EECS 2001A : Introduction to the Theory of Computation

Suprakash Datta

Course page: `http://www.eecs.yorku.ca/course/2001`
Also on Moodle

Turing Machines - Inventor

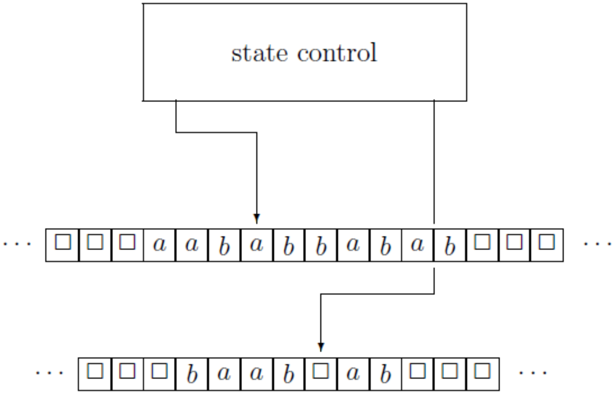
Alan M. Turing (1912–1954)

"On Computable Numbers, with an application to the Entscheidungsproblem"

Turing Machines

- The standard model of computation in theoretical computer science
- More powerful model of computation than NFA, PDA
- "Equivalent" to current computers
- Easier to reason about than modern computers
- We will study them to understand what is solvable using computers

Turing Machines - Structure



A 2-tape Turing Machine

- Tape, tape heads
- Finite state control

Turing Machines - Role of the Tape(s)

- One-way infinite
- Each cell contains one character from the *tape alphabet*
- A head (on on each tape) can move left or right and write characters on the tape
- The head is "controlled" by the finite state machine
- The input is written on a tape at the start of the execution
- Additional tapes are useful as scratch memory
- Output(s) written on the tape

Turing Machines - Things to Note

Note that:

- A TM can execute infinite loops
- Therefore, termination must be explicitly indicated through states
- Usually there are explicit accept and reject states, and these signify termination of execution
- Unless these states are reached the machine cannot be assumed to have terminated

Turing Machines - Execution

- Execution is a sequence of *computation steps* In each step, given the current state r and the k symbols read by the k tape heads,
  - The machine transitions to a state r' (may be the same as r),
  - Each tape head writes a symbol in the cell it is scanning
  - Each tape head moves right or stays in the same cell or moves left (unless it is at the leftmost end of a tape)

Turing Machines - Formal Description

A Turing Machine M is defined by a 7-tuple $(Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$:

- finite set of states Q
- finite input alphabet Σ
- finite **tape** alphabet Γ
- start state $q_0 \in Q$
- accept state $q_{accept} \in Q$
- reject state $q_{reject} \in Q$
- transition function $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$
- If $\delta(q_i, b) = (q_j, c, R)$, the book writes uaq_ibv yields $uacq_jv$

Turing Machines - Language Accepted

$L(M)$ is the set of all strings in Σ^* that are accepted by M.
$w \notin Ł(M)$ if on input w

- the computation of M terminates in the state q_{reject} or
- the computation of M does not terminate

Turing Machines - Example

Detecting palindromes on a 1-tape Turing Machine

- the key in such problems is to think “low level”, not at the level of high-level programs
- the tape is not random access, like an array; it is sequential access, like a doubly linked list
- What are the basic steps? (there are many ways to solve this)