Part II: computability
We would like to study problems that can and cannot be solved by computers
We need more powerful model
Finite automata: small memory (states)
PDA: unlimited memory (stack) by push/pop
Turing machine: unlimited and unrestricted memory
This is about everything a real computer can do
Thus problems not solved by Turing machine
⇒ beyond the limit of computation
Turing Machines: II

- A TM has a tape as the memory

- Differences from finite automata
  - write/read tape
  - head moves left/right
  - infinite space in the tape
  - rejecting/accepting take immediate effect
  - machine goes on forever, otherwise
Example

$$B = \{ w \# w \mid w \in \{0, 1\}^* \}$$

This language is known to be not a CFL (example 2.22; details not discussed)

Running a sample input. Figure 3.2

⊔: blank symbol

We assume infinite ⊓’s after the input sequence

Strategy: zig-zag to the corresponding places on the two sides of the # and determine whether they match.
Turing Machines: IV

Algorithm:
1. scan to check \(\#\)
2. check \(w\) and \(w\)
Formal definition of TM I

- It’s complicated and seldom used
- \( \delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\} \)
- Example:
  \[ \delta(q, a) = (r, b, L) \]

- \( q \): current state
- \( a \): pointed in tape
- \( r \): next state
- \( b \): replace \( a \) with \( b \)
- \( (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}}) \)
Formal definition of TM II

- $Q$: states
- $\Sigma$: input alphabet (blank: $\sqcup \notin \Sigma$)
- $\Gamma$: tape alphabet, $\sqcup \in \Gamma$, $\Sigma \subset \Gamma$
- $\delta$: 
  \[ Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\} \]
- $q_0 \in Q$, start
- $q_{accept} \in Q$
- $q_{reject} \in Q$, $q_{reject} \neq q_{accept}$
- Single $q_{accept}$, $q_{reject}$
The input $w_1 \cdots w_n$

is put in positions $1 \ldots, n$ of the tape in the beginning

Assume $sqcup$ in all the rest of the tape
Example 3.7

Consider the following language

\[ \{0^{2^n} | n \geq 0\} \]

Strings in this language are

0, 00, 0000, 00000000, \ldots

Idea: crossing off every other 0 and the remaining string should still have even length
Example 3.7 II

Example:

0000
00
0

Procedure

1. left → right, mark every other 0
2. if in step 1, only one 0 left, then accept
3. if in step 1, odd ≠ 0 left, then reject
4. move head to the beginning
Example 3.7 III

- go back to stage 1
- Formally definition
  \[ Q = \{ q_1, q_2, q_3, q_4, q_5, q_{accept}, q_{reject} \} \]
  \[ \Sigma = \{ 0 \} \]
  \[ \Gamma = \{ 0, x, \square \} \]
- The diagram
Example 3.7 IV
Example 3.7 V

0 \rightarrow R \equiv 0 \rightarrow 0, R

- Consider the input 0000

\[
\begin{array}{cccccc}
q_1 & 0000 & q_2 & 000 & q_3 & 00 \\
\square & x0 & q_5 & x & \square & xq_3 \\
\quad & x0 & q_5 & 0x & q_5 & x0x \\
\quad & xq_2 & 0x & q_5 & xxxq_3 & \quad \\
\quad & q_5 & xxx & q_5 & \quad & \quad \\
\quad & xxxq_2 & xxx & q_a & \quad & \quad
\end{array}
\]

- The \( \delta \) function:
Example 3.7 VI

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>x</th>
<th>$\sqcup$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$q_1$</td>
<td>$q_2$, $\sqcup$, $R$</td>
<td>$q_{reject}$, $x$, $R$</td>
<td>$q_{reject}$, $\sqcup$, $R$</td>
</tr>
<tr>
<td>$q_2$</td>
<td>$q_3$, $x$, $R$</td>
<td>$q_2$, $x$, $R$</td>
<td>$q_{accept}$, $\sqcup$, $R$</td>
</tr>
</tbody>
</table>

- No need to have rows for $q_{accept}$, $q_{reject}$
  $\Rightarrow$ accepting/rejecting takes immediate effect
- Now a deterministic TM
- We can have nondeterministic TM later
- They are equivalent
- Main idea of $\delta$:
  - $q_1$ : mark the start by $\sqcup$
    - first element must be 0, otherwise, reject
Example 3.7 VII

- Using $\sqcup$, so the start is known
- $q_2 \rightarrow q_3$: handle initial 00
- $q_3 \rightarrow q_4$: sequentially 00 $\rightarrow$ 0x
  - key of this algorithm
  - If not pairs (e.g. 0x0x0x), fails
  - The place where $#$ is even checked
- $q_3 \rightarrow q_5 \rightarrow q_2$ back to beginning
- First 0 (or $\sqcup$) is considered the single final 0

$q_2 \rightarrow \cdots \rightarrow q_2 \rightarrow \cdots \rightarrow q_{\text{accept}}$

check if single 0