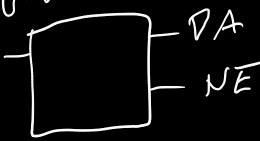
IRZUFRI 2024-12-19

[(ne)izračunljivost]

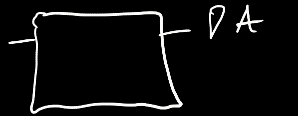
različne vrste strojev:

odločljivi



vrne da, če je beseda v jeziku, sicer ne

polodločljivi:



vrne da, če je beseda v jeziku, sicer se zacikla

univerzalni jezik je jezik univ. turingovega stroja

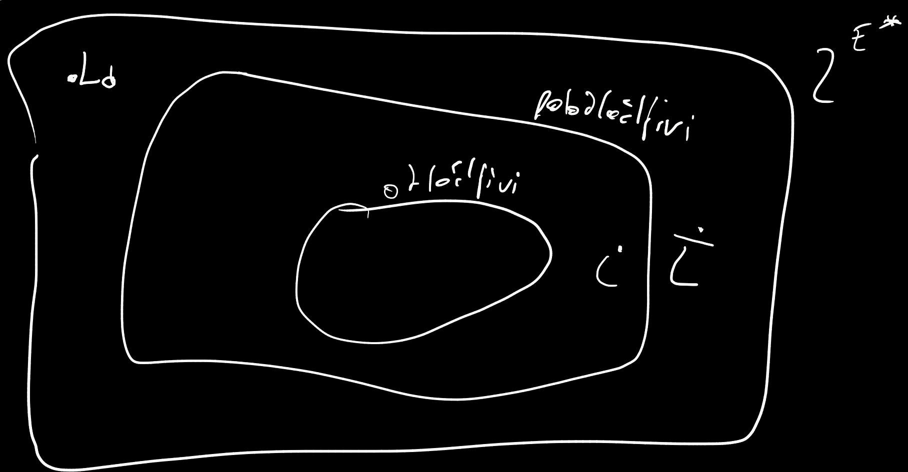
$$L_u = \{\langle M, w\rangle ; w \in L(M)\}$$

stroj za ta jezik obstaja, a le tako, da simulira M — je torej polodločljiv.

dokazali bomo, da se ne da bolje, kot simulirati M.

Izrek: L odločljiv $\Rightarrow$ $\bar{L}$ odločljiv.

Izrek: L in L^c polodločljiva $= ????$ ...



L^d je jezik, ki nima algoritma. Gotovo obstaja, ker je turingovih strojev $|\mathbb{N}|$, problemov pa $|2^{\mathbb{N}}|$ (ne obstaja bijekcija $\Rightarrow |\mathbb{N}| < |2^{\mathbb{N}}|$)

definirajmo L_d: 2^{Σ^*} pripadajoč jezik

besede v Σ^* oštevilčimo.

Turingov stroj	w_1	w_2	w_3	w_4	w_5	...
M_1	0	1	0	0	0	
M_2	0	0	1	0	0	
M_3	0	1	0	0	1	...
M_4	1	1	1	0	0	
M_5	0	0	0	0	1	
⋮			⋮			

negacija te diagonale je diagonalni jezik in nima pripadajočega turingovega stroja.

$$L_d := \{ w_i \in L_d \iff w_i \notin L(M_i) \}$$

L_d se od vsake vrstice razlikuje za natanko eno besedo.

Prevedba: je relacija

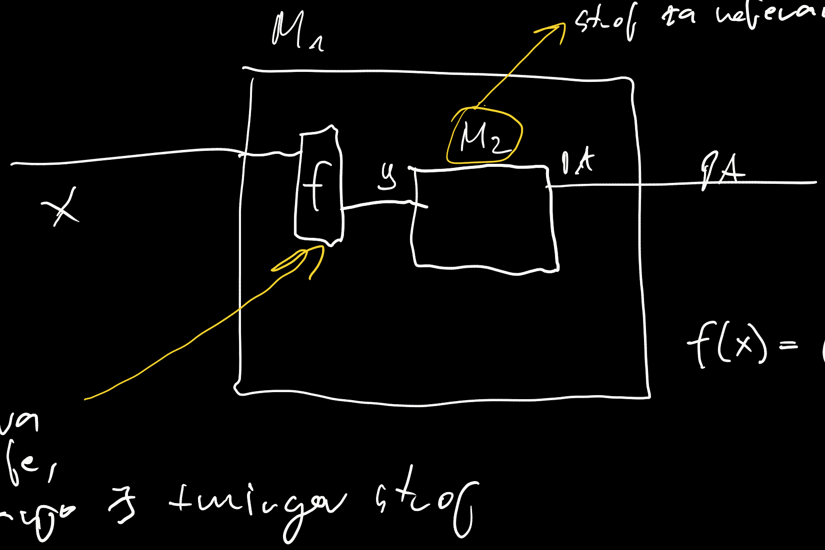
$L_1 \leq L_2$ ~ L_2 je vsaj toliko težek kot L_1

tvori delno urejenost.

[alternativna oznaka $L_1 \to L_2$

če za L_2 ne obstaja TS in $L_1 \leq L_2 \Rightarrow$ za L_1 $\nexists$ TS.

implementiramo stroj za reševanje L_1:



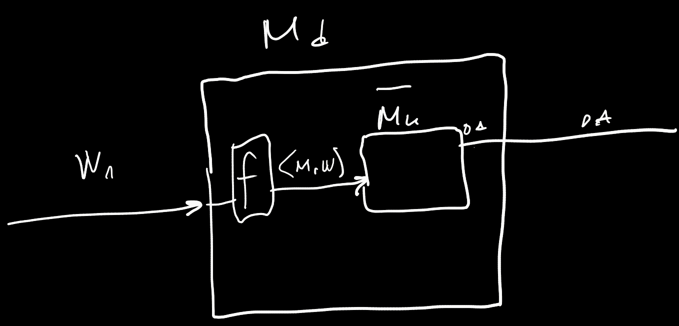
$$f(x) = y \in L_2 \Leftrightarrow x \in L_1$$

1. M_2 obstaja $\Rightarrow$ M_1 obstaja „programiranje"
   ? ? ?
2. M_1 ne obstaja $\Rightarrow$ M_2 ne obstaja

Pokažimo, da $\overline{L_u}$ $\nexists$

$$\overline{L_u} = \{ \langle M, w \rangle ; w \notin L(M) \}$$

naredimo prevedbo $L_d \leq \overline{L_u}$ $(L_d \rightarrow \overline{L_u})$



$$w_i \in L_d \Leftrightarrow \langle M, w \rangle \in \overline{L_u}$$

$$\Updownarrow$$

$$w_i \notin L(M_i) \Leftrightarrow w \notin L(M)$$

f: $w = w_i$
$M = M_i$

torej velja $\exists_{TS} \overline{M_u} \Rightarrow \exists_{TS} M_d$

torej posledično $\nexists_{TS} M_d \Rightarrow \nexists_{TS} \overline{M_u}$

torej, ker $\nexists$ TS M_d, velja $\nexists$ TS $\overline{M_u}$.

funkcija $M_d(w_i)$:

$w := w_i$

$M :=$ izdelaj TS(i)

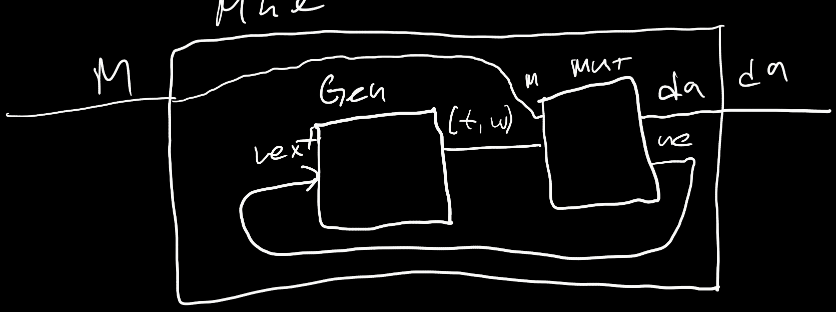
vrni $\overline{M_u}(w, M)$

f je TS

N

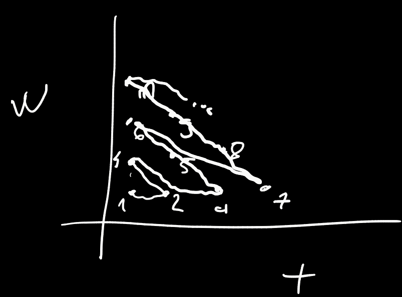
non empty

$L_{ne} = \{ \langle M \rangle ; L(M) \neq \emptyset \}$



Mne sprejme M, če je neprazen, sicer se zacikla.

Gen generira vse besede za vse možne timeoute.



$M_{u,t}(M,w)$ pove v končnih korakih, če M v t korakih sprejme w.

Dokažimo, da ni moč narediti M_{ne} tako, da bi bil razreda določenih problemov.

ogledimo si komplement tega jezika.

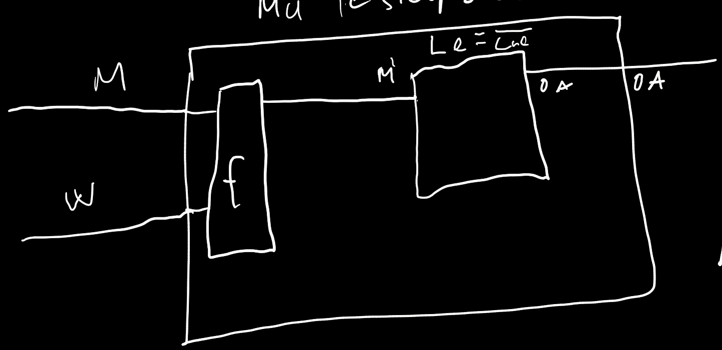
$L_e := \overline{L_{ne}}$ (empty)

↳ vsi programi, katerih jezik je prazen.

dokažimo, da $\nexists$ TS za L_e.

za $\overline{L_u}$ smo dokazali, da $\nexists$ TS zanj. (je neodločljiv)

naredimo prevedbo $\overline{L_u} \leq L_e$; $\overline{L_u} \to L_e$.



to želimo.

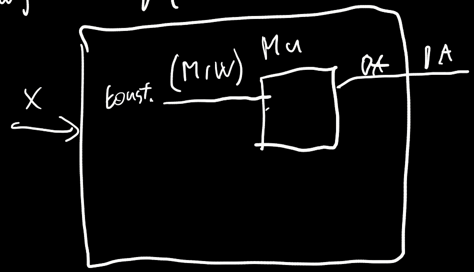
$L(M') = \emptyset \Leftrightarrow w \notin L(M)$

najdimo f, da bo

$L(f(M,w)) = \emptyset \Leftrightarrow w \in L(M)$

f da stroj. bi je prazen $\Leftrightarrow w \notin L(M)$

$f: (M,w) \mapsto M'$



če $w \in L(M) \Rightarrow L(M') = \Sigma^*$

če $w \notin L(M) \Rightarrow L(M') = \emptyset$

```
def Mue(M):
    nestonično ponavljaj:
        w, t = genNext();
        če M_{u,t}(M, w):
            vrni DA
```

in Mue obstaja

∃ ts

```
def ‾Mu‾(M, w):
    M' := (x) => { return Mu(M, w) }
    return Me(M')
```

toda ker za $\overline{M_u}$ ∄ TS, sledi, da za M_e ∄ TS.