Mașina Turing Universală

M.T. → (K, Σ, δ, s)

cum K, Σ → mulțimi finite ⇒ pot să repr. ca un șir, codificând stările și simb. de intrare peste un alfabet fixat.

Convenții

pp. că există mulț. infinit numărabile

$$K_\infty = \{q_1, q_2, q_3, \dots\}$$

$$\Sigma_\infty = \{a_1, a_2, a_3, \dots\}.$$

ai pt fiec. M.T. mulțimea stărilor este o submulțime finită din K_∞, alf. de intrare → submulțime finită din Σ_∞.

Codific stările, simb de intrare ca șiruri peste alf. $\{I\}$

σ	$\lambda(\sigma)$
q_i	I^{i+1}
h	I
L	I

σ	$\lambda(\sigma)$
a_i	I^{i+2}

Fie c → alt simbol

Codificăm M.T. peste alf. $\{I, c\}$

$M = (K, \Sigma, \delta, s)$, $K \subseteq K_\infty$, $\Sigma \subseteq \Sigma_\infty$

$K = \{q_{i_1}, q_{i_2}, \dots, q_{i_k}\}$, $i_1 < i_2 < \dots < i_k$

$\Sigma = \{a_{j_1}, a_{j_2}, \dots, a_{j_l}\}$, $j_1 < j_2 < \dots < j_l$

$s = q_{i_m}$, $1 \le m \le k$

Def. kl șiruri S_{pr}, $1 \le p \le k$, $1 \le r \le l$

S_{pr} → codifică valoarea funcției de tranziție pt. perechea (stare, simbol), respectiv (q_{i_p}, a_{j_r})

$\delta(q_{i_p}, a_{j_r}) = (q', b)$, $q' \in K \cup \{h\}$
$b \in \Sigma \cup \{L, R\}$

$S_{pr} = c\, w_1 c\, w_2 c\, w_3 c\, w_4 c$,

$w_1 = \lambda(q_{i_p})$
$w_2 = \lambda(q_{i_r})$

$w_3 = \lambda(g')$

$w_4 = \lambda(h)$

$f(M) = c\, s_0\, c\, s_{11}\, s_{12} \dots s_{1e}\, s_{21}\, s_{22} \dots s_{2e} \dots s_{k1}\, s_{k2} \dots s_{ke}\, c$

↑ codific. M.T.

$s_0 = \lambda(\Delta) \to$ codific. st. inițiale

din $f(M)$ se poate reconst. M.

Pot. utiliza M.T. Universală → folosește codificarea $f(M)$ a unei alte M.T. cu programe.

intuiții → M.T. Universală (U) primește 2 arg. → o descriere a unei M.T. M și un șir de intrare w și execută opțiile pe care le-ar fi executat M.

Atât M.T. cât și w → codificate

$w = h_1 \dots h_n, \quad h_i \in \Sigma_\infty$

$f(w) = c\,\lambda(h_1)\, c\,\lambda(h_2)\, c \dots c\,\lambda(h_n)\, c$

Proprietatea pe care $U = (K_U, \Sigma_U, \delta_U, s_U)$ trebuie să o aibă:

$\forall M = (K, \Sigma, \delta, s)$, $\forall w \in \Sigma^*$

1° Dc $(h, u\underline{a}v)$ este o config. de oprire a lui M ai

$$(s, \#w\underline{\#}) \vdash_M^* (h, u\underline{a}v)$$

Atunci $(s_U, \#\rho(M)\rho(w)\underline{\#}) \vdash_U^* (h, \#\rho(uav)\underline{\#})$

2° Dc $(s_U, \#\rho(M)\rho(w)\underline{\#}) \vdash_U^* (h, u'a'v')$

atunci $a' = \#$, $v' = e$, $u' = \#\rho(uav)$

pt u, a, v ai $(h, u\underline{a}v)$ este o config. de oprire a lui M și

$$(s, \#w\underline{\#}) \vdash_M^* (h, u\underline{a}v)$$

Funcționarea M.T. Universale $\rightarrow$ utiliz. o M.T. cu 3 benzi U

U utilizează benzile astfel:

$\rightarrow$ 1-ma bandă conține codific. benzii lui M

$\rightarrow$ 2-a banda $\rightarrow$ conține codific. lui M.

$\rightarrow$ 3a bandă $\rightarrow$ conține codific. st. lui M la pasul curent al simulării

$U' \to$ pornește cu $\# f(M) f(w)$ pe 1ma bandă și celelalte benzi goale.
U' pune $f(M)$ pe a 2a bandă și deplas. $f(w)$ la capatul stg. al primei benzi precedat de $\# c f(\#)$ și încheiat $\lambda(\#) c$.

<u>1ma bandă</u> $\to$ $\# f(\# w \#)$
Din $f(M) \to U'$ extrage codific. st. init. a lui M $\to$ pune pe a 3a bandă.
U' simulează pașii aplic lor lui M. Intre pașii simulați, U' păstrează capetele coresp. $B2$, $B3$ la lim. stg., capul $B1$ în dreptul c-ului care marchează st. codific. simb. scanat de M crt.
U' gasește pe a 2a bandă

$$c c \underbrace{I^i}_{\downarrow} c \underbrace{I^j}_{\to B1} c I^k c I^l c c$$

șirul de I-uri care se termină la poz. crtă a capului pe $B3$

Dc $I^l \to \lambda(L)$ sau $\lambda(R)$ $\to$ muta capul cu cateva simb. stg/dr
$I^l \to \lambda(a)$, $a \in \Sigma_\infty$ $\Rightarrow$ în loc codific. pe $B1$ (cu shiftare!)
In final U' pune I^k pe a 3a bandă.

Uncomputability

Teorema

Orice limbaj Turing decidabil este Turing acceptat.

dem.

Fie $L \to$ limbaj decis de M.T. M, L este acceptat de M.T. M'

$$\begin{tikzcd} > M\,L \arrow[r, "N"] & N \arrow[loop right] \end{tikzcd}$$

Teorema

Dacă L este Turing decid. atunci $\bar{L}$ (complement) este Turing-decid.

dem.

$L \to$ decis de M.T. M, $\bar{L} \to$ decis de M.T. $\bar{M}$

$$\begin{tikzcd} > M\,L \arrow[r, "Y"] \arrow[d, "N"] & N\,R \\ Y\,R & \end{tikzcd}$$

Întrebări:

Este orice limbaj Turing accept. Turing decidabil?

Este complement oricărui lbj Turing acceptat, Turing acceptat? → NU

Dc. M_1 ar fi o M.T. care acceptă L atunci putem proiecta o M.T. M_2 care decide L astfel: M_2 execută opțiile necesare pt. a prezice event. rezultat al opții lor lui M_1 pe intr. w;

M_2 s-ar opri cu (Y) sau (N) pe bandă în fcție de accept a lui w de către M_1.

Revine la întrebarea dc ∃ o super M.T. care prezice rezult. opții lor efectuate de M.T. arbitrare pe intr. arbitrare.

⇒ dc ∃ o "super" M.T. care prezice astfel de rez:

$$K_0 = \{ \rho(M) \rho(w) \mid M \text{ se oprește pe șirul } w \}$$

este decis de o M.T. M_0 ⇒ ∀ L. T-acceptat este T-decidabil.

Pt orice M.T. M_1 care acceptă limb. L, putem construi o M.T. M_2 care decide L, astfel:

a) construim M.T. M_1^* care $\forall w$, transformă $\# w \#$ în $\# f(M_1) f(w) \#$

Fie $M_2 = M_1^* M_0$ => transf $\# w \#$ în $\# f(M_1) f(w) \#$ și cedează controlul lui M_0 care prin ipoteză decide dc M_1 acceptă w.

Se poate dem. $K_0 \to$ Turing acceptat => acceptat de o variantă a M.T.O.

$\forall$ limbaj Turing-acceptat este Turing decidabil <=> K_0 Turing acceptat este Turing decid.

<u>Arătăm</u> $K_0 \to$ <u>nu</u> este Turing decid.

Pp. $K_0 \to$ Turing decid. =>

$K_1 = \{ f(M) \mid M \text{ acceptă } f(M) \}$ Turing decid.

Dc. M_0 ar decide K_0, atunci M_1 care decide K_1 poate fi construită

=> transformă $\# w \# \Longrightarrow \# w f(w) \#$ și cedează contr. lui M_0.

M_1 obține acelaș rez. pt $\# w \#$ ca M_0 pt $\# w f(w) \#$

Din def. K_0, M_0 obț (*) pt $\# w f(w) \#$ <=>:

a) w este $f(M)$ pt o an. M.T. M

f) M.T. M acceptă w, adică $f(M)$

Dar asta este def. lui K_1. Este suf. să dem. K_1 nu este decid. Turing

pp. $K_1 \to$ Turing decid. $\Rightarrow$ $\overline{K_1}$ Turing-decid.

$$\overline{K_1} = \{ w \in \{I, c\}^* \mid w \text{ nu este codific nici unei M.T. } M \text{ sau } w = f(M) \text{ pt o an. M.T. } M \text{ care nu acceptă } f(M) \}$$

Dar $\overline{K_1}$ $\to$ nu poate fi Turing acceptat.

Să pp. că M^* este o M.T. care acceptă $\overline{K_1}$. Este $f(M^*)$ în $\overline{K_1}$?

Din def. lui $\overline{K_1}$, $f(M^*) \in \overline{K_1}$ <=> M^* nu acceptă $f(M^*)$.

Dar M^* se pp. că $\overline{K_1}$, $f(M^*) \in \overline{K_1}$ <=> M^* acceptă $f(M^*)$

=> M^* acceptă $f(M^*)$ <=> M^* nu acceptă $f(M^*)$ => absurd.

Teorema

Nu orice limbaj Turing acceptat este Turing decidabil.

Teorema

Compl. unor limbaje Turing accept. nu sunt Turing acceptate.

(pt că K_1, K_0 → Turing accept. dar $\bar{K_1}$ nu este Turing acceptat.)

Probleme care nu admit sol. alg. ⇒ problema opririi M.T., descrisă de K_0.

Teorema

a) Nu există niciun alg. care să det. dată fiind o M.T. M și un șir w, dc M acceptă w.

b) Pt o M.T. fixată M_0, nu există niciun alg. care să det. dat fiind w, dc M_0 acceptă w.

$K_0 = \{ f(M) f(w) \mid M$ acceptă $w \}$

$\bar{K_0}$ → poate fi spart în infinit de multe submulțimi

$K_M = \{ f(M) f(w) \mid M$ acceptă $w \}$

câte una pt fiec M.T. M.

Nu rezultă imediat din faptul că $\bar{K_0}$ → nu este Turing decid.

că anumite submulțimi K_M nu sunt Turing decid.

ex. limbajul care constă dintr-un sg. șir ⇒ decis de M.T. care verifică

că intrarea este acel șir

Nu există o met. generală de a combina inf. de multe proc de

decizie într-una sing.