CMSC 141 Automata and Language Theory
Turing Machines

Mark Froilan B. Tandoc

October 24, 2014

Formal Definition of a Turing Machine

A **Turing machine** is a 7-tuple, $(Q, \Sigma, \Gamma, \delta, q_0, \sqcup, q_{accept}$, where:

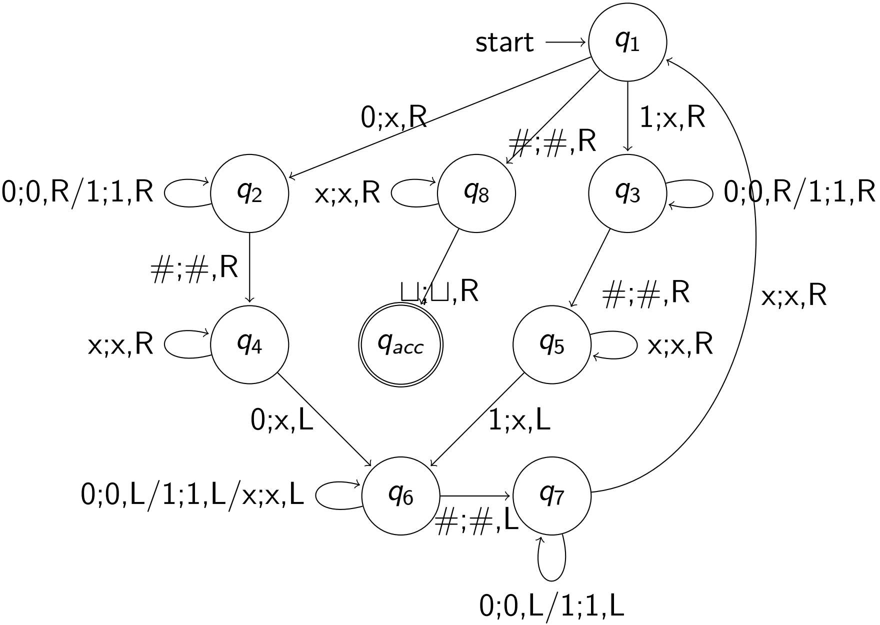
- Q is the set of states,
- Σ is the input alphabet not containing the **blank symbol** $\sqcup$,
- Γ is the tape alphabet, where $\sqcup \in \Gamma$ and $\Sigma \subseteq \Gamma$,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times L, R$ is the transition function,
- $q_0 \in Q$ is the start state,
- $\sqcup$ is the blank symbol
- $q_{accept} \in Q$ is the accept state

- The heart of the definition of a Turing machine is the transition function δ because it tells us how the machine gets from one step to the next
- The formal descriptions of Turing machines are rarely used because they tend to be very big

Example

Turing machine that tests membership in the language $\{w\#w \mid w \in \{0,1\}^*\}$

Example


start
q_1
0;x,R
#;#,R
1;x,R
0;0,R/1;1,R
q_2
x;x,R
q_8
q_3
0;0,R/1;1,R
#;#,R
⊔;⊔,R
#;#,R
x;x,R
x;x,R
q_4
q_{acc}
q_5
x;x,R
0;x,L
1;x,L
0;0,L/1;1,L/x;x,L
q_6
q_7
#;#,L
0;0,L/1;1,L

Turing Machines and Languages

The set of strings that a Turing machine M accepts is **the language of** M, or **the language recognized by** M, denoted by $L(M)$.

These languages are called **Turing-recognizable** or **recursively enumerable** languages

Turing Machines and Languages

- There are three possible outcomes for a TM, *accept, reject, or loop (does not halt)*
- A string can be rejected by entering the rejecting state or by looping
- Its difficult to say when a machine is looping or merely taking long time to compute
- Turing machines that halt on all inputs, those that never loop, are preferred and are called **deciders**

Languages are called **Turing-decidable** or simply **decidable** if some Turing machine decides it

Variants of Turing Machines

- Instead of left or right step for the head, we add a **stay** option
- The tape can extend infinitely both ways
- Multitape Turing machines
  - Simple multiple tapes
  - Multiple tapes with multiple independent heads
  - 2-dimensional tapes that can also add *up* and *down* steps for the head
- Allow non-determinism

None of these "extensions" add real power. They only simplify the programming process.

Another Example

The successor function - Given the binary alphabet $\Sigma = \{0, 1\}$ and any non-empty input string x over Σ representing a binary number, compute for $f(x) = x + 1$

Algorithm:

- Move to the right-most bit
- Flip 1's to 0's and move to the left. Repeat this step until we reach a 0 or a blank
- Replace the 0 or the blank with a 1
- Move to the left-most bit

$\Downarrow$
10111
...
$\Downarrow$
10111
$\Downarrow$
10110
...
$\Downarrow$
10000
$\Downarrow$
11000

REFERENCES

- Previous slides on CMSC 141
- M. Sipser. Introduction to the Theory of Computation. Thomson, 2007.
- J.E. Hopcroft, R. Motwani and J.D. Ullman. Introduction to Automata Theory, Languages and Computation. 2nd ed, Addison-Wesley, 2001.
- E.A. Albacea. Automata, Formal Languages and Computations, UPLB Foundation, Inc. 2005
- JFLAP, www.jflap.org
- Various online LaTeX and Beamer tutorials