Chapter 8

Turing Machine (TMs)
Turing Machines (TMs)

- Accepts the languages that can be generated by *unrestricted* *(phrase-structured)* grammars

- No computational machine (i.e., computational language recognition system) is *more powerful* than the class of TMs due to the language processing power, i.e., the *generative power of grammars*, its unlimited memory, and time of computations

- Proposed by Alan Turing in 1936 as a result of *studying algorithmic processes* by means of a *computational model*

- TMs are similar to FSAs since they both consist of
  1. a *control mechanism*, and
  2. an *input tape*

In addition, TMs can
  1. *move* their tape head *back and forth*, and
  2. *write* on, as well as *read* from, their tapes.
Turing Machines

- Defn. 8.1.1 A TM is a quintuple \( M = (Q, \Sigma, \Gamma, \delta, q_0) \), where
  - \( Q \) is a finite set of states
  - \( \Gamma \) is a finite set called the tape alphabet which contains \( B \), a special symbol that represents a blank
  - \( \Sigma \subseteq \Gamma - \{ B \} \), is the input alphabet
  - \( \delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{ L, R \} \), a transition function, which is a partial function
  - \( q_0 \in Q \), is the start state
Turing Machines

Machine Operations:

- **Write** operation - replaces a symbol on the tape with another (not necessarily distant) symbol

- **Move** operation - moves the tape head one cell to the right (left, respectively) and then *shift* to a new (or current) state

- **Halt** operation - halts when the TM encounters a \(<\text{state, input symbol}>\) pair for which no transition is defined
Turing Machines

- **Machine Operations.** TMs are designed to perform *computations* on strings from the input alphabet.

- **Example 8.1.2.** A TM produces a *copy* of input string over \{ \( a, b \) \} with input \( BuB \) and terminates with tape \( BuBuB \), where \( u \in (a \cup b)^* \).
Transitions of TMs:

- \[ uq_i vB \xrightarrow{m} xq_j yB \] denotes that \( xq_j yB \) is obtained from \( uq_i vB \) by a single transition of \( M \).
- \[ uq_i vB \xrightarrow{* m} xq_j yB \] denotes that \( xq_j yB \) is obtained from \( uq_i vB \) by zero or more transitions of \( M \).

TM as Language Acceptors

- TMs can be designed to accept languages besides computing functions, and accepting a string does not require the entire input string to be read.
- Defn. 8.2.1 Let \( M = (Q, \Sigma, \Gamma, \delta, q_0, F) \) be a TM. A string \( u \in \Sigma^* \) is accepted by final state if the computation of \( M \) with input \( u \) halts in a final state. A computation that terminates abnormally rejects the input regardless of the state in which the machine halts. The language of \( M \), \( L(M) \), is the set of all string accepted by \( M \). A language accepted by a TM is called a recursively enumerable language.
Example 8.2.2. A TM that accepts the language \( \{ a^i b^i c^i | i \geq 0 \} \) is
An input string $S$ is accepted by TM $M$ if the computation with $S$ causes $M$ to halt. $M$ rejects $S$ when $M$ terminates abnormally or $M$ never halts with $S$.

A TM of which its acceptance is defined by halting (normally) is defined by the quintuple $(Q, \Sigma, \Gamma, \delta, q_0)$.

**Theorem 8.3.2** The following statements are equivalent:

i) The language $L$ is accepted by a TM that accepts by final state

ii) The language $L$ is accepted by a TM that accepts by halting

**Example 8.3.1** A TM that accepts $(a \cup b)^* aa(a \cup b)^*$ by halting.
Transitions of TMs

- Design a TM that computes the \textit{proper-subtraction function}, i.e., \( m \div n = \max(m - n, 0) \) such that \( m \div n \) is \( m - n \), if \( m > n \), and 0, if \( m \leq n \). The TM will start with a tape consisting \( 0^{m}10^{n} \) surrounded by blanks, i.e., \( B0^{m}10^{n}B \). The machine halts with its result on its tape, surrounded by blanks.
8.7 Nondeterministic TMs (NTMs)

- Provide *more than one applicable transition* for some current state/input symbol pair, i.e., > 1 non-deterministic choice

- **Formal Definition:** A NTM \( M = (Q, \Sigma, \Gamma, \delta, q_0) \) or \( M = (Q, \Sigma, \Gamma, \delta, q_0, F) \), where \( Q, \Sigma, \Gamma, \delta, q_0, \) and \( F \) are as defined in any DTMs and \( \delta: Q \times \Gamma \to 2^{Q \times \Gamma \times \{L, R\}} \)

- **Example 8.7.1** A NTM that accepts strings containing a ‘c’, which is either *preceded* or *followed* by ‘ab’
8.7 Nondeterministic TMs (NTMs)

- Acceptance in NTMs can be defined by *final state* or by *halting* alone, similar to the DTMs.
- Every NTM can be transformed into an equivalent DTM that accepts by *halting*, which is chosen because it reduces the number of computations from 3 to 2.
- The language accepted by NTMs are precisely those accepted by DTMs.
  - The converted can be done by using multiple (3-) tapes.
  - *Multiple* computations for a single input string are sequentially generated and examined.
  - A *maximum* number of transitions can be defined for any combination of <state, input symbol>.
8.7 Nondeterministic TMs (NTMs)

- **Example.** Transforming the NTM in Example 8.7.1 into its DTM

```
q_0 -> q_1 -> q_2 -> q_3 -> q_4
```

### Table 8.1: Ordering of Transitions

<table>
<thead>
<tr>
<th>State</th>
<th>Symbol</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_0</td>
<td>B</td>
<td>1q_1, B, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_1, B, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_1, B, R</td>
</tr>
<tr>
<td>q_1</td>
<td>a</td>
<td>1q_1, a, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_1, a, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_1, a, R</td>
</tr>
<tr>
<td>q_1</td>
<td>b</td>
<td>1q_1, b, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_1, b, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_1, b, R</td>
</tr>
<tr>
<td>q_1</td>
<td>c</td>
<td>1q_1, c, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_2, c, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_5, c, L</td>
</tr>
</tbody>
</table>

```
<table>
<thead>
<tr>
<th>State</th>
<th>Symbol</th>
<th>Transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>q_2</td>
<td>a</td>
<td>1q_3, a, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_3, a, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_3, a, R</td>
</tr>
<tr>
<td>q_3</td>
<td>b</td>
<td>1q_4, b, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_4, b, R</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_4, b, R</td>
</tr>
<tr>
<td>q_5</td>
<td>b</td>
<td>1q_6, b, L</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_6, b, L</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_6, b, L</td>
</tr>
<tr>
<td>q_6</td>
<td>a</td>
<td>1q_7, a, L</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2q_7, a, L</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3q_7, a, L</td>
</tr>
</tbody>
</table>
```

Examples of transitions:

- $q_0 B a c a b B 1$
- $q_0 B a c a b B 1$
- $B q_1 a c a b B 1$
- $q_1 a c a b B 1$
- $B a q_1 a c a b B 1$
- $B a q_1 a c a b B 2$
- $B a c q_1 a b B 1$
- $B a c q_1 a b B 1$
- $B a c a q_1 b B 1$
- $B a c a q_3 b B 1$
- $B a c a b q_1 B$
- $B a c a b q_4 B$
Transitions of TMs

- Design a TM that takes as input a number \( N \) and adds 1 to it in *binary*. The tape initially contains a $ followed by \( N \) in *binary*. The tape head is initially scanning the $ in the initial state \( q_0 \). The TM should halt with \( N+1 \), in *binary*, on the tape, scanning the leftmost symbol of \( N+1 \) in the final state \( q_f \) with $ removed. For instance, \( q_0$10011 \) yields \( q_f10100 \), and \( q_0$11111 \) yields \( q_f100000 \).
8.6 Multitape TMs

- A $K$-tape TM
  - consists of $k$ tapes and $k$ independent tape heads
  - reads $k$ tapes simultaneously, but has only one state
  - is configured by the current tape symbol being pointed to by each tape head and the current state, e.g.,

```
Tape 3

Tape 2

Tape 1
```

- A transition in a multitape TM may
  - change the current state,
  - (over)write a symbol on each tape, and
  - independently reposition each of the tape heads
8.6 Multitape TMs

- A transition of a $k$-tape TM is defined as
  \[ \delta(q_i, x_1, x_2, \ldots, x_k) = [q_j; y_1, d_1; y_2, d_2; \ldots; y_k, d_k] \]
  where $q_i, q_j \in Q$, $x_n, y_n \in \Gamma$, and $d_n \in \{L, R, S\}$, $1 \leq n \leq k$.

- Initialize configuration:
  - The input string is placed on tape 1, whereas all the other tapes are assumed to be blank to begin with.
  - The tape heads scan the leftmost position of each tape.
  - Any tape head attempts to move to the left of the leftmost position terminates the computation abnormally.

- A language accepted by a TM is a recursively enumerable language.

- A language that is accepted by a TM that halts for all input strings is said to be recursive.
Example 8.6.2 The set \( \{ a^k \mid k \text{ is a perfect square} \} \) is a recursively enumerable language (and is also a recursive language).

- Tape 1 holds the input string, a string of \( a \)'s
- Tape 2 holds a string of \( X \)'s whose length is a perfect square
- Tape 3 holds a string of \( X \)'s whose length is \( \sqrt{|S|} \), where \( S \) is the string on Tape 2

- Step 1: Since the input is not a null string, initialize tapes 2 and 3 with an \( X \), and all the tape head move to Position 1

- Step 2: Move the heads of tapes 1 and 2 to the right, since they have scanned a nonblank square

Accept: if both read a blank
Reject: if tape head 1 reads a blank and tape head 2 reads an \( X \)
Example 8.6.2 (Continued).

(iii) Tape 3 - 2

\[ X \quad X \]

Tape 2 – \(2^2\)

\[ X \quad X \quad X \quad X \]

Tape 1 - input

\[ a \quad a \quad a \quad a \quad a \]

• Step 3: Reconfiguration for comparison with the next perfect square by
  - adding an \(X\) on tape 2 to yield \(k^2+1\) \(X\)'s
  - appending two copies of the string on tape 3 to the end of the string on tape 2 to yield \((k+1)^2\) \(X\)'s
  - adding an \(X\) on tape 3 to yield \((k + 1)\) \(X\)'s on tape 3
  - moving all the tape heads to Position 1

(iv) Tape 3 - 2

\[ X \quad X \]

Tape 2 – \(2^2\)

\[ X \quad X \quad X \quad X \]

Tape 1 - input

\[ a \quad a \quad a \quad a \quad a \]

\(q_2\)

• Step 4: Repeat Steps 2 through 3.

(v) Tape 3 - 3

\[ X \quad X \quad X \]

Tape 2 – \(3^2\)

\[ X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \quad X \]

Tape 1 - input

\[ a \quad a \quad a \quad a \quad a \]

\(q_2\)

• Another iteration of Step 2 halts and rejects the input.
Example 8.6.2 (Continued). The transition function of the TM that accepts \( \{ a^k | k \text{ is a perfect square} \} \):

\[
\begin{align*}
q_0 \quad [B/B R, B/B R, B/B R] & \quad q_1 \quad [a/a S, B/X S, B/X S] & \quad q_2 \quad [a/a R, X/X R, X/X S] \\
[\text{oward } B] & \quad [B/B S, B/B S, X/X S] \\
q_3 & \quad q_4 \quad [a/a S, B/X R, X/X S] & \quad q_5 \quad [a/a S, B/X R, X/X S] \\
[\text{oward } B] & \quad [a/a S, B/X R, X/X R] \\
q_6 & \quad q_2 \quad [a/a L, B/B L, B/X L] & \quad q_6 \quad [a/a L, X/X L, X/X L], \\
[\text{oward } B] & \quad [a/a L, X/X L, B/B S], \\
& \quad [a/a L, B/B S, B/B S], \\
& \quad [B/B S, X/X L, B/B S]
\end{align*}
\]
Example 8.6.2 (Continued). The transition function of the TM that accepts \( \{ a^k \mid k \text{ is a perfect square} \}:

[Step 1]
\[
\begin{align*}
\delta(q_0, B, B, B) &= [q_1; B, R; B, R; B, R] \quad \text{(initialize the tape)} \\
\delta(q_1, a, B, B) &= [q_2; a, S; X, S; X, S] \quad \text{(}q_1\text{ is a final state)}
\end{align*}
\]

[Step 2]
\[
\begin{align*}
\delta(q_2, a, X, X) &= [q_2; a, R; X, R; X, S] \quad \text{(compare strings on tapes 1 and 2)} \\
\delta(q_2, B, B, X) &= [q_3; B, S; B, S; X, S] \quad \text{(accept; } q_3\text{ is a final state)} \\
\delta(q_2, a, B, X) &= [q_4; a, S; X, R; X, S] \quad \text{(add an } X\text{ to tape 2 and re-compute)}
\end{align*}
\]

[Step 3]
\[
\begin{align*}
\delta(q_4, a, B, X) &= [q_5; a, S; X, R; X, S] \quad \text{(rewrite tapes 2 and 3)} \\
\delta(q_5, a, B, X) &= [q_4; a, S; X, R; X, R] \quad \text{(add two } X\text{'s to tape 2 for each } X\text{ on tape 3 – to generate } (k+1)^2) \\
\delta(q_4, a, B, B) &= [q_6; a, L; B, L; X, L] \quad \text{(add an } X\text{'s to tape 2 to yield } k+1)
\end{align*}
\]

[Step 4]
\[
\begin{align*}
\delta(q_6, a, X, X) &= [q_6; a, L; X, L; X, L] \quad \text{(reposision tape heads)} \\
\delta(q_6, a, X, B) &= [q_6; a, L; X, L; B, S] \quad \text{(tape 3 at 1\textsuperscript{st} cell, but not tapes 1 & 2)} \\
\delta(q_6, a, B, B) &= [q_6; a, L; B, S; B, S] \quad \text{(tape 2 & 3 at 1\textsuperscript{st} cell, but not tape 1)} \\
\delta(q_6, B, X, B) &= [q_6; B, S; X, L; B, S] \quad \text{(tape 1 & 3 at 1\textsuperscript{st} cell, but not tape 2)} \\
\delta(q_6, B, B, B) &= [q_2; B, R; B, R; B, R] \quad \text{(repeat comparison cycle)}
\end{align*}
\]
8.6 Multitape TMs

- A multitape TM can be represented by a state transition diagram.
- **Example 8.6.3** A 2-tape TM that accepts \{ uu \mid u \in \{ a, b \}^* \}.

Computation:
1) Make a *copy* of the input \( S \) (on tape 1) to tape 2; tape heads: right of \( S \).
2) Move both tape heads one step to the left.
3) Move the head of tape 1 *two* squares for each square move of tape 2.
4) *Reject* the input \( S \) if the TM halts in \( q_3 \). (i.e. \(|S|\) is *odd*.)
5) *Compare* the 1st half with the 2nd half of \( S \) in \( q_4 \)
6) *Accept* \( S \) in \( q_5 \)

\[
\begin{array}{c}
q_0 \\
\text{[B/B R, B/B R]} \quad \Rightarrow \quad q_1 \\
\text{[x/x R, B/x R]} \quad \Rightarrow \quad q_2 \\
\text{[B/B L, B/B L]} \quad \Rightarrow \quad q_3 \\
\text{[x/x L, y/y L]} \quad \Rightarrow \quad q_4 \\
\text{[x/x L, y/y S]} \quad \Rightarrow \quad q_5 \\
\text{[y/y R, B/B R]} \quad \Rightarrow \quad q_5 \\
\end{array}
\]
Theorem 8.6.1 A language $L$ is accepted by a multitape TM iff it is accepted by a standard TM.

*Proof.* By simulating a multitape TM using a single tape with multitacksTM.