

Study Formulæ: Chapter 9

Philip Gross

December 21, 1998

9 Turing Machines

9.2 definition

A *Turing machine* (TM) is a 5-tuple $T = (Q, \Sigma, \Gamma, q_0, \delta)$, where

Q is a finite set of states, assumed not to contain h , the *halt* state (the same symbol will be used for the halt state of every TM).

Σ and Γ are finite sets, the *input* and *tape* alphabets, respectively, with $\Sigma \subseteq \Gamma$; Γ is assumed not to contain Δ , the *blank* symbol.

q_0 , the initial state, is an element of Q .

$\delta : Q \times (\Gamma \cup \{\Delta\}) \rightarrow (Q \cup \{h\}) \times (\Gamma \cup \{\Delta\}) \times \{R, L, S\}$ is a partial function (that is, possibly undefined at certain points).

Def A *configuration* of $T = (Q, \Sigma, \Gamma, q_0, \delta)$ is a pair $(q, x\underline{a}y)$, where q is a state, x and y are strings over $\Gamma \cup \{\Delta\}$, $a \in \Gamma \cup \{\Delta\}$, and the underlined symbol represents the current position of the tape head.

Def A string $x \in \Sigma^*$ is *accepted* by TM $T = (Q, \Sigma, \Gamma, q_0, \delta)$ if there exist $y, z \in (\Gamma \cup \{\Delta\})^*$, and $a \in \Gamma \cup \{\Delta\}$, so that

$$(q_0, \underline{\Delta}x) \vdash_T^* (h, y\underline{a}z)$$

The *language accepted by T* is the set $L(T)$ of input strings on which T halts.

9.4 Partial Functions

Def Let $T = (Q, \Sigma, \Gamma, q_0, \delta)$ be a Turing machine, and let f be a partial function on Σ^* with values in Γ^* . We say that T computes f if for every $x \in \Sigma^*$ on which f is defined,

$$(q_0, \underline{\Delta}x) \vdash_T^* (h, \underline{\Delta}f(x))$$

and for every other $x \in \Sigma^*$, T fails to halt on input x .

If f is a partial function on $(\Sigma^*)^k$ with values in Γ^* , T computes f if for every k -tuple (x_1, x_2, \dots, x_k) on which f is defined,

$$(q_0, \underline{\Delta}x_1 \Delta x_2 \Delta \dots \Delta x_k) \vdash_T^* (h, \underline{\Delta}f(x_1, x_2, \dots, x_k))$$

Def For any language $L \subseteq \Sigma^*$, the *characteristic function* of L is the function $\chi_L : \Sigma^* \rightarrow 0,1$ defined by the formula

$$\chi_L(x) = \begin{cases} 1 & \text{if } x \in L \\ 0 & \text{otherwise} \end{cases}$$

9.5 MultiTape TMs

Theorem 9.1 Let $n \geq 2$, and let $T_1 = (Q_1, \Sigma, \Gamma_1, q_1, \delta_1)$ be an n -tape Turing machine. Then there is a one-tape TM $T_2 = (Q_2, \Sigma, \Gamma_2, q_2, \delta_2)$, with $\Gamma_1 \subseteq \Gamma_2$, satisfying the following two conditions.

- $L(T_2) = L(T_1)$; that is, for an $x \in \Sigma^*$, T_2 halts on input x iff T_1 halts on input x .
- For any $x \in \Sigma^*$, if

$$(q_1, \underline{\Delta}x, \underline{\Delta}, \dots, \underline{\Delta}) \vdash_{T_1}^* (h, y\underline{a}z, y_2\underline{a}_2z_2, \dots, y_n\underline{a}_nz_n)$$

(for some $a, a_i \in \Gamma_1 \cup \{\Delta\}$ and $y, z, y_i, z_i \in (\Gamma_1 \cup \{\Delta\})^*$), then

$$(q_2, \underline{\Delta}x) \vdash_{T_2}^* (h, y\underline{a}z)$$

In other words, if T_1 halts on input x , then T_2 halts on input x and produces the same output as T_1 .

Corollary 9.1 Any language that is accepted by an n -tape TM can be accepted by an ordinary TM, and any function that is computed by an n -tape TM can be computed by an ordinary TM.

9.6 Nondeterministic TMs

Theorem 9.2 Let $T_1 = (Q_1, \Sigma, \Gamma_1, q_1, \delta_1)$ be a nondeterministic Turing machine. Then there is an ordinary (deterministic) TM $T_2 = (Q_2, \Sigma, \Gamma_2, q_2, \delta_2)$ with $L(T_2) = L(T_1)$.

Sketch of Proof: There are (let's say) two choices at every step of TM execution. This can be represented as an *computation tree* (see p. 278) which our deterministic TM searches in a breadth first pattern. Have T_2 use a 3-tape TM with original input, pseudo- T_1 machine, and pseudo- T_1 tape. Then run five subroutines: init2and3, CopyInput, Execute, EraseTape3, and NextSequence.

9.7 Universal TMs

Def: Encoding function e : Let

- $s(\Delta) = 0$
- $s(a_i) = 0^{i+1}$ (for each $a_i \in \mathcal{S}$)
- $s(h) = 0$
- $s(q_i) = 0^{i+1}$ (for each $q_i \in \mathcal{Q}$)
- $s(S) = 0$
- $s(L) = 00$
- $s(R) = 000$

Each move m of a TM, described by the formula

$$\delta(p, a) = (q, b, D)$$

is encoded by the string

$$e(m) = s(p)1s(a)1s(q)1s(b)1s(D)1$$

and for any TM T with initial state q , T is encoded by the string

$$e(T) = s(q)1e(m_1)1e(m_2)1 \cdots e(m_k)1$$

where m_1, m_2, \dots, m_k are the distinct moves of T , arranged in some arbitrary order,. Finally any string $z = z_1z_2 \cdots z_k$, where each $z_i \in \mathcal{S}$, is encoded by

$$e(z) = 11s(z_1)1s(z_2)1 \cdots s(z_k)1$$

Again, use a three-tape simulator.