Formal Languages, Automata and Computation

Turing Machines

Carnegie Mellon University in Qatar
We now turn to a much more powerful model of computation called Turing Machines (TM).

TMs are similar to a finite automaton, but a TM has an unlimited and unrestricted memory.

A TM is a much more accurate model of a general purpose computer.

Bad News: Even a TM can not solve certain problems.

Such problems are beyond theoretical limits of computation.
Turing Machines
A TM can both read from the tape and write on the tape.
The read-write head can move both to the left (L) and to the right (R).
The tape is infinite (to the right).
The states for rejecting and accepting take effect immediately (not at the end of input.)
**How does a TM compute?**

- Consider \( B = \{ w \# w \mid w \in \{0, 1\}^* \} \).
- The TM starts with the input on the tape.

```
0 1 1 0 0 0 # 0 1 1 0 0 0
X 1 1 0 0 0 # 0 1 1 0 0 0
  → → ⋯
X 1 1 0 0 0 # X 1 1 0 0 0
  ← ← ⋯
X 1 1 0 0 0 # X 1 1 0 0 0
X X 1 0 0 0 # X 1 1 0 0 0
  → → ⋯
X X X X X X X # X X X X X X
```

ACCEPT
A TM is 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{\text{accept}}, q_{\text{reject}})$ where $Q, \Sigma, \Gamma$ are all finite sets.

1. $Q$ is the set of states,
2. $\Sigma$ is the input alphabet (blank symbol $\sqcup \notin \Sigma$),
3. $\Gamma$ is the tape alphabet ($\sqcup \in \Gamma$ and $\Sigma \subset \Gamma$),
4. $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$ is the state transition function,
5. $q_0 \in Q$ is the start state,
6. $q_{\text{accept}} \in Q$ is the accept state,
7. $q_{\text{reject}} \in Q$ is the reject state and $q_{\text{reject}} \neq q_{\text{accept}}$
How does a TM Compute?

- $M$ receives its input $w = w_1 w_2 \cdots w_n$ on the leftmost $n$ squares on the tape. The rest of the tape is blank.
- The head starts on the leftmost square on the tape.
- The first blank symbol on the tape marks the end of the input.
- The computation proceeds according to $\delta$.
- The head of $M$ never moves left of the beginning of the tape (stays there!)
- The computation proceeds until $M$ enters either $q_{\text{accept}}$ or $q_{\text{reject}}$, when it halts.
- $M$ may go on forever, never halting!
Configuration of a TM

- As a TM proceeds with its computation, the state changes, the tape changes, the head moves.
- We capture each step of a TM computation, by the notion of a configuration.

The machine is in state $q_7$, $u = 1011$ is to the left of the head, $v = 01111$ is under and to the right of the head. Tape has $uv = 101101111$ on it.

We represent the configuration by $1011q_701111$. 
Configuration $C_1$ yields ($\Rightarrow$) configuration $C_2$ if TM can legally go from $C_1$ to $C_2$.

$ua q_i bv \Rightarrow u q_j acv$ if $\delta(q_i, b) = (q_j, c, L)$

$ua q_i bv \Rightarrow uac q_j v$ if $\delta(q_i, b) = (q_j, c, R)$

If the head is at the left end, $q_i bv \Rightarrow q_j cv$ if the transition is left-moving.

If the head is at the left end, $q_i bv \Rightarrow cq_j v$ if the transition is right-moving.

Think of a configuration as the contents of memory and a transition as an instruction.
CONFIGURATIONS

- The start configuration is $q_0w$.
- $uq_{accept}v$ is an accepting configuration,
- $uq_{reject}v$ is a rejecting configuration.
- Accepting and rejecting configurations are halting configurations.
**Accepting Computation**

- A TM $M$ accepts input $w$ if a sequence of configurations $C_1, C_2, \ldots, C_k$ exists, where
  1. $C_1$ is the start configuration of $M$ in input $w$,
  2. $C_i \Rightarrow C_{i+1}$, and
  3. $C_k$ is an accepting configuration.

- $L(M)$ is the set of strings $w$ recognized by $M$.

- A language $L$ is **Turing-recognizable** if some TM recognizes it (also called **Recursively enumerable**)

- A TM is called a **decider** if it halts on all inputs.

- A language is **Turing-decidable** if some TM decides it (also called **Recursive**)

- Every decidable language is Turing recognizable!
EXAMPLE TM-1

\[ Q_1 \quad \text{start} \]

- \( X \rightarrow R \)
- \( 0 \rightarrow X, R \)
- \( \# \rightarrow R \)
- \( 1 \rightarrow X, R \)
- \( 0,1 \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow R \)
- \( \# \rightarrow L \)
- \( \# \rightarrow L \)
- \( 0,1 \rightarrow L \)
- \( 0,1 \rightarrow L \)
- \( 0,1 \rightarrow L \)
- \( 0,1 \rightarrow L \)
**Example TM-1**

Let us see how this TM operates on input 001101\#001101