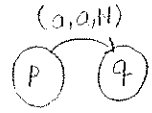
• Calculabilitat i decidibilitat.

MT ≡ DFA amb la possibilitat de moure el capçal i modificar els símbols.

$M = (Q, \Sigma, \Gamma, \delta, q_0, q_F)$! Només un estat final.

$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times (E, D, N)$

$\delta(p,a) = (q,a,n)$



$(q,a) \in \mathrm{Dom}\,\delta$ si la funció de transició no està definida	$\Rightarrow M(\omega)\downarrow$ $\Longrightarrow$	Estat Final no té transicions definides

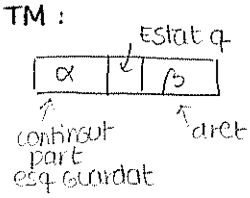
– Funció calculada per una MT – És el que queda a la cinta quan la màq s'atura.

$$f_M = \begin{cases} M(x) & \text{si } M(x)\downarrow \\ \uparrow & \text{altrament} \end{cases}$$

– llenguatge reconegut: $\rightarrow L(M) = \{\omega \in \Sigma^* \mid M(\omega)\downarrow$ i accepta$\}$
↳ Para en estat final

Calcular una codificació d'una TM :

$\omega = \alpha \cdot q \cdot \beta \in \Gamma^* Q \Gamma^*$



codificació inicial : $\triangleright q_0 x$

configuració final : $\alpha q_F \beta \vdash \alpha' q' \beta'$ ← en 1 pas de computació $\quad \vdash^t$ t passos $\quad \vdash^*$ un nombre desconegut

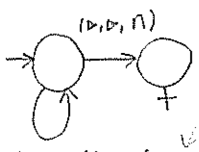
Redefinim L(M) : $L(M) = \{\omega \in \Sigma^* \mid \exists \alpha\beta \in \Gamma^* : \triangleright q_0 \omega \vdash^* \alpha q_F \beta\}$

! La funció calculada – no ens importa que accepti o no (s'aturi en estat final o no)

EXEMPLE :

$\Sigma = \{0,1\}$

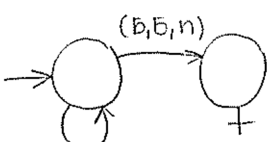
a)



com queda transformada l'entrada a la cinta.

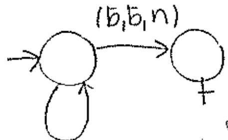
$f_M(x) =$ indefinida
$L(M) = \emptyset$ no arribem a l'estat final

b)



$f_M(x) = x$
$L(M) = \{0^*\}$
funció identitat
$\Rightarrow$ tot el que entra es queda igual

c)



$L(M) = \Sigma^* = \{0,1\}^*$
$f_M(x) = 0^{|x|}$

conjunt de totes les entrades del sobre l'alfabet d'entrada, per les quals m assoleix estat acceptació

Sigui $M \in TM$ i $x \in \Sigma^*$. Digueu si són certes les següents afirmacions :

ⓐ Si $M(x)$ no repeteix configuració aleshores $M(x)\downarrow$
Fals. contraexemple apartat a ! $\Rightarrow$! repetir conf $\Rightarrow \alpha q \beta \vdash^* \alpha q \beta \Rightarrow$ al tirar cap a la dreta anem incrementant α i decrem β.

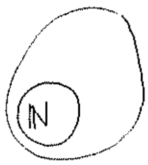
ⓑ Si $M(x)\downarrow$ aleshores $M(x)$ no repeteix config
Cert.
Demostrem el contrari : si $M(x)\uparrow$ llavors repeteix configuració
$\Longrightarrow$ si tenim una màquina que rep conf $\Rightarrow$ la màq es penja $\quad \alpha q \beta \vdash^* \alpha q \beta$

© Si $M(x)$ no repeteix configuració $\Rightarrow M(x)\uparrow$

Fals. contraexemple

Aturada segura: maq. que amb qualsevol entrada s'atura per qualsevol entrada. definida al alfabet d'entrada.

= **Núm. de Godel**



El conjunt de TM és enumerable $\Leftarrow$ (ja que una TM es pot codificar amb $w \in \{0,1\}^*$ $\forall \Sigma : \Sigma^*$ és enumerable)

Núm. Godel d'una TM = posició de la llista indexada de TM. (ordenació)

M_x = TM de núm. Godel x

$\varphi_x = f_x$ = funció calculada per M_x

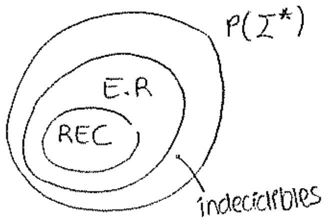
$L_x = L(M_x)$ llenguatge reconegut per M_x acceptat per M_x

$K = \{x \mid M_x(x)\downarrow\}$

$HALT = \{\langle x,y\rangle \mid M_x(y)\downarrow\}$

$PERT = \{\langle x,y\rangle \mid M_x(y)\downarrow$ i accepta$\}$
$= \{\langle x,y\rangle \mid y \in L(M_x)\}$

= **Decidibilitat i E.R**



$E.R = \{L \subseteq \Sigma^* \mid \exists_{M \in TM} : L(M) = L\}$

$REC = \{L \subseteq \Sigma^* \mid \exists_{M \in TM} : L(M) = L \wedge \forall x\ M(x)\downarrow\}$
reconeix el lleng. i és de parada segura.

Tancament:

Si $A, B \in REC$ (Decidibles) $\Rightarrow A\cup B, A\cap B, A-B, \bar{A}, \bar{B} \in REC$

Si E.R $\Rightarrow A\cup B, A\cap B \in$ E.R

K, Halt i Pert no $\in$ a REC
$\Rightarrow$ ja que les seves funcions característiques no són calculables.

Teorema del complementari

Si $A \in ER$ i $\bar{A} \in E.R \Leftrightarrow A \in REC$

Teorema de la projecció

$L \in ER \Leftrightarrow L = \{x \mid \exists y : \underbrace{(x,y) \in B}_{\text{Predicat recursiu}}\}$, $B \in REC$

1. simular(M,x): simulem fins que M s'atura
2. simular(M,x,t): simular t passos de computació de la màquina M amb entrada x.

1. simular(M,x)
Entrada $\langle w,x\rangle$ — w: codif, x: paraula
c := configuració_inicial (M_w,x)
mentre $\neg$(c es conf. terminal) **fer**
  c := configuració_següent (M_w,x,c)
fmentre

Simular M, x, t

$c :=$ configuració_inicial
$i := 0$.
<u>mentre</u> $\neg$(c es conf terminal) $\wedge$ $i < t$ <u>fer</u>
  $c :=$ configuració_següent(M, x, c)
  $i := i+1$
<u>fmentre</u>

✓ accepta en t o menys passos

EXEMPLES:

- $K = \{x \mid M_x(x)\downarrow\}$ x: núm Gödel
  $\equiv \{x \mid \exists t : \underbrace{M_x(x)\downarrow \text{ en } t \text{ passos}}_{\text{Entrades}: t, x}\}$ Queda de l'estil Teor Projecció

  Entrada x, t
  simular t passos de $M_x(x)$
  si $M_x(x)\downarrow$ en t passos $\Rightarrow$ Acceptar
  sino REBUTJAR
  fsi

  màq. d'aturada segura
  $\{x \mid \exists t : M_x(x)\downarrow$ en t passos$\} \in$ REC
  $\Downarrow$
  K és E.R.

- $L = \{x \mid \text{Dom } \varphi_x \neq \emptyset\} \equiv \{x \mid \exists y : M_x(y)\downarrow\} \in$ E R
  $\equiv \{x \mid \exists y, t : \underbrace{M_x(y)\downarrow \text{ en } t \text{ passos}}_{B}\}$

  Entrada x, t, y
  simular t passos de $M_x(y)$
  si $M_x(y)\downarrow$ en t passos $\rightarrow$ Acceptar
  sino Rebutjar
  fsi

  T. Projecció
  B $\in$ REC $\Rightarrow$ (L $\in$ ER)

- $L = \{x \mid \text{Dom } \varphi_x \cap K \neq \emptyset\} \equiv \{x \mid \exists y : y \in \text{Dom } \varphi_x \wedge y \in K\}$
  $\equiv \{x \mid \exists y : \underbrace{\underbrace{M_x(y)\downarrow}_{\text{Halt}} \wedge \underbrace{M_y(y)\downarrow}_{K}}_{\text{Hi ha un núm de passos que fan que la màq. s'aturi}}\} \equiv \{\exists y, t_1, t_2 : M_x(y)\downarrow$ en t_1 passos $\wedge$
  $\underbrace{M_y(y)\downarrow \text{ en } t_2 \text{ passos}\}}_{\text{REC}}$
  $\Downarrow$
  L $\in$ E R

  <u>Entrada</u> x, y, t_1, t_2
  simular t_1 passos de $M_x(y)$
  simular t_2 passos de $M_y(y)$
  si $\underset{\text{en } t_1 \text{ p}}{M_x(y)\downarrow} \wedge \underset{\text{en } t_2 \text{ p}}{M_y(y)\downarrow} \longrightarrow$ Acceptar
  sino rebutjar

i) Donada una TM determinar si accepta algun mot de longitud parell

ii) $L = \{x \mid Dom\ \varphi_x \cap Im\ \varphi_x \neq \emptyset\}$ $\longrightarrow Im\ \varphi_x \equiv \{y \mid \exists z : M_x(z)\downarrow \text{ i } M_x(z) = y\}$

paraules que pertanyen al llenguatge

i) Formalització del llenguatge

$L_i = \{x \mid |x| = \dot{2} \wedge M_x(x)\downarrow \wedge \text{accepta}\}$

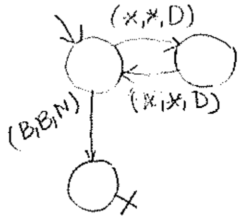
$\equiv \{x \mid \exists y,t : |y| = \dot{2} \wedge \underbrace{M_x(y)\downarrow \text{ i accepta en } t \text{ passos}}_{REC}\}$

Entrada x, t, y

simular $M_x(y)$ en t passos i $|y|$ és parell

si $M_x(y)\downarrow$ en t passos $\rightarrow$ Acceptar

sino rebutjar



ii) $\equiv \{x \mid \exists y,z,t_1,t_2 : M_x(y)\downarrow \text{ en } t_1 \text{ passos} \wedge M_x(z)\downarrow \text{ en } t_2 \text{ passos} \wedge M_x(z) = y\}$

$B \in REC \Rightarrow L \in ER$

- Altra versió del T. de la Projecció:

$TP_1 : L \in ER \iff L = \{x \mid \exists y : (x,y) \in B\},\ B \in REC$

$\Leftarrow$:

$L = \{x \mid \exists y : (x,y) \in B\},\ B \in ER \overset{TP}{\Longrightarrow} B = \{(x,y) \mid \exists z : (xyz) \in C\}\ C \in REC$

$\overset{TP}{\Longrightarrow} L = \{x \mid \exists y,z : (x,y,z) \in C\}\ C \in REC \Rightarrow L$ és ER.

$\Rightarrow$:

L és ER $\Rightarrow L = \{x \mid x \in L\}$ L és ER.

Reduccions : comparador de complexitat

definida x totes les entrades

$A \leq_m B \iff \exists f : A \rightarrow B$ total i calculable que compleix

$x \in A \iff f(x) \in B$

"A és com a molt tan difícil com B"

Diem que A es redueix a B quan existeix una funció total i calculable que passa entrades de A a entrades de B, i es compleix que:

$x \in A \iff f(x) \in B$

$\forall x \in \Sigma^*$

Propietats de tancament :

$A \leq_m B \wedge B \in REC \Rightarrow A \in REC$

$A \leq_m B \wedge B \in ER \Rightarrow A \in ER$

$A \leq_m B \wedge A \notin REC \Rightarrow B \notin REC$

$A \leq_m B \wedge A \notin ER \Rightarrow B \notin ER$

K, HALT, PERT són ER però no són REC

$\overline{K}, \overline{HALT}, \overline{PERT}$ no són ER

$K \leq_m A \Rightarrow A \notin REC$

$\overline{K} \leq_m A \Rightarrow A \notin ER$

Conjunts d'índexs

Són subconjunts de $\mathbb{N}$ (o Σ^*) de la forma $\{x \in \mathbb{N} \mid P(\varphi_x)\}$. La propietat P, que determina la pertinença d'un elem x al conjunt, fa referència a la funció computada per la TM donada. $\Rightarrow$ la pertinença o no d'un element depén de propietats relatives a la funció computada per TM (o del lleng. reconegut) i no del seu índex o núm. de Gödel.

només 1 TM com a entrada.

A és conjunt d'índex si: A conté índexs de TM $A = \{x \mid \ldots\}$
$$\forall i,j: \varphi_i = \varphi_j \quad i \in A \Rightarrow j \in A$$
$$\forall i,j: L_i = L_j \quad i \in A \Rightarrow j \in A$$

Un llenguatge diem que és cnj d'índexs si conté índexs de TM i la pertinença d'un índex x al llenguatge depén només de la funció computada o del lleng. reconegut per la TM M_x.

La família d'índexs d'una funció f està formada per totes les descripcions de TM que computen la funció f.

Exemples:

- $A = \{x \in \mathbb{N} \mid M_x$ té 10 estats$\}$
  Està clar que podem afegir a Mx tants estats com vulguem sense canviar per això el resultat de la seva computació.

- K no és c. índexs
  perquè una TM $\in$ K depén del seu núm de Gödel i llavors pot ser que dues màq. que calculen la mateixa funció una $\notin$ K i l'altra no
  $Dom(\varphi_{x_0}) = \{x_0\}$ aleshores x qualsevol $y \neq x_0$ tq. $\varphi_y = \varphi_{x_0} \Rightarrow y \notin K$

- $L = \{x \mid Dom\, \varphi_x \neq 0\}$ és CI
  Siguin i,j índexs de TM tals que:
  $$\forall i,j: \varphi_i = \varphi_j \wedge i \in L \overset{?}{\Rightarrow} \text{cal demostrar que } j \in L$$
  $$\Rightarrow Dom\,\varphi_i = Dom\,\varphi_j$$
  $$\Rightarrow Dom\,\varphi_i \neq 0 \Rightarrow Dom\,\varphi_j \neq 0 \Rightarrow j \in L$$

- $L = \{x \mid \exists y: |y| = \dot{2} \wedge y \in L_x\}$ és CI
  Siguin i,j índexs de TM tals que:
  $$\boxed{\begin{matrix}\varphi_i = \varphi_j \\ L_i = L_j\end{matrix}} \wedge i \in L \Rightarrow \exists y: |y| = \dot{2} \wedge y \in L_i$$
  $$\Rightarrow \exists y: |y| = \dot{2} \wedge y \in L_j \Rightarrow j \in L$$

- $L = \{x \mid \exists y: \varphi_x(y) = y\}$ funció identitat si fos $\forall y: \varphi_x(y) = y$
  Siguin i,j índexs de TM tals que:
  $$\varphi_i = \varphi_j \wedge i \in L \Rightarrow \exists y: \varphi_i(y) = y$$
  $$\Rightarrow \exists y: \varphi_j(y) = y \Rightarrow j \in L$$

- $L = \{x \mid \mathrm{Dom}\,\varphi_x \cap \mathrm{Im}\,\varphi_x \neq \emptyset\}$

siguin i, j índexs de TM tals que:

$$\varphi_i = \varphi_j \wedge i \in L \Rightarrow \begin{cases} \mathrm{Dom}\,\varphi_i = \mathrm{Dom}\,\varphi_j \\ \mathrm{Im}\,\varphi_i = \mathrm{Im}\,\varphi_j \end{cases}$$

$\Rightarrow \mathrm{Dom}\,\varphi_i \cap \mathrm{Im}\,\varphi_i \neq \emptyset$
$\Rightarrow \mathrm{Dom}\,\varphi_j \cap \mathrm{Im}\,\varphi_j \neq \emptyset$
$\Rightarrow j \in L$

- $L = \{x \mid \exists y, z : |y| = |z| = 10 \wedge y \in L \wedge z \in L\}$

siguin i, j índexs de TM tals que:

$\boxed{\begin{matrix}\varphi_i = \varphi_j \\ L_i = L_j\end{matrix}} \wedge i \in L \Rightarrow \exists y, z : |y| = |z| = 10 \wedge y \in L_i \wedge z \in L_i$
$\Rightarrow \exists y, z : |y| = |z| = 10 \wedge y \in L_j \wedge z \in L_j$
$\Rightarrow j \in L$

Teorema de Rice:

sigui L un conjun d'índexs, llavors es compleix que:

$$L \in REC \Leftrightarrow L = \emptyset \vee L = \mathbb{N}$$

un conj d'índexs és recursiu si o bé es buit o bé són tots els naturals.

$$\boxed{L \neq \emptyset \wedge L \neq \mathbb{N} \Rightarrow L \notin REC}$$

$L \neq \emptyset$ Trobar una TM que compleixi la propietat $x \in L$
$L \neq \mathbb{N}$ Trobar-ne una que no la compleixi $\rightarrow x \notin L$

Corol·lari de Rice:

sigui L un conjunt d'índexs no recursiu ($L \notin REC$)
si existeix índex x de TM tal que $\mathrm{Dom}\,\varphi_x = \emptyset$ } si $x \in L \Rightarrow L \notin ER$

funció buida
lleng. buit

= Exemples:

- $L = \{x \mid \mathrm{Dom}\,\varphi_x \neq \emptyset\}$

($L \neq \emptyset$) $\Rightarrow$ sigui x índex de TM tq $\forall y\ \varphi_x(y) = y \Rightarrow \mathrm{Dom}\,\varphi_x \neq \emptyset = \Sigma^*$
funció identitat $\Rightarrow x \in L \Rightarrow L \neq \emptyset$

($L \neq \mathbb{N}$) $\Rightarrow$ sigui x índex de TM tq $\forall y\ \varphi_x(y)\uparrow \Rightarrow \mathrm{Dom}\,\varphi_x = \emptyset \neq \Sigma^*$
funció buida $\Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

T.Rice
$\Downarrow$
$L \notin REC$

① $L = \{x \mid Dom\ \varphi_x \cap K \neq \emptyset\}$ és ER $\notin$ REC

- L és conjunt d'índexs perquè:

$$\forall i,j : \varphi_i = \varphi_j \wedge i \in L \Rightarrow Dom\ \varphi_i = Dom\ \varphi_j$$
$$\Rightarrow Dom\ \varphi_i \cap K \neq 0$$
$$\Rightarrow Dom\ \varphi_j \cap K \neq 0$$
$$\Rightarrow j \in L$$

- Apliquem T. RICE:

$\boxed{L \neq \emptyset}$: sigui x índex de TM tal que $Dom\ \varphi_x = \Sigma^* \Rightarrow \Sigma^* \cap K \neq \emptyset \Rightarrow x \in L \Rightarrow L \neq \emptyset$

$\boxed{L \neq \mathbb{N}}$: sigui x índex de TM tal que $Dom\ \varphi_x = \emptyset \Rightarrow \emptyset \cap K = \emptyset \Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$\overset{TR}{\Rightarrow} L \notin$ REC. / Demostrat $L \in$ ER pel T. Proj

② $L = \{x \mid \exists n \in \mathbb{N} : \|L_x\| = n\}$ $\notin$ REC $\wedge \notin$ ER
→ TM que reconeix un lleng. finit

- L és conjunt d'índexs perquè:

$\forall i,j$ índexs de TM tq. $\varphi_i = \varphi_j$ ($L_i = L_j$) $\wedge\ i \in L \Rightarrow \exists n \in \mathbb{N} : \|L_i\| = n$
$\Rightarrow \exists n \in \mathbb{N} : \|L_j\| = n$
$\Rightarrow j \in L$

- Apliquem el T RICE:

$\boxed{L \neq \emptyset}$: sigui x índex de TM
si $L_x = \emptyset \Rightarrow \|L_x\| = 0 \in \mathbb{N} \Rightarrow x \in L \Rightarrow L \neq \emptyset$

$\boxed{L \neq \mathbb{N}}$: sigui x índex de TM tq $L_x = \Sigma^* \Rightarrow \forall n \in \mathbb{N} : \|L_x\| > n \Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$\overset{TR}{\Rightarrow}$ **L $\notin$ REC**

! Corol·lari RICE: $Dom\ \varphi_x = \emptyset \Rightarrow L_x = \emptyset \Rightarrow \ldots\ldots \Rightarrow x \in L \overset{CR}{\Rightarrow}$ **L $\notin$ ER**

③ • $L = \{x \mid Dom\ \varphi_x \cap Im\ \varphi_x \neq \emptyset\} \notin$ REC

- L és cnj índex demostrat a la pàg anterior.
- Apliquem el T. RICE:

$\boxed{L \neq 0}$: sigui x un índex de TM tal que $\varphi_x(y) = y \Rightarrow$
$Dom\ \varphi_x = \Sigma^* \wedge Im\ \varphi_x = \Sigma^* \Rightarrow Dom\ \varphi_x \cap Im\ \varphi_x \neq 0 \Rightarrow$
$\Rightarrow x \in L \Rightarrow L \neq 0$

$\boxed{L \neq \mathbb{N}}$: sigui x índex de TM tal que $\varphi_x(y)\uparrow\ \forall y \Rightarrow$
$\Rightarrow (Dom\ \varphi_x = \emptyset \wedge Im\ \varphi = \emptyset) = \emptyset \Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$\overset{TR}{\Rightarrow} L \notin$ REC

④ $L = \{ x \mid \exists y : \varphi_x(y) = y \} \notin REC$

- L és CÍndexs
- Apliquem TRice:

$\boxed{L \neq \emptyset}$: Sigui x índex de TM tq $\varphi_x(y) = y \ (\forall y)$ $\Rightarrow$ (funció identitat)

$\Rightarrow \forall y : \varphi_x(y) = y \Rightarrow \exists y : \varphi_x(y) = y \Rightarrow x \in L \Rightarrow L \neq \emptyset$

$\boxed{L \neq \mathbb{N}}$: Sigui x índex de TM tq $\forall y \ \varphi_x(y)\uparrow \Rightarrow$ $Dom\ \varphi_x = \emptyset \Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$\xRightarrow{TR} L \notin REC$

!* — Classificació REC, ER —

T. Projecció	$\Rightarrow$	$L \in ER$		
T. Rice	$\Rightarrow$	$L \notin REC$	$\Rightarrow$ C. Rice $\Rightarrow$	$L \notin ER$
Reduccions	$\Rightarrow$	$L \notin REC$, $L \notin ER$		

⑤. $L = \{ x \mid \forall y : |\varphi_x(y)| \leq |y| \}$ $\leftarrow$ amb un $\forall$ mai T.Proj

- L és cnj índexs $\varphi_i = \varphi_j \wedge i \in L \Rightarrow \forall y : |\varphi_i(y)| \leq |y| \Rightarrow \forall y : |\varphi_j(y)| \leq |y| \Rightarrow j \in L$
- Apliquem T Rice:

Essent L un CÍndexs:

$L \neq \emptyset \wedge L \neq \mathbb{N} \Rightarrow L \notin REC$

$\boxed{L \neq \emptyset}$: Sigui x índex de TM tq $\forall y : \varphi_x(y) = y$

$\Rightarrow \forall y : |\varphi_x(y)| = |y| \Rightarrow$

$\Rightarrow \forall y : |\varphi_x(y)| \leq |y| \Rightarrow x \in L \Rightarrow L \neq \emptyset$

$\boxed{L \neq \mathbb{N}}$: Sigui x índex de TM tal que $\varphi_x(y) = y \cdot y$

$\Rightarrow \exists y : |\varphi_x(y)| > |y| \Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$\xRightarrow{TR} L \notin REC$

Intentem aplicar el corol·lari del T Rice:

L és ci no rec $\Rightarrow$

Sigui x un índex de TM tal que $Dom\ \varphi_x = \emptyset$

$Dom\ \varphi_x = \emptyset \Rightarrow \forall y : |\varphi_x(y)| < |y| \Rightarrow x \in L \xRightarrow{CR} L \notin ER$

1. Donades dues TM, M i N i un enter $n>0$ determinar si $\|Dom\ f_M \cap Dom\ f_N\| \geq n$

entrades x les quals M s'atura

a. Formalitzar amb un llenguatge C.
b. $C \in ER$
c. $C \notin REC$

num codi dif
M_x, M_y

a). $C = \{\langle M, N, n\rangle \mid \|Dom\ f_M \cap Dom\ f_N\| \geq n\}$

$\equiv \{\langle M, N, n\rangle \mid \exists x_1 \ldots x_n$ (diferents) $: \forall i: 1 \leq i \leq n : \underbrace{M(x_i)\downarrow}_{HALT} \wedge N(x_i)\downarrow\}$ no rec

temps perquè l'entrada x_i s'aturi amb la màq. M

b) $\Rightarrow$ $\equiv \{\langle M, N, n\rangle \mid \exists x_1 \ldots x_n, t_1^M \ldots t_n^M, t_1^N \ldots t_n^N : \forall i: 1 \leq i \leq n : M(x_i)\downarrow$ en t_i^M passos $\wedge$ $N(x_i)\downarrow$ en t_i^N passos $\}$

$\downarrow$
REC $\Rightarrow C \in ER$.

c) Fem una reducció de K al llenguatge.

$K \leq_m L$

$f(\underline{x}) = (M, N, n)$

volem saber si $\in K$ — si $x \in K \rightarrow$ compleixen la propietat del llenguatge

$\left[\begin{array}{l} x \in K \Rightarrow x \in Dom\ \varphi_x \\ x \notin K \Rightarrow x \notin Dom\ \varphi_x \end{array}\right.$

$f(x) = (M_x, N, \underline{1})$

$\downarrow$ només mirem si comparteixen x o no.

per cada x d'entrada construir TM $\rightarrow$

N: entrada y
  si $y \neq x$ llavors
    mentre cert fer
    fmentre
  fsi

$\uparrow$ només s'atura amb entrada x

$Dom\ \varphi_N = \{x\}$ $\Rightarrow$ Tindrem 2 casos, 1 amb intersecció nula i l'altre no

$x \in K \Rightarrow M_x(x)\downarrow \Rightarrow \left.\begin{array}{l} x \in Dom\ \varphi_x \\ x \in Dom\ \varphi_N \end{array}\right\}$ $\|Dom\ \varphi_x \cap Dom\ \varphi_N\| \geq 1 \Rightarrow (M_x, N, 1) \in C$

totes dues tenen x

$x \notin K \Rightarrow M_x(x)\uparrow \Rightarrow \left.\begin{array}{l} x \notin Dom\ \varphi_x \\ \wedge\ Dom\ \varphi_N = \{x\} \end{array}\right\} \Rightarrow$ intersecció buida $\|Dom\ \varphi_x \cap Dom\ \varphi_N\| = 0 \Rightarrow (M_x, N, 1) \notin C$

$\boxed{x \in K \Leftrightarrow (M_x, N, n) \in C}$

f redueix K a C i és total i calculable

$K \notin REC \Rightarrow C \notin REC$

Dem f és calculable

F: "Entrada x
  si $y \neq x \longrightarrow$ mentre cert fer
  (x constant) fmentre
  fsi"
  escriure$(M_x, n, 1)$

$L = \{x \mid \exists y: |y| = \dot{2} \land y \in L_x\}$ $\notin$ REC

$K \leq_m L$ $\xleftarrow{\text{núm. Gödel}}$

$f(x) = z$

$x \in K \iff z \in L$

M_z: Entrada y
  simular $M_x(x)$
  Acceptar

x és una constant. serà diferent depenent de la x que entrem a la reducció

mentre cert fer (mentre si $x \notin K$)

inclou les $|y| = \dot{2}$

$x \in K \Rightarrow M_x(x)\downarrow \Rightarrow \forall y: M_z(y)\downarrow$ i accepta
$\Rightarrow \exists t: |t| = \dot{2}$ i $M_z(t)\downarrow$ i accepta $\Rightarrow z \in L$

$x \notin K \Rightarrow M_x(x)\uparrow \Rightarrow \forall y\ M_z(y)\uparrow \Rightarrow z \notin L$

redueix K a L
f és total i calculable $\Big\} \Rightarrow K \notin REC \Rightarrow L \notin REC$

$L = \{x \mid Dom\ \varphi_x \cap Im\ \varphi_x \neq \emptyset\}$ $\notin$ REC

HALT $\leq_m L$

$f(x,y) = z$

$x \in K \iff z \in L$

M_z: Entrada t
  simular $M_x(y)$
  escriure t

$(x,y) \in HALT \Rightarrow M_x(y)\downarrow \Rightarrow \forall t: M_z(t)\downarrow \land M_z(t) = t$
$\Rightarrow Dom\ \varphi_z = \Sigma^* \land Im\ \varphi_z = \Sigma^*$
$\Rightarrow Dom\ \varphi_z \cap Im\ \varphi_z \neq \emptyset \Rightarrow z \in L$

$(x,y) \notin HALT \Rightarrow M_x(y)\uparrow \Rightarrow \forall t: M_z(t)\uparrow$
$\Rightarrow Dom\ \varphi_z = \emptyset$
$\Rightarrow Dom\ \varphi_z \cap Im\ \varphi_z = \emptyset \Rightarrow z \notin L$

f és total i calculable
$x \in K \iff z \in L$
f redueix halt a L
$\Rightarrow$ HALT $\notin$ REC $\Rightarrow L \notin$ REC

- $L = \{x \mid \varphi_x$ és la funció identitat$\}$ $\notin$ ER

$\overline{K} \leq_m L$

$f(x) = z$

$x \in \overline{K} \iff z \in L$

M_z: entrada t
  simulem t passos de $M_x(x)$
  si $M_x(x)\downarrow$ en t passos llavors
    escriure $(t+1)$
  sino
    escriure (t)

$\overline{K} \notin ER$
$\Downarrow$
$L \notin ER$ $\Leftarrow$

$x \in \overline{K} \Rightarrow M_x(x)\uparrow \Rightarrow \forall t: M_x(x)\uparrow$ en t passos
$\Rightarrow \forall t: M_z(t) = t$
$\Rightarrow \varphi_z$ és la f identitat
$\Rightarrow z \in L$

$x \notin \overline{K} \Rightarrow M_x(x)\downarrow \Rightarrow \exists t: M_x(x)\downarrow$ en t passos $\Rightarrow \exists t: M_z(t) = t+1 \Rightarrow \varphi_z$ no és la funció ident
$\Rightarrow z \notin L$

(i) $L = \{x \mid L_x \neq \Sigma^*\} \notin E.R$

(ii) $L = \{x \mid \forall y: \varphi_x(y) = 2 * y\} \notin E.R$

(i) $L = \{x \mid L_x \neq \Sigma^*\}$

$$\boxed{\overline{K} \leq_m L \equiv K \leq_m \overline{L}}$$

$\overline{L} = \{x \mid L_x = \Sigma^*\}$

M_z: Entrada y
simular M_x
Acceptar

si $M_x \uparrow$ buit — Entra y while(1){}
si $M_x \downarrow$ $\Sigma^* L$ → Ent y acceptar

$\overline{K} \leq_m L$

$f(x) = z$

$x \in \overline{K} \Leftrightarrow z \in L$

$(x \notin K) \Leftrightarrow x \in \overline{K} \Rightarrow M_x(x)\uparrow \Rightarrow \forall y: M_z(y)\uparrow$
$\Rightarrow L_z = \phi \neq \Sigma^* \Rightarrow z \in L$

$\Leftrightarrow x \notin \overline{K} \Rightarrow M_x(x)\downarrow \Rightarrow \forall y\, M_z(y)\downarrow$ i accepta $\Rightarrow Dom\,\varphi_z = \Sigma^*$
$\Rightarrow L_z = \Sigma^* \Rightarrow z \in L$

(ii) $\overline{K} \leq_m L$

$f(x) = z$

$x \in \overline{K} \Leftrightarrow z \in L$

M_z: Entrada t
simulem $M_x(x)$ en t passos
si $M_x(x)\downarrow$ en t passos
escriure(0) / escriure($t * 3$)
sino
escriure($2 * t$)

$x \in \overline{K} \Rightarrow M_x(x)\uparrow \Rightarrow \forall t: M_x(x)\uparrow$ en t passos
$\Rightarrow \forall t: \varphi_z(t) = t * 2 \Rightarrow z \in L$

$x \notin \overline{K} \Rightarrow M_x(x)\downarrow \Rightarrow \exists t: M_x(x)\downarrow$ en t passos
$\Rightarrow \exists t: \varphi_z(t) = t * 3 \Rightarrow z \notin L$

* Donada una TM determinar si calcula la funció identitat

i) Formalitzar
ii) Reducció $K \leq_m L$
iii) Aplicar el T. Rice
iv) Reducció $\overline{K} \leq_m L$

i) $L = \{x \mid \forall y: \varphi_x(y) = y\}$

ii) $K \leq_m L$

$f(x) = z$ — num Gödel $M^x = M_z$

M_z: Entrada y
simular $M_x(x)$
Sortida y

$x \in K \Leftrightarrow z \in L$
$x \in K \Leftrightarrow f(x) \in L$

- $x \in K \Rightarrow M_x(x)\downarrow \Rightarrow \forall y: M_z(y)\downarrow \wedge M_z(y) = y \Rightarrow$
  $\Rightarrow \forall y: \varphi_z(y) = y \Rightarrow z \in L$

- $x \notin K \Rightarrow M_x(x)\uparrow \Rightarrow \forall y: M_z(y)\uparrow \Rightarrow \varphi_z$ no és la funció identitat $\Rightarrow z \notin L$

f és total (definida x totes les entrades), calculable
i redueix K a L $\underset{K \notin REC}{\Longrightarrow} L \notin REC$

iii) • L és un conjunt d'índexs
siguin i, j índexs de TM tq:
$\forall i,j: \varphi_i = \varphi_j \wedge i \in L \Rightarrow \forall y: \varphi_i(y) = y$
$\Rightarrow \forall y: \varphi_j(y) = y$
$\Rightarrow j \in L$

• Apliquem ara el Teorema:
sigui L ci si $L \neq \emptyset \wedge L \neq \mathbb{N} \Rightarrow L \notin REC$

$\boxed{L \neq \emptyset}$: sigui x índex de TM tq $\forall y\ \varphi_x(y) = y$
$\Rightarrow \forall y\ \varphi_x(y) = y \Rightarrow x \in L \Rightarrow L \neq \emptyset$

$\boxed{L \neq \mathbb{N}}$: sigui x índex de TM tq $\varphi_x(y) = 0\ (\forall y)$
$\varphi_x(y) = y+1$ (fun successor)

$\Rightarrow x \notin L \Rightarrow L \neq \mathbb{N}$

$L \neq 0 \wedge L \neq \mathbb{N} \Rightarrow L \notin REC$

iv) $\overline{K} \leq_m L$

$x \in \overline{K} \Leftrightarrow f(x) \in L$, $f(x) = z$

M_z: Entrada t
Simular t passos de $M_x(x)$
si $M_x(x)\downarrow$ en t passos $\rightarrow$ escriure $t+1$
sino escriure t

$(x \notin K) \Leftrightarrow x \in \overline{K} \Rightarrow M_x(x)\uparrow \Rightarrow \forall t: M_x(x)\uparrow$ en t passos $\Rightarrow \forall t: M_z(t) = t \Rightarrow \varphi_z$ és l'ident $\Rightarrow z \in L$

$(x \in K) \Leftrightarrow x \notin \overline{K} \Rightarrow M_x(x)\downarrow \Rightarrow \exists t: M_x(x)\downarrow$ en t passos $\rightarrow \exists t: M_z(t) = t+1 \Rightarrow \varphi_z$ no és l'ident $\Rightarrow z \notin L$

* Donada una TM determinar si reconeix el llenguatge K

i) Formalitzar
ii) Reducció HALT $\leq_m L$
iii) T. Rice.

i) $L = \{ x \mid L_x = K \}$

ii) HALT $\leq_M L$

$(x,y) \in HALT \Leftrightarrow f(x,y) \in L$

M_z : Entrada w
Simular $M_x(y)$
simular $M_w(w)$
Acceptar

$(x,y) \in HALT \Rightarrow M_x(y)\downarrow \Rightarrow$

17 Juny 05.

Normalització CFG's

1 • Eliminació de produccions nul·les
2 • Eliminació de símbols no fecunds
3 • Eliminació de símbols no accessibles.

A és anul·lable si $A \overset{*}{\Rightarrow} \lambda$

$G: S \to aXbY \mid aYbX$
$X \to \ldots \quad \lambda$
$Y \to \ldots \quad \lambda$

$S' \to S \mid \lambda$
$S \to aXbY \mid abY \mid aXb \mid ab \mid aYbX \mid aYb \mid abX \mid$

Exemple:

$S \to aXbS \mid bYaS \mid \lambda$
$X \to aXbX$
$Y \to bYaY$

① $S' \to S \mid \lambda$
$S \to aXbS \mid aXb \mid abS \mid ab \mid bYaS \mid bYa \mid baS \mid ba$
$X \to aXbX \mid aXb \mid abX \mid ab$
$Y \to bYaY \mid bYa \mid baY \mid ba$

② X és útil si existeix $w \in \Sigma^*$ tal que
$\underbrace{S \overset{*}{\Rightarrow} \alpha X \beta}_{\text{Accessible}} \underbrace{\overset{*}{\Rightarrow} w}_{\text{fecund}}$

Expressions Regulars

Λ és una expressió regular (λ)
$\forall a \in \Sigma$: a és una expressió regular

si a i b són expressions regulars

$(a+b)$, ab, a^*, a^+ són exp. regulars

Passar d'autòmata a expressió regular:

$X = AX + B$ eq. lineal per la dreta
$X = XA + B$ eq. lineal per l'esquerra

si L és solució de l'equació $\Rightarrow L = AL + B$

Lema d'Arden — Eq. lineals x la dreta.

1. El llenguatge A^*B és solució de l'equació
2. Qualsevol solució de l'eq. $X = AX + B$ conté A^*B ($A^*B \subseteq L$)
3. Si $\lambda \notin A$, llavors A^*B és l'única solució.

Eq. lineals x l'esquerra $\Rightarrow BA^*$

Demostració: $X = AX + B$ substituïm

$A^*B = A \cdot A^*B \cup B$

$A \cdot A^*B \cup B = A^+B \cup B = (A^+ \cup \{\lambda\}) \cdot B = A^*B$

$M = (Q, \Sigma, \delta, I, F)$

EQ. ESQUERRA

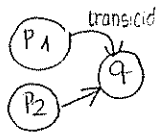
Arribem al mot des de l'estat inicial a q

- $L(q) = \{ w \in \Sigma^* \mid \exists p \in I : q = p \cdot w \}$

Transicions d'entrada a q.

- $L(q) = \bigcup_{p,a:\ q = p \cdot a} L(p)$

Parlem d'antecessors (p_i)



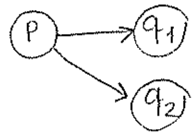
EQ. DRETA.

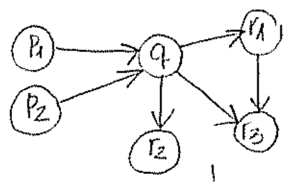
Desde p s'arriba al final

- $L(p) = \{ w \in \Sigma^* \mid \exists q \in F : p \cdot w = q \}$

- $L(p) = \bigcup_{q,a:\ p \cdot a = q} L(q)$

Parlem de successors (r_i)

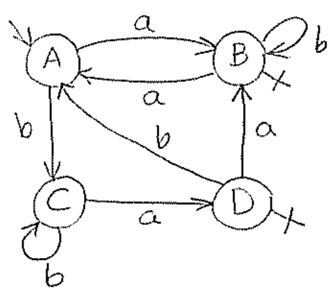




- $L(M) = \bigcup_{q \in F} L(q) \rightarrow$ Unió dels llenguatges dels estats finals

$L(M) = \bigcup_{q \in I} L(q) \rightarrow$ Està fet per NFA per això hi ha unió. Si fos DFA només n'hi hauria un.

EXEMPLE:



ESQ.

a l'inicial

$A = Ba + Db + \lambda$

$B = Bb + Aa + Da$

$C = Cb + Ab$

$D = Ca$

DRET

$A = aB + bC$

$B = aA + bB + \lambda$

$C = aD + bC$

$D = aB + bA + \lambda$

En els estats finals

ESQ

$L(M) = \underset{qF}{B + D}$

DRET

$L(M) = \underset{q_0}{A}$

! Trobem l'expressió regular per cadascún dels estats

ESQ: $L = LA + B \qquad BA^*$

$B = Aa + Bb + Da$

$B = \underset{L}{B}\underset{A}{b} + \underset{B}{Aa + Da}$

$\boxed{B = (Aa + Da)\, b^*}$

$C = \underset{L}{C}\underset{A}{b} + \underset{B}{Ab}$

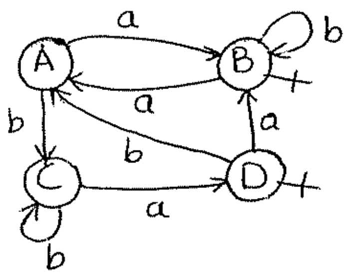
$\boxed{C = (Ab)\, b^*}$

$D = Abb^*a \qquad D = \underset{B}{Ca}$

$A = Ba + Db + \lambda = ((Aa + Da)b^*)a + (Abb^*a)b + \lambda =$

$= Aab^*a + Abb^*aab^*a + Abb^*ab + \lambda$

$L = \ L \ A \ + \ D \ B$



* Equacions lineals per l'esquerra

$A = Ba + Db + \Lambda$ ↙ als símbols inicials !

$B = Bb + Aa + Da$

$C = Cb + Ab$

$D = Ca$

$L(M) \underset{qf}{=} B + D$

Segons el lema d'arden una solució de l'equació és tot llenguatge que compleix $L = LA + B \longrightarrow$ solució $= BA^*$

Resolem el sistema :

$B = Aa + Bb + Da$

$\underset{L}{B} = \underset{L}{B}\underset{A}{b} + \underbrace{Aa + Da}_{B}$

$B = (Aa + Da)b^*$

$B = (Aa + Abb^*ca)b^*) = A(a + bb^*a)b^* = A(bb^*aab^* + ab^*)$

$D = Ca$

$D = (Ab)b^*a$

$\underset{L}{C} = \underset{L}{C}\underset{A}{b} + \underset{B}{Ab}$

$C = (Ab)\cdot b^*$

A

$L(M) = B + D = A(bb^*aab^* + ab^*) + Abb^*a + \Lambda$!

Resolent pel lema d'Arden :

$A = Ba + Db + \Lambda = A(ab^* + bb^*aab^*)a + Abb^*ab =$

$= \underset{B}{\Lambda} + \underset{L}{A}\underbrace{(ab^*a + bb^*aab^*a + bb^*ab)}_{A}$

Pel lema d'Arden : $A = (ab^*a + bb^*aab^*a + bb^*ab)^*$

$= (ab^*a + bb^*aab^*a + bb^*ab)^*(bb^*aab^* + ab^* + bb^*a)$

* Equacions lineals per la Dreta. (transició de sortida) $X = AX + B \quad \rightarrow A^*B$

$A = aB + bC$ ↙ estats finals !

$B = aA + bB + \Lambda$

$C = aD + bC$

$D = aB + bA + \Lambda$

A

$\underset{L}{B} = \underset{A}{b}\underset{L}{B} + \underbrace{aA + \Lambda}_{B}$

$\boxed{B = b^*(aA + \Lambda)}$

$C = aD + bC$

$\underset{L}{C} = \underset{A}{b}\underset{L}{C} + \underbrace{a ab^*(aA + \Lambda) + bA + \Lambda}_{B}$

$D = aB + bA + \Lambda$

$\boxed{D = ab^*(aA + \Lambda) + bA + \Lambda}$

$= ab^*aA + ab^* + bA + \Lambda$

$C = b^*(aab^*(aA + \Lambda) + bA + \Lambda)$

$C = bC + aD$

$C = bC + aab^*aA + ab^* + bA + \Lambda$

$L(M) = A = aB + bC =$

$= ab^*(aA + \Lambda) + b$

b^*

```
/* Copia  l          En cada iteració copiem els 4 bytes del task_union que
té el pr
void cop               a(int origen, int desti)
{
  int i;
  for(i =              i++) task[desti].stack[i] = task[origen].stack[i];
}


/* Task_               que permet efectuar el canvi de context. */
void tas               task_union* t)
{
   /* Em               sició en el task del procés que està corrent. */
   int p                long)t - (unsigned long)task)>>12;

   /* Emm              eça de l'esp dins de la pila de sistema. Aquest registre
   es tr               ions per sobre de la part més baixa, la qual inclou
   regis
   DWord               d) &(t->stack[1024-16]);

   /* Ac               S perquè apunti a la pila de sistema. */
   set_e

   /* A                e la taula de pàgines col·loquem l'adreça del marc de la
   pila                cés actual emmagatzemada en el corresponent camp del
   task_
   set_s               sk.marc);

    /* F              5 que permet invalidar les entrades de la TLB. Només cal
   reesc               e cr3. */
     set_

     /* A              stre esp l'adreça calculada anteriorment perquè realitzi
     el c               correctament. */
    __as                (

     "mo   %
     :
     :"g
    );

   /* Re                 res i retornem. */
   __asm
       "p
       "p  l
       "p  l
       "p
       "p
       "p
       "p
       "p  l
       "p
       "p  l
       "p  l
       "
   );
}
```

$$A = aB + bC$$
$$B = b^*(aA + \Lambda) = b^*aA + b^*$$
$$C = b^*aD$$
$$D = aB + bA + \Lambda$$

$$L(M) = A = aB + bC = a(b^*aA + b^*) + b(b^*aD) =$$
$$= ab^*aA + ab^* + bb^*aD = ab^*aA + ab^* + bb^*a(aB + bA + \Lambda) =$$
$$= ab^*aA + ab^* + bb^*a(ab^*aA + ab^* + bA + \Lambda) =$$
$$= ab^*a\underline{A} + ab^* + bb^*aab^*a\underline{A} + bb^*aab^* + bb^*ab\underline{A} + bb^*a =$$
$$= \underbrace{(ab^*a + bb^*aab^*a + bb^*ab)}_{A}\underbrace{A}_{L} + \underbrace{ab^* + bb^*aab^* + bb^*a}_{B}$$

$$\boxed{L(M) = A = A^*B = (ab^*a + bb^*aab^*a + bb^*ab)^*(ab^* + bb^*aab^* + bb^*a)}$$

NOM FITXER: ərrupt.c

ESTRUCTURA DADES AFEG

temps → Comptabilitza el temps transcorregut des del boot sistema.
tics → Comptabilitza el nombre de tics que produeix en el ːema.
tics = 1 seg.
quantum; → Comptabilitza el quantum (en tics) transcorregut ada procés.

FUNCIONS:

```
d setInterruptHandler(int vector, void (*handler)(), int
AccessibleFromPL);
 setTrapHandler(int vector, void (*handler)(), int
ccessibleFromPL);

 divide_error_routine();
 debug_routine();
 nmi_routine();
 breakpoint_routine();
 overflow_routine();
 bounds_check_routine();
 invalid_opcode_routine();
 device_not_avaiable_routine();
 double_fault_routine();
 coprocessor_segment_overrun_routine();
 invalid_tss_routine();
 segment_not_present_routine();
 stack_exception_routine();
 general_protection_routine();
 page_fault_routine();
 floatin_point_error_routine();
 alignment_check_routine();
 rellotge_routine();
 teclat_routine();
```

JUSTIFICACIC

Excepcio_routine()
rograma la rutina de servei per cada excepció.

Lotge_routine()
utina de rellotge conta el temps de cpu que cada procés corrent i a més duu a terme la planificació Round Robin.

A cada tic, incrementem el camp t_cpu de cada procés que es a en el task_struct per tal de donar informacions dístiques posteriorment, i el visualitzem amb el seu pid.

A cada tic incrementem el quantum que ha estat inicialitzat , en el cas que el quantum del procés hagi arribat al que té definit en el camp del task_struct "quantum", realitzem canvi de context cap al següent procés a executar nificació amb política Round Robin) havent retornat abans un eoi.

at_routine()
cada make de teclat vàlid (que sigui un char alfanumèric) magatzema al buffer i es mostra per pantalla sobre el otge.
hi ha algun procés bloquejat i, o bé tots els caràcters que na al buffer són útils o bé el buffer està ple procedim a la ura del buffer escrivint a la pila d'usuari del esponent procés (el primer de la keyboardqueue).
p finalitzada la lectura, passem el corresponent procés a ng i ens tornem a col·locar en la posició correcta de la lla.