10 Apr 2019

TURING MACHINES AND COMPUTABILITY

Announcements.

1. Prelim 2 tomorrow (Thurs) 7:30–9 pm.
   Phillips 101 and Hollister B14.
   On Piazza we'll announce who goes where, based on netID.

2. Handout for computability theory will be put on website later today. SORRY!

Formalizing "algorithm" and "computer." Began in 1930's.
Alonzo Church: Lambda calculus. (cf. CS 3110 & 4110)
Alan Turing: Turing machine.
These 2 formalisms, and many others, turned out to be computationally equivalent. Any function that could be computed in one of these models could be computed in all of them.
Commonly held belief: these models describe every computational process that can be carried out in our universe — EVEN if our universe is infinitely large and we can harness its infinite data storage capacity. CHURCH–TURING THESIS.

Turing machine: Basically a finite state machine (a.k.a. DFA) augmented with an infinite tape on which it can store & retrieve data.

Have you seen DFA's? (A) Yes (B) No
(in 2800)

Multi-tape TM: (1) Alphabet $\Sigma \supset \Omega$.
symbols M can read/write — input symbols
$\Sigma \setminus \Omega$ contains a "blank symbol", $\_$.
(2) State set, Q. Includes s = starting state, t = terminal state.
(3) Set of tapes, $[T] = \{1, \dots, T\}$.
$I \subseteq T$ "input tapes", $O \subseteq T$ "output tapes",
"Program that M runs" (4) Transition function
$$\delta: Q \times \Sigma^T \longrightarrow Q \times \Sigma^T \times \{+1, 0, -1\}^T$$

Multi-tape TM: (1) Finite Alphabet $\Sigma \supset \Omega$.

symbols M can read/write — input symbols

$\Sigma \setminus \Omega$ contains a "blank symbol", $\_$.

(2) Finite State set, Q. Includes s = starting state, t = terminal state.

(3) Set of tapes, $[T] = \{1, \dots, T\}$.

$I \subseteq T$ "input tapes", $O \subseteq T$ "output tapes",

"Program that M runs" (4) Transition function

$$\delta : \underset{\text{current state}}{Q} \times \underset{\text{what the machine is reading}}{\Sigma^T} \longrightarrow \underset{\text{next state}}{Q} \times \underset{\text{overwrite tapes with new symbols}}{\Sigma^T} \times \underset{\text{move read/write heads,}}{\{+1, 0, -1\}^T}$$

All of this data can be summarized in a finite length binary string called the "description of M",

How do Turing machines compute?

A computation is represented by a sequence of configurations, such that the first config in the sequence is a valid initial config, and each consecutive pair is a valid transition.

Configuration: $(q, \vec{x}, \vec{k})$

$q \in Q$ (current state)

$x = (x_1, \dots, x_T)$

$x_i \in \Sigma_c^\infty$ (contents of tape i)

$k = (k_1, \dots, k_T) \in \mathbb{N}^T$

(locations of read/write heads)

$\Sigma_c^\infty = \{$ infinite sequences of Σ elements that have only finitely many non-blank symbols $\}$

Initial config: $q = s$, $\vec{k} = (0, 0, \dots, 0)$, $x_i = \_^\infty \quad \forall i \in T \setminus I$.

Transition: Let $\vec{\sigma} = (\sigma_1, \dots, \sigma_T)$ $\quad \sigma_i = x_i[k_i]$.

Compute $(q', \vec{\rho}, \vec{\ell}) = \delta(q, \vec{\sigma})$.

New state q', $x_i[k_i] \leftarrow \rho_i$, $k_i \leftarrow \begin{cases} k_i + \ell_i & \text{if } k_i + \ell_i \geq 0 \\ 0 & \text{otherwise.} \end{cases}$

An infinite computation on input x means M never halts.
Denoted $M(x) = \nearrow$

A finite computation is one where $q = t$ at some time m. We say m is the running time.
And if z denotes the contents of the output tapes (discarding every symbol after initial blank on each tape)
we write

$$M(x) = z$$

and call z the output of M on input x.