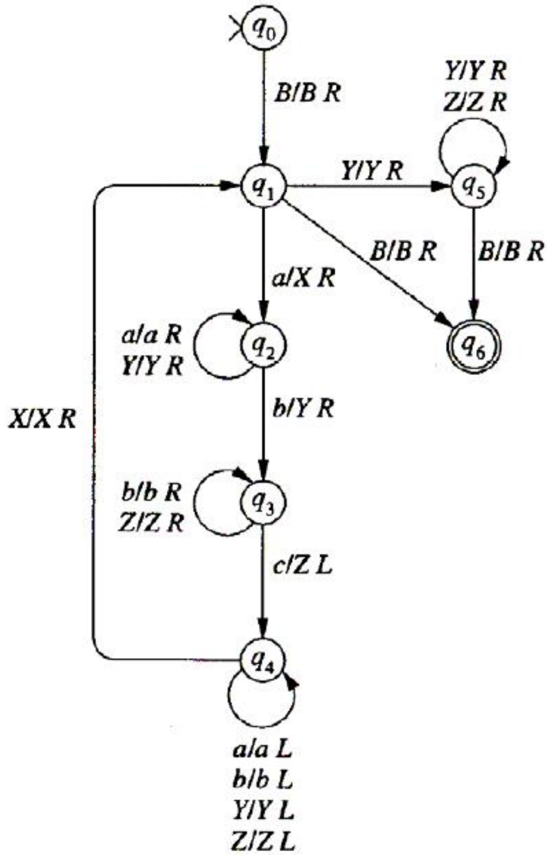


Figure 3: A TM for $\{a^i b^i c^i \mid i \geq 0\}$

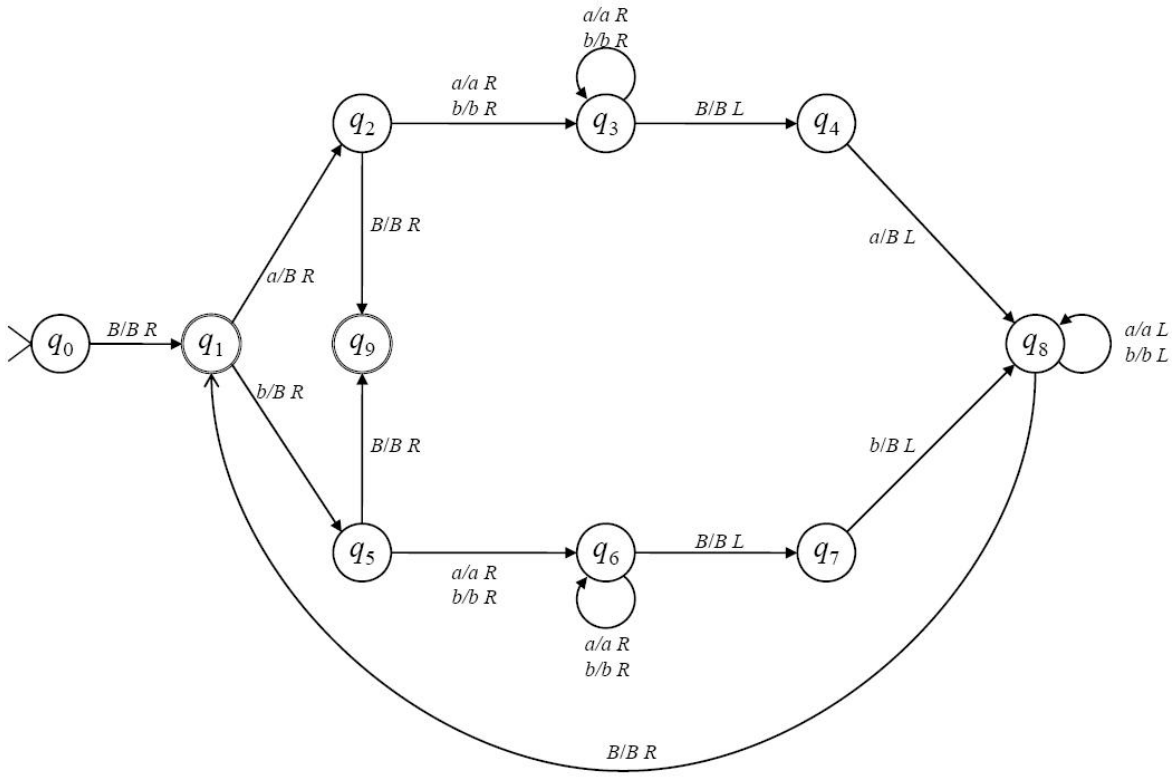


Figure 4: A TM for palindromes over a and b.

3.14 Example (Palindrome Example)
A TM for palindromes over a and b (Figure 4).
Number of steps required with input string size n:

$$1 + \sum_{i=1}^{n} i = 1 + \frac{1}{2} \cdot (n+1) \cdot n \in O(n^2) \qquad \text{(worst case)}$$

3.15 Definition (Time Complexity)
For any TM M, the *time complexity* of M is the function $tc_M : \mathbb{N} \to \mathbb{N}$ s.t. $tc_M(n)$ is the maximum number of transitions processed by a computation of M on input of length n.

3.16 Definition (Acceptance)
A language L is *accepted in deterministic time* $f(n)$ if there is a single tape deterministic TM M for it with $tc_M(n) \in O(f(n))$.

3.17 Remark (Worst-case Behavior)
Note, that worst-case behavior can happen when a string is *not* accepted. For example, it is (frequently) straightforward TM to accept strings containing an a.