<Lecture 22>

TM M computes as follows

- input $w = w_1 w_2 \cdots w_n \in \Sigma^*$ is on the leftmost n squares of the tape, and the rest of the tape is blank ($\sqcup$)
- Initially the head is on the lefmost square

Q How do we know the end of the input?

When computation starts,

- proceeds according to δ.
- If M tries to move beyond the left-end of the tape, it doesn't move.
- continues until q_{accept} or q_{reject} is reached.
- otherwise runs forever

<Configuration>

- Computation changes
  - current state
  - current head position
  - tape contents
- Configuration

  $1011q_7 0111$

  means

C_1 "yields" C_2

- uaq_ibv yields uq_jacv

  if $q_i \xrightarrow{b \to c, L} q_j$

- uaq_ibv yields $uacq_jv$

  if $q_i \xrightarrow{b \to c, R} q_j$

- Special cases

– when the head is at the left-end,

$q_i b v$ yields $q_j c v$ if

$$q_i \xrightarrow{b \to c, L} q_j$$

– when the head is at the right-end,

$u a q_i$ (same as $u a q_i \sqcup$)
yields $u a \sqcup q_j \sqcup$

- Start configuration : $q_0 w$

accepting configuration : $u\, q_{accept}\, v$

rejecting configuration : $u\, q_{reject}\, v$

halting configurations

M accept w if there is a sequence of configurations $C_1 \cdots C_k$ s.t.

- C_1 is the start configuration of M on w
- Each C_i yields C_{i+1}
- C_k is an accept configuration

- The set of strings M accept is called the language recognized by M denoted $L(M)$

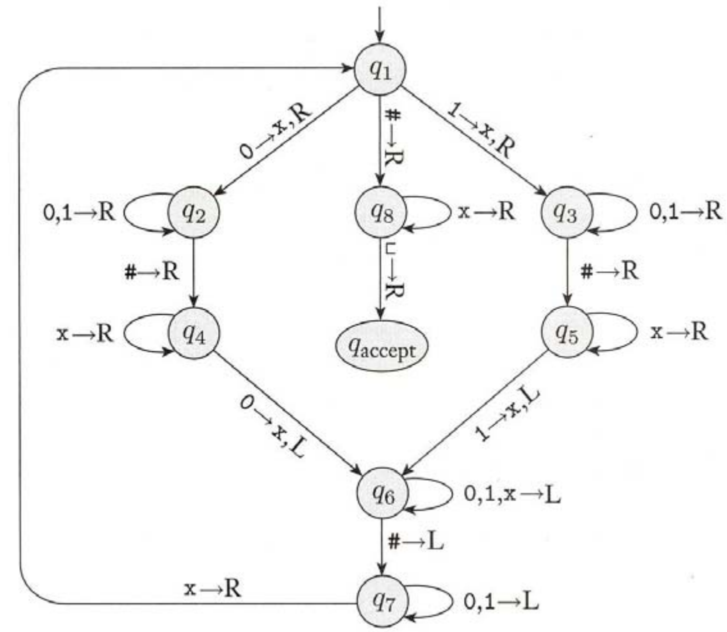
- A language is Turing-recognizable (a.k.a. enumerable) if there is a TM that recognizes it.

Q: When does not M accept w?

A TM decides a language if it recognizes the language and halts for every input.

A language is <u>Turing-decidable</u> (a.k.a. <u>recursive</u>) if there is a TM that decides it.

Example 01#01



3 ways to describe TM.

1) High level : pseudo code of algorithms w/o TM notations

2) Implementation level – describe how TM operates on tape, no explicit mention of state and transitions

3) low level : State diagram

Example 3.12 Element distinctness problem

$E = \{ \# x_1 \# x_2 \# \ldots \# x_\ell \mid \text{each } x_i \in \{0,1\}^* \text{ and } x_i \neq x_j \text{ for each } i \neq j \}$

$\#011\#00\#1111 \in E$

$\#01\#01 \notin E$

Nondeterministic Turing Machines

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, R\})$$

- Computation is a tree
- Each branch corresponds to different possibilities for running NTM
- Accept if some branch leads to the accept state.

Theorem 3.16. NTM and DTM have equivalent expressive power.

Example : NTM that accept

$C = \{ w \in \{0,1\}^* : w \text{ is the binary encoding of a composite number} \}$

$110 = 10 \times 11$

M= "On input w,

1. nondeterministically choose two binary numbers p and q, both greater than 1 s.t $|p| \le |w|$, $|q| \le |w|$.
   Write them on the tape, separated by #.
   (110 # 10 # 11)
2. Multiply p and q and put it after w # (110 # 110)
3. Compare the two numbers, If they are equal, accept ; else reject

Remarks

- Many models have been proposed for general-purpose computation.
- Remarkably, all "reasonable" models are equivalent to Turing machine.
- All "reasonable" programming languages are equivalent.
- The notion of an algorithm is model-independent

Church-Turing Thesis

Formal notion appeared in 1936

- λ-calculus of Alonzo Church
- Turing machine of Alan Turing

They look very different, but are equivalent.

Intuitive notion of algorithms
equals
Turing machine algorithms

Consider Processor X that works like TM except that

- takes first step in 1 second
- takes second step in 1/2 second
- takes i-step in $1/2^i$ second

What is "unreasonable" about this model?

Hint)