Introduction to Turing Machines

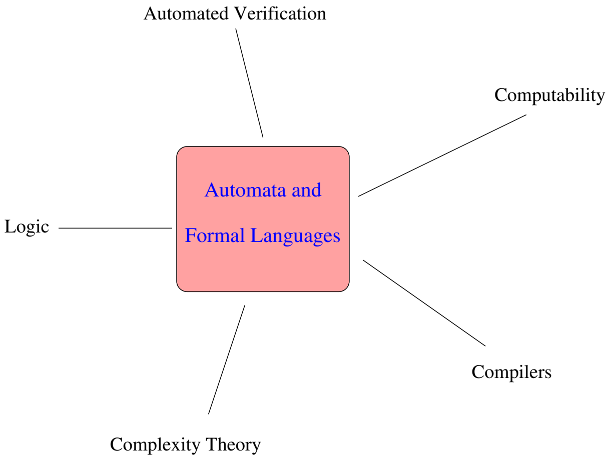
Deepak D'Souza

Department of Computer Science and Automation
Indian Institute of Science, Bangalore.

11 November 2019

Outline

Role of Automata Theory in other subjects



Brief history of logic and computability

David Hilbert 1928: Entscheidungsproblem (deciding validity of FO logic)

Alonzo Church

1936: Undecidability of Entsche
using Lambda–calculus

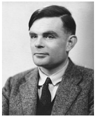
Kurt Godel

1929: Completeness of FO logic
1931: Incompleteness of FO arithmetic
1931: Primitive Recursive Functions

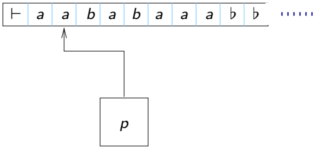
Alan Turing

Kleene, Rosser, Scot, Rabin, ...

1936: Undecidability of Entscheidungsproblem using TMs



How a Turing machine works



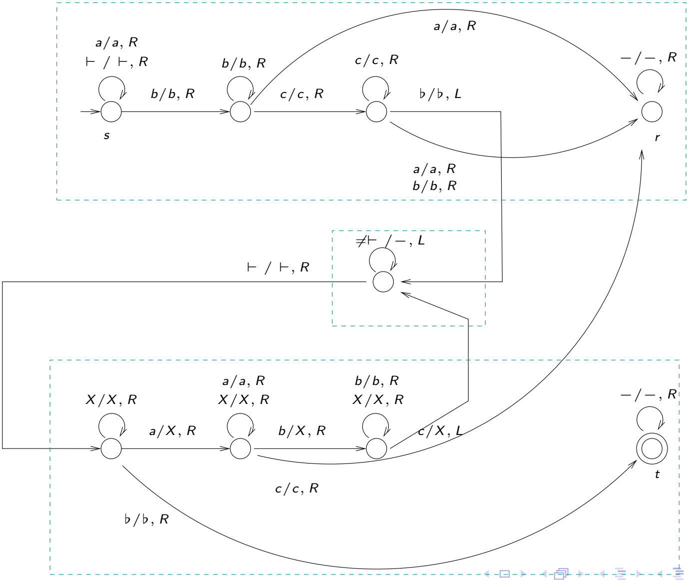
- Finite control
- Tape infinite to the right
- Each step: In current state p, read current symbol under the tape head, say a: Change state to q, replace current symbol by b, and move head left or right.

$$(p, a) \to (q, b, L/R).$$

How a Turing machine works

- Special designated accept state t and reject state r. These states are assumed to be "sink" states.
- TM accepts its input by entering state t.
- TM rejects its input by entering state r.
- TM never falls off the left end of the tape (i.e it always moves right on seeing '$\vdash$').

Example TM for $a^n b^n c^n$



Exercise: TM for adding numbers in unary

Design a TM that accepts $\{1^m\#1^n\#1^{n+m} \mid m, n \geq 0\}$.

Turning machines more formally

A Turing machine is a structure of the form

$$M = (Q, A, \Gamma, s, \delta, \vdash, \flat, t, r)$$

where

- Q is a finite set of states,
- A is the finite input alphabet,
- Γ is the finite tape alphabet which contains A,
- $s \in Q$ is the start state,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ is the (deterministic) transition relation,
- $\vdash \in \Gamma$ is the left-end marker.
- $\flat \in \Gamma$ is the blank tape symbol.
- $t \in Q$ is the accept state.
- $r \in Q$ is the reject state.

Configurations, runs, etc. of a Turing machine

- A configuration of M is of the form $(p, y\flat^\omega, n) \in Q \times \Gamma^\omega \times \mathbb{N}$, which says "$M$ is in state p, with "non-blank" tape contents y, and read head positioned at the n-th cell of the tape.
- Initial configuration of M on input w is $(s, \vdash w\flat^\omega, 0)$.
- 1-step transition of M: If $(p, a) \to (q, b, L)$ is a transition in δ, and $z(n) = a$: then

$$(p, z, n) \stackrel{1}{\Rightarrow} (q, s_b^n(z), n-1).$$

- Similarly, if $(p, a) \to (q, b, R)$ is a transition in δ, and $z(n) = a$: then

$$(p, z, n) \stackrel{1}{\Rightarrow} (q, s_b^n(z), n+1).$$

- M accepts w if $(s, \vdash w\flat^\omega, 0) \stackrel{*}{\Rightarrow} (t, z, i)$, for some z and i.
- M rejects w if $(s, \vdash w\flat^\omega, 0) \stackrel{*}{\Rightarrow} (r, z, i)$, for some z and i.

Language accepted by a Turing machine

- The Turing machine M is said to halt on an input if it eventually gets into state t or r on the input.
- Note that M may not get into either state t or r on a particular input w. In that case we say M loops on w.
- The language accepted by M is denoted $L(M)$ and is the set of strings accepted by M.
- A language $L \subseteq A^*$ is called recursively enumerable if it is accepted by some Turing machine M.
- A language $L \subseteq A^*$ is called recursive if it is accepted by some Turing machine M which halts on all inputs.

Computability and languages

- Notion of a function $f : \mathbb{N} \to \mathbb{N}$ being "computable" (informally if we can give a "finite recipe" or "algorithm" to compute $f(n)$ for a given n.)
- We say f is computable if we have a TM M that given $\vdash 0^n$ as input, outputs $0^{f(n)}$ on its tape, and halts.
- View f as a language

$$L_f = \{(n, f(n)) \mid n \in \mathbb{N}\}.$$

- Then f is computable iff L_f is recursive.