



2 Теория на изчислимостта

2.1 Функции изчислими интуитивно и тезис на Чърч



Ефект

Най-сигурната рецепта да объркате един неинформатик:
Ожесточен дебат: кой е **най-добрият** език за
прогармиране

- ☐ Всички програмни езици и машинни модели са
"‘**еднакво** ’’ мощни
- ☐ Във всеки модел има едни и същи **не изчислими**
проблеми



2.2 Интуитивно понятие за изчислимост

$f : \mathbb{N}^k \rightarrow \mathbb{N}$ (частична) функция.

f е **изчислима**, ако

$\exists$ ефективна процедура (=алгоритъм), която **изчислява** f .

Ефективна процедура = Java-програма (, ..., “подходящ” програмен език)

Вход: $(x_1, \dots, x_k) \in \mathbb{N}^k$

Изход: $f(x_1, \dots, x_k)$

Програмата **завършва** за краен брой стъпки, ако

$(x_1, \dots, x_k) \in$ дефиниционната област на f

и **не завършва** иначе.



Пример

input n

repeat

until false

изчислява **никъде недефинираната функция** Ω



Пример

$$f_{\pi}(n) = \begin{cases} 1 & \text{ако } n \text{ е начален сегмент в десет. представяне на } \pi. \\ 0 & \text{иначе} \end{cases}$$

f_{π} е изчислима:



Някои апроксимации на π

Архимед: Апроксимации с правилни многоъгълници.

Древните индийци: 1682 преоткрито от Лайбниц

$$\pi = 4 \sum_{i=0}^{\infty} \frac{-1^i}{2i+1} = 1 - \frac{1}{3} + \frac{1}{5} - \dots$$

Baile-Borwein-Plouffe 1996:

$$\pi = \sum_{i=0}^{\infty} \frac{1}{16^i} \left(\frac{4}{8i-1} - \frac{2}{8i+4} - \frac{1}{8i+5} - \frac{1}{8i+6} \right)$$



Пример

$$f(n) = \begin{cases} 1 & \text{ако } n \times '7' \text{ участват някъде в десет. предст. на } \pi. \\ 0 & \text{иначе} \end{cases}$$

Дали f е изчислима ?



Пример

$$f(n) = \begin{cases} 1 & \text{ако } n \times '7' \text{ участват някъде в десет. предст. на } \pi. \\ 0 & \text{иначе} \end{cases}$$

f е **изчислима**

Ако $\forall n : n \times '7'$ се среща: $\forall n : f(n) = 1$

Ако 7 се среща максимум n_0 пъти:

$$\longrightarrow f(n) = \begin{cases} 1 & \text{ако } n \leq n_0 \\ 0 & \text{иначе} \end{cases}$$

Очевдно е изчислима, защото е константа, за всички n , с изключение на кр. брой!



Неизчислими функции

- ☐ Множеството от всички изчислими функции е изброимо
(тъй като програмите, които ги изчисляват са изброимо много-крайни низове от краен брой символи).
- ☐ Всички функции са неизброимо много



Тезис на Чърч-Тюринг

Изчислимите функции с машини на Тюринг са точно изчислимите в интуитивен смисъл. Не е тероема (не можем да го докажем), но всички го приемат.

Обосновка

- Всички познати изчислителни модели са еквивалентни.

това можем да го докажем

- Всички известни "интуитивно" изчислими функции са изчислими по Тюринг.



Детерминистични - машини на Тюринг (DMT)

$T = (Q, \Sigma, \Gamma, \delta, s, F)$:

□ Q състояния, Σ входна азбука

□ Γ азбука на лентата,

□ $\sqcup \notin \Sigma$: шпация,

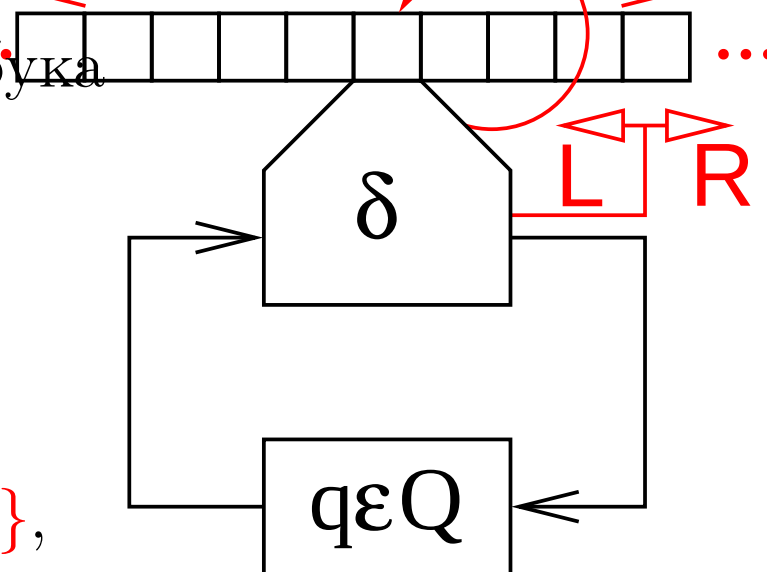
$\Sigma \cup \{\sqcup\} \subseteq \Gamma$

□ $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, N\}$,
функция на прехода;

□ $s \in Q$, начално състояние

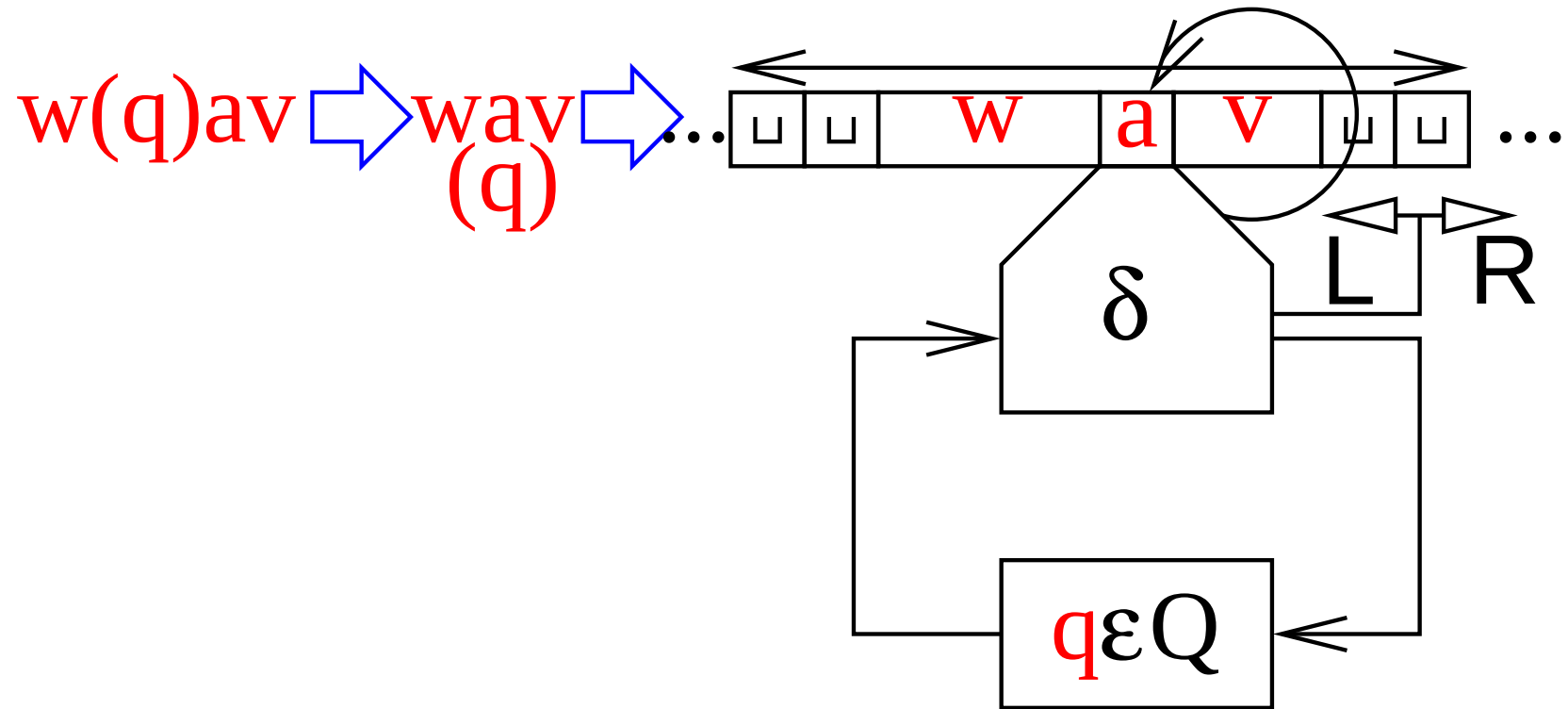
□ $F \subseteq Q$, крайни състояния

Eingabe $w \in \Sigma^*$ Ausgabe





Конфигурации на МТ



$$w, v \in \Gamma^*, a \in \Gamma, q \in Q$$



Как работи DMT

$$\begin{array}{ccc}
 wa(q)bcv & \delta(q,b)=(q',b',N) & \\
 & \vdash & wa(q')b'cv \\
 \\
 wa(q)bcv & \delta(q,b)=(q',b',L) & \\
 & \vdash & w(q')ab'cv \\
 \\
 wa(q)bcv & \delta(q,b)=(q',b',R) & \\
 & \vdash & wab'(q')cv
 \end{array}$$

Функцията на прехода δ дава:

- ☐ Ново състояние като КА
- ☐ Нов символ — замества текущия символ на лентата (който сочи главата)
- ☐ Посоки за придвижване на главата



Кога завършва една ДМТ?

T завършва при конфигурация $w(q)av$, ако $\delta(q, a) = (q, a, N)$.

Конвенция:

$$\forall q \in F : \forall a \in \Gamma : \delta(q, a) = (q, a, N)$$



Интерпретация с граф

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ дефинира

безкраен мулти граф

Възли: конфигурации на T .

Дъги: допустими от δ преходи между конфигурации.

$w \in L(A) \Leftrightarrow \exists$ път $P = (s)w \rightarrow \dots \rightarrow u(f)v : f \in F$



Машините на Тюринг като разпознаватели

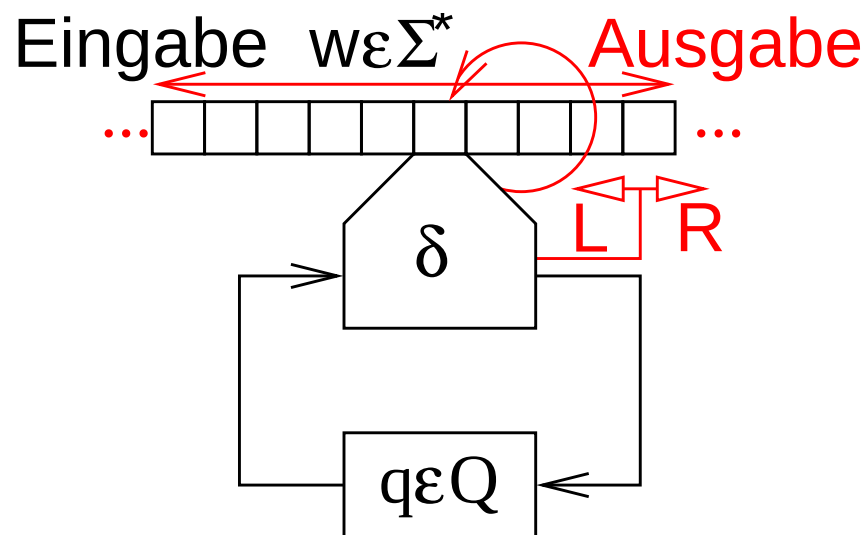
$$T = (Q, \Sigma, \Gamma, \delta, s, F).$$

$L(T)$?

T приема (разпознава) $w \Leftrightarrow$

$$\exists \alpha, \beta \in \Gamma^*, f \in F : (s)w \vdash^* \alpha(f)\beta$$

$$L(T) := \{w \in \Sigma^* : T \text{ приема } w\}.$$





Интерпретация с граф

$$T = (Q, \Sigma, \Gamma, \delta, s, F).$$

$$L(T)?$$

Дефиниция:

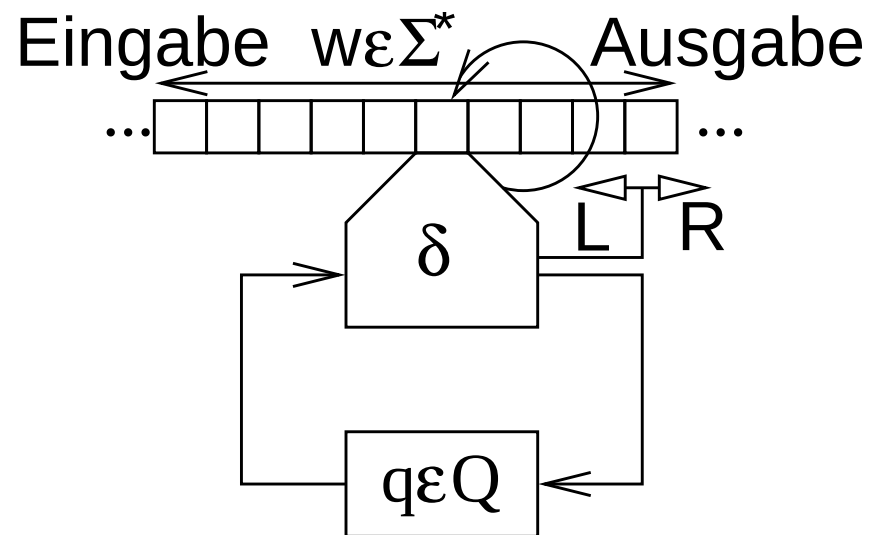
T приема $w \in \Sigma^*$, ако

$\exists$ редица от (**допустими** от δ)

преходи между конфигурации

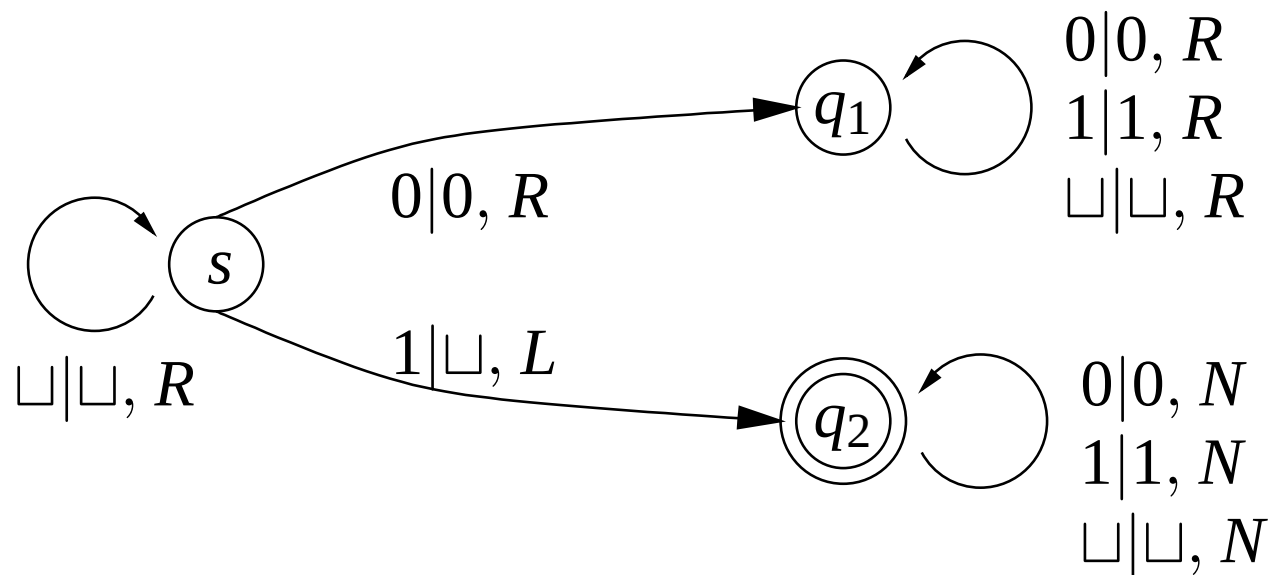
$(s)w \rightarrow \dots \rightarrow x(f)y$ с $f \in F$.

$$L(T) := \{w \in \Sigma^* : T \text{ приема } w\}.$$





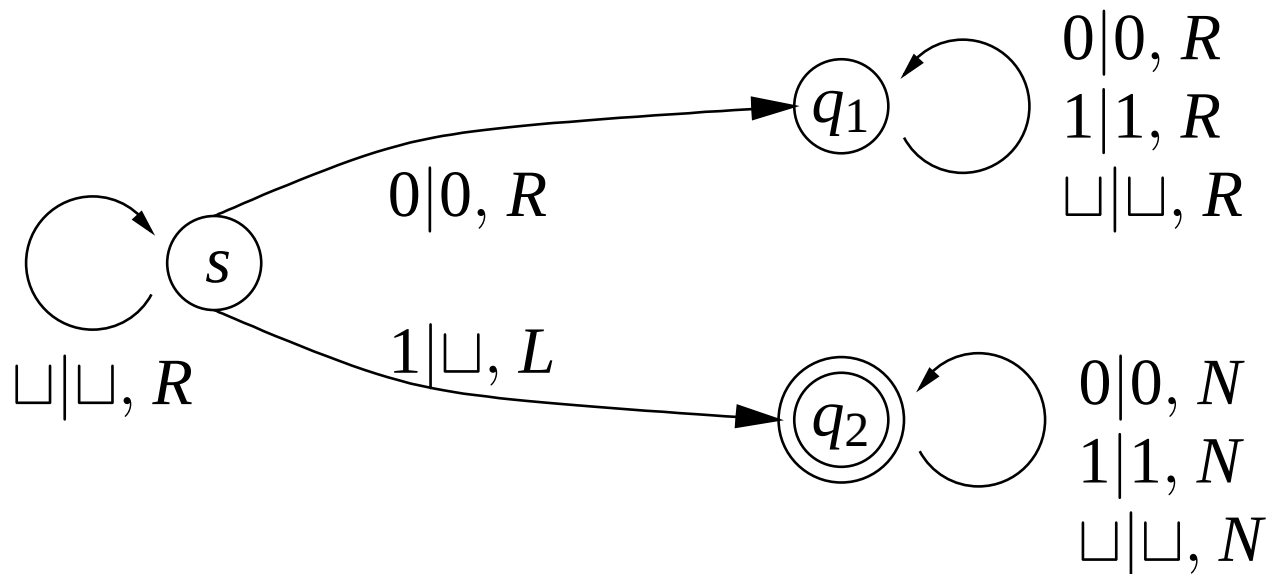
Пример: разпознавател за $L = 1(0,1)^*$



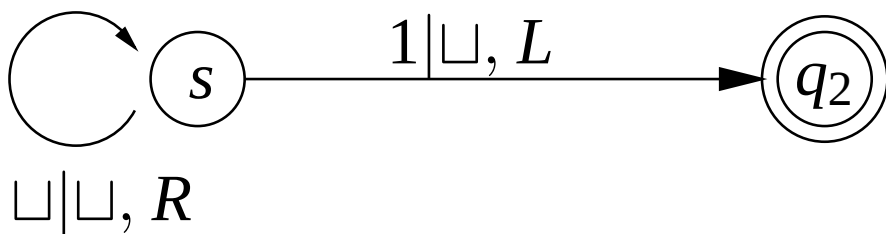
Графични означения:

Разширение на КА. Разширено маркиране на дъгите:

входни символи | изходни символи, L, N, R

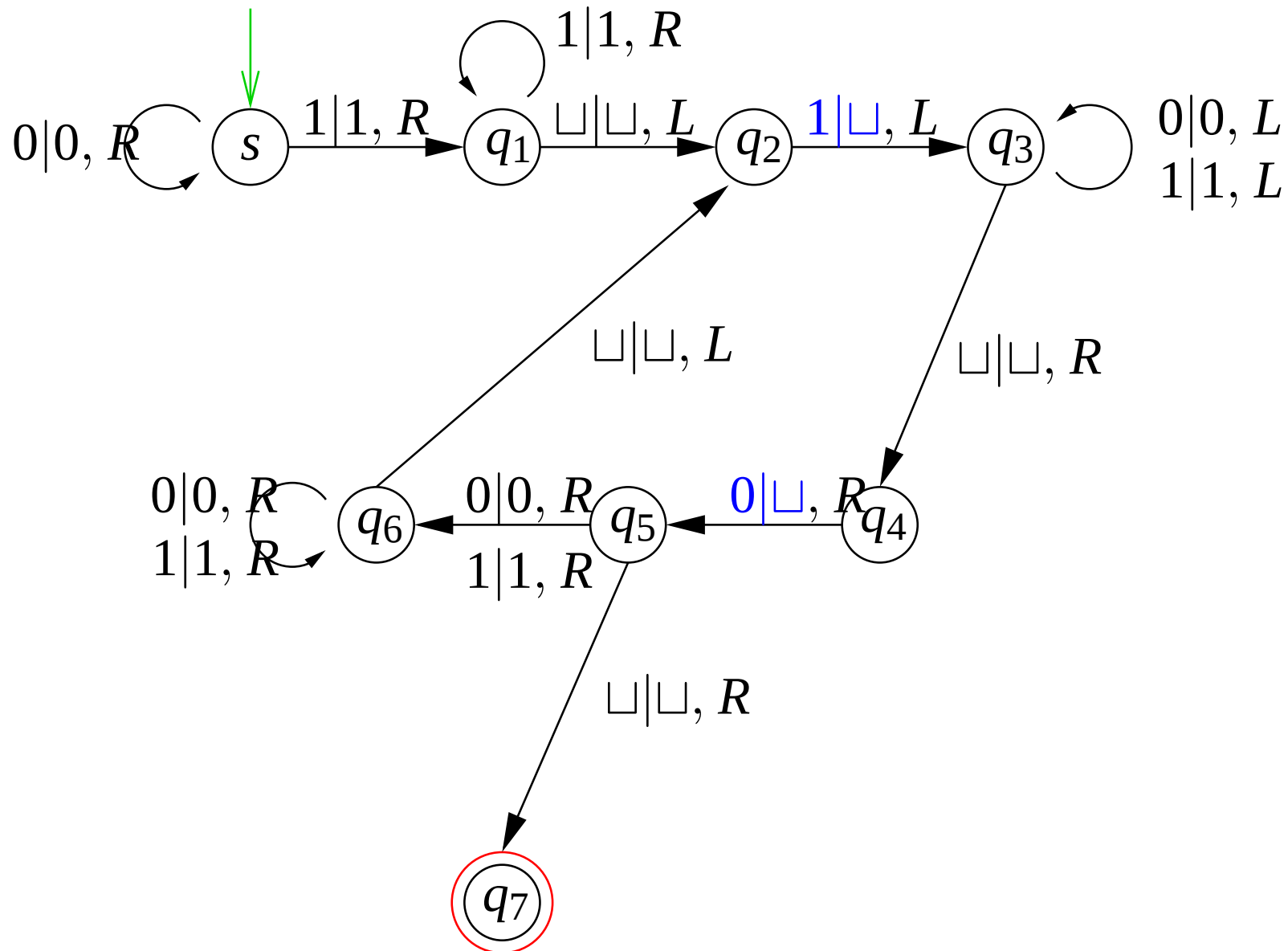


Конвенция: Завършващите с Error преходи $\rightsquigarrow$ MT завършва.



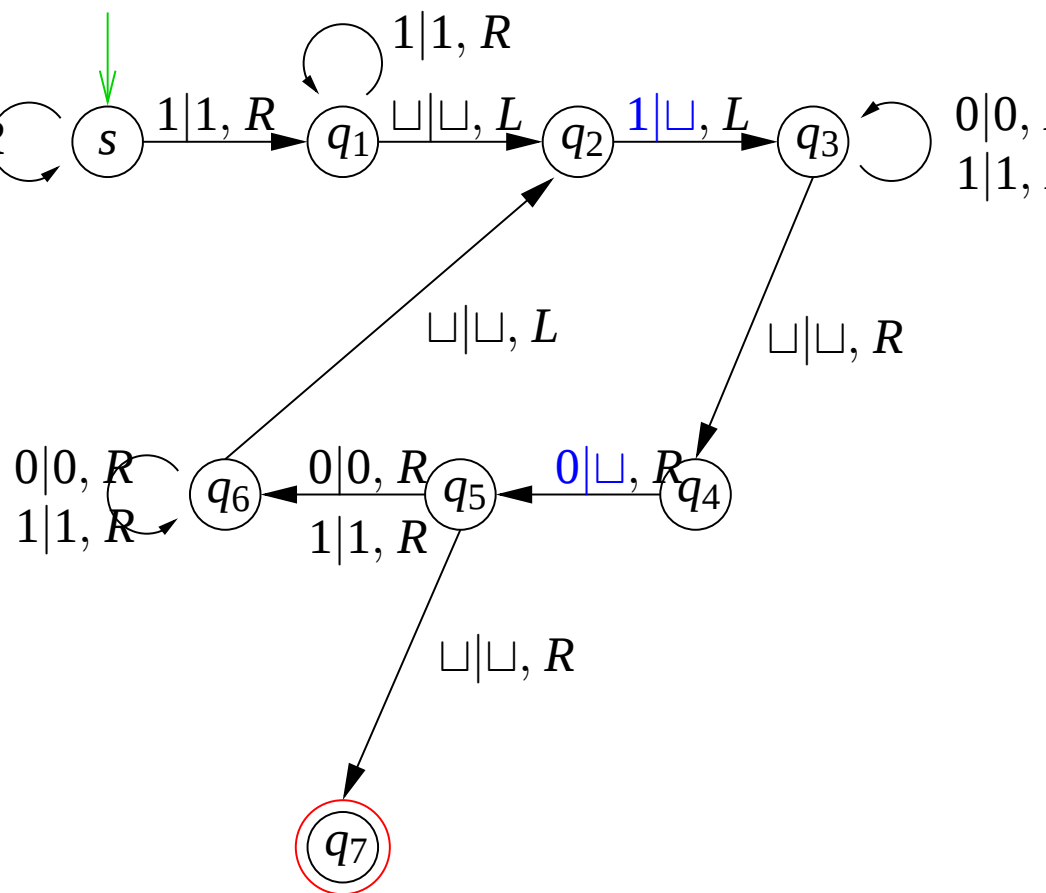


Пример: разпознаватели за $\{0^n 1^n : n \geq 1\}$



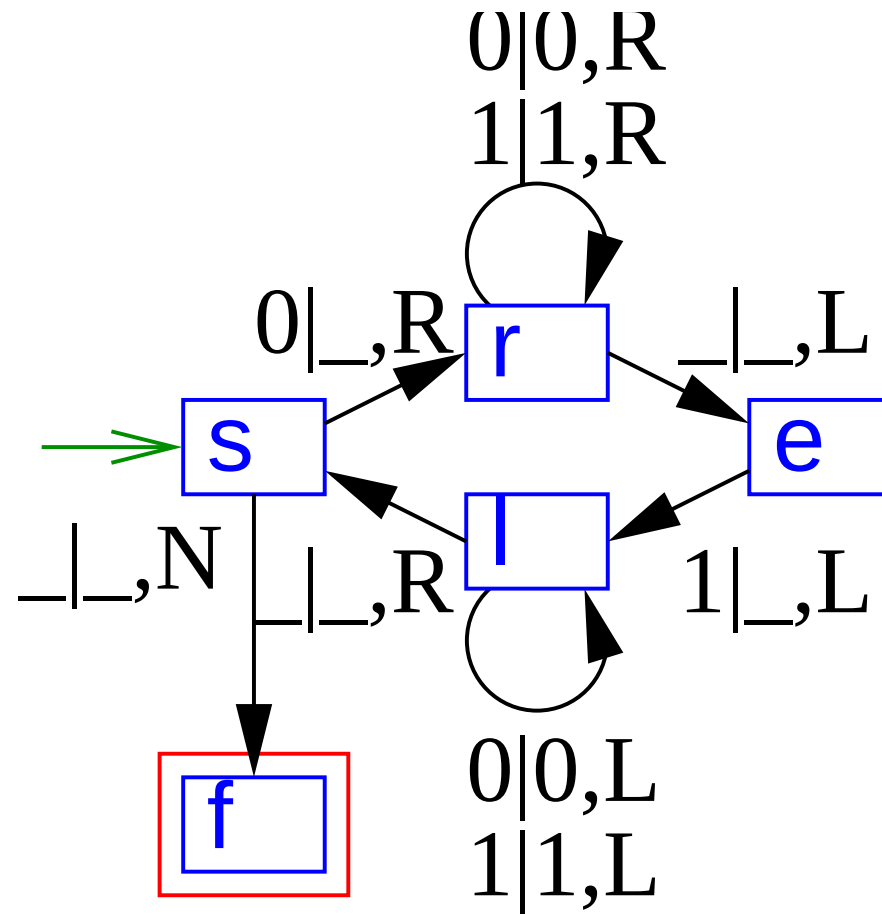


(s)000111	(q ₄)00011	0(q ₂)1	(q ₇)
0(s)00111	␣(q ₅)0011	(q ₃)0␣	
00(s)0111	0(q ₆)011	(q ₃)␣0	
000(s)111	00(q ₆)11	(q ₄)0	
0001(q ₁)11	001(q ₆)1	␣(q ₅)	
00011(q ₁)1	0011(q ₆)		
000111(q ₁)	001(q ₂)1	0 0, R	
00011(q ₂)1	00(q ₃)1␣		
0001(q ₃)1␣	0(q ₃)01		
0001(q ₃)1	(q ₃)001		
000(q ₃)11	(q ₃)␣001		
00(q ₃)011	(q ₄)001		
0(q ₃)0011	␣(q ₅)01		
(q ₃)00011	0(q ₆)1		
(q ₃)␣00011	01(q ₆)		





Пример: $\{0^n 1^n : n \geq 0\}$.





Пример: $\{0^n 1^n : n \geq 0\}$.

Нека $k \geq 1$, $w \in \{0, 1\}^*$

ε : $(s) \vdash (f)$.

0 : $(s)0 \vdash (r) \vdash (e)$ завършва.

$1w$: $(s)1w$ завършва.

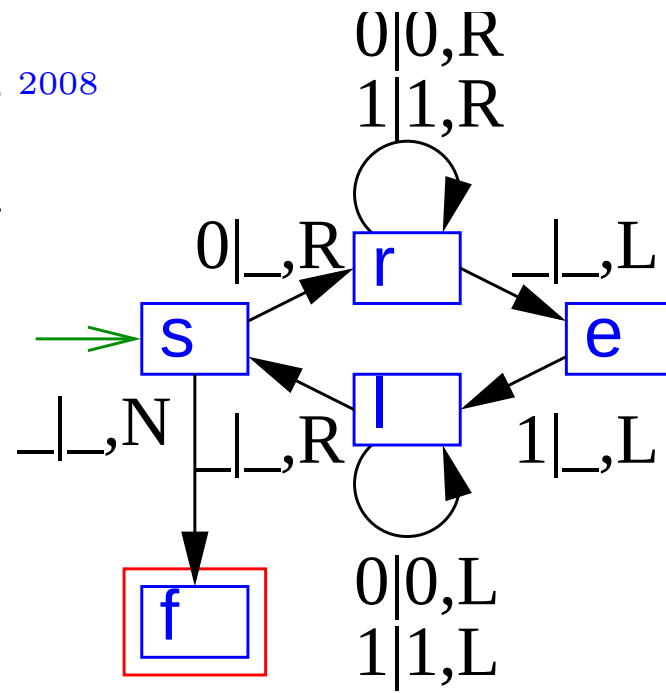
$0w0$: $(s)0w0 \vdash (r)w0 \vdash^{w|+1} w0(r) \vdash w(e)0$ завършва.

$0w1$: $(s)0w1 \vdash (r)w1 \vdash^{w|+1} w1(r) \vdash w(e)1 \vdash^{w|+1} (\ell) \sqcup w \vdash (s)w$

$0^n 1^n$: $(s)0^n 1^n \vdash^* (s)0^{n-1} 1^{n-1} \vdash^* \dots \vdash^* (s) \vdash (f)$

$0u1$, $u \notin \{0^n 1^n : n \geq 0\}$: $(s)0u1 \vdash^* (s)u$. Не завършва в f .

(Индукция)





Изчислими **функции** с машини на Тюринг

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ **изчислява** частичната функция

$$f_T : \Sigma^* \rightarrow \Gamma^* \Leftrightarrow$$

$$f_T(w) := \begin{cases} v & \text{ако } T \text{ завършва при } \textbf{ВХОД } w \text{ с } \textbf{ИЗХОД } v \\ ((s)w \Rightarrow u(q)\textbf{v}), q \in F \\ \perp = (\text{не е дефинирана}) & \text{иначе} \end{cases}$$

g е **изчислима по Тюринг** $\Leftrightarrow \exists T : f_T = g$

Забележка: ако $g(x) = \perp$, T не завършва.



Разрешими езици

Един език $L \subseteq \Sigma^*$ е **разрешим** $\iff$ ако характеристичната функция χ_L е изчислима Тюринг.

$$\chi_L: \Sigma^* \rightarrow \{0, 1\} \quad \text{където} \quad \chi_L(w) = \begin{cases} 1 & \text{ако } w \in L \\ 0 & \text{иначе} \end{cases}$$

Например: $\{0^n 1^n : n \geq 0\}$ е разрешим.

Лема: Един език $L \subseteq \Sigma^*$ е разрешим, ако има машина на Тюринг T , която завършва винаги, и:

- $\square \quad \forall w (w \in L \Rightarrow (s)w \vdash^* x(f)y) \text{ за някое } f \in F$
- $\square \quad \forall w (w \notin L \Rightarrow (s)w \vdash^* x(q)y) \text{ за някое } q \notin F \text{ (Error)}$



Полуразрешими езици

Един език $L \subseteq \Sigma^*$ е **полуразрешим** $\iff$ ако полухарактеристичната му функция χ'_L е изчислима Тюринг, където

$$\chi'_L(w) = \begin{cases} 1 & \text{ако } w \in L \\ \perp & \text{иначе} \end{cases}$$

Лема: Всеки разрешим език е полуразрешим.

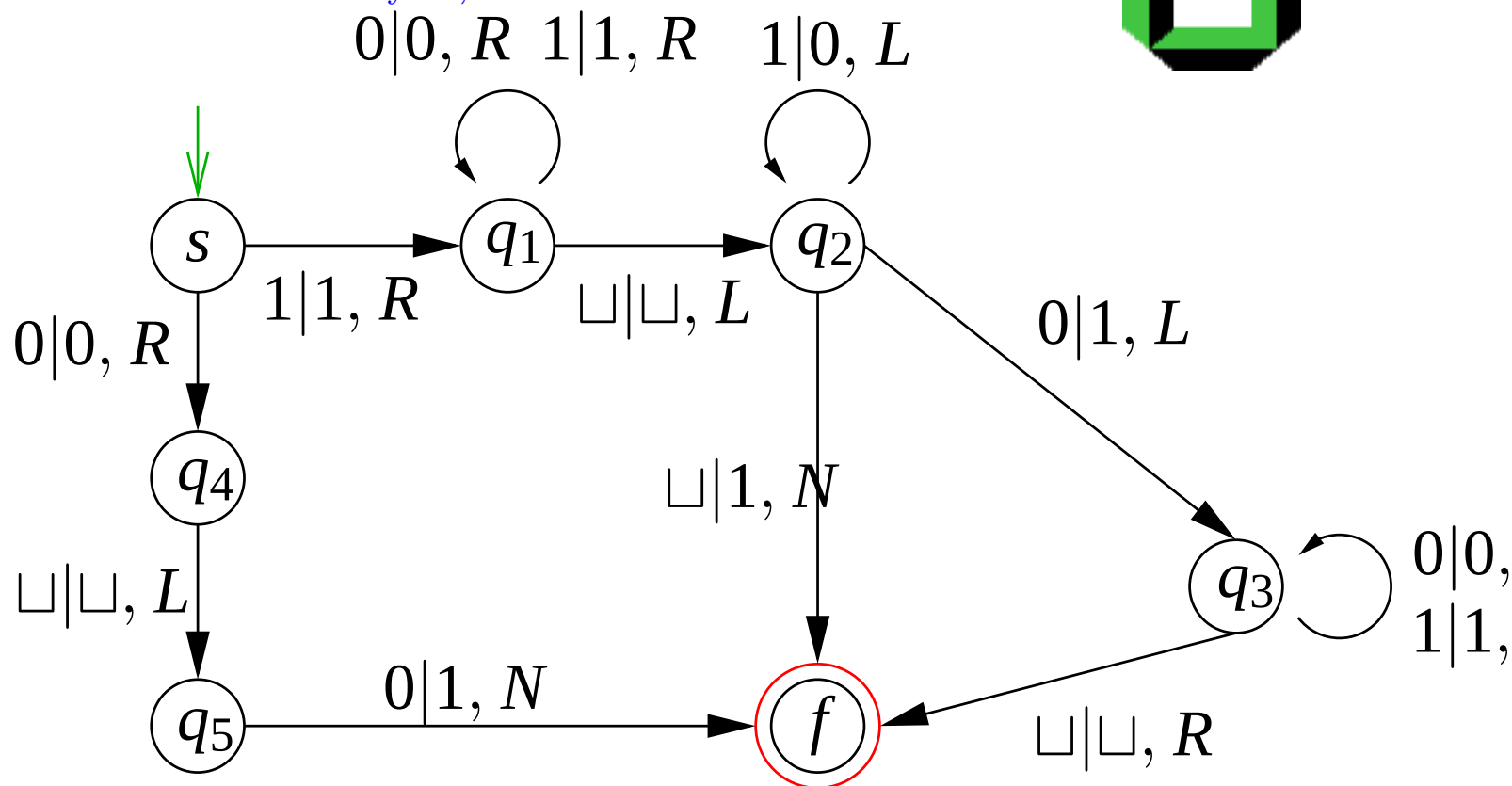


Изчислими с машини на Тюринг функции в естествените числа

$f : \mathbb{N}^k \rightarrow \mathbb{N}$ е изчислима по Тюринг $\Leftrightarrow$
 $\exists T = (Q, \Sigma, \Gamma, \delta, s, F) : \forall n_1, \dots, n_k, m \in \mathbb{N} :$
 $f(n_1, \dots, n_k) = m \Leftrightarrow$
 $(s)\text{bin}(n_1)\# \dots \# \text{bin}(n_k) \vdash^* u(q)\text{bin}(m), q \in F$



Пример

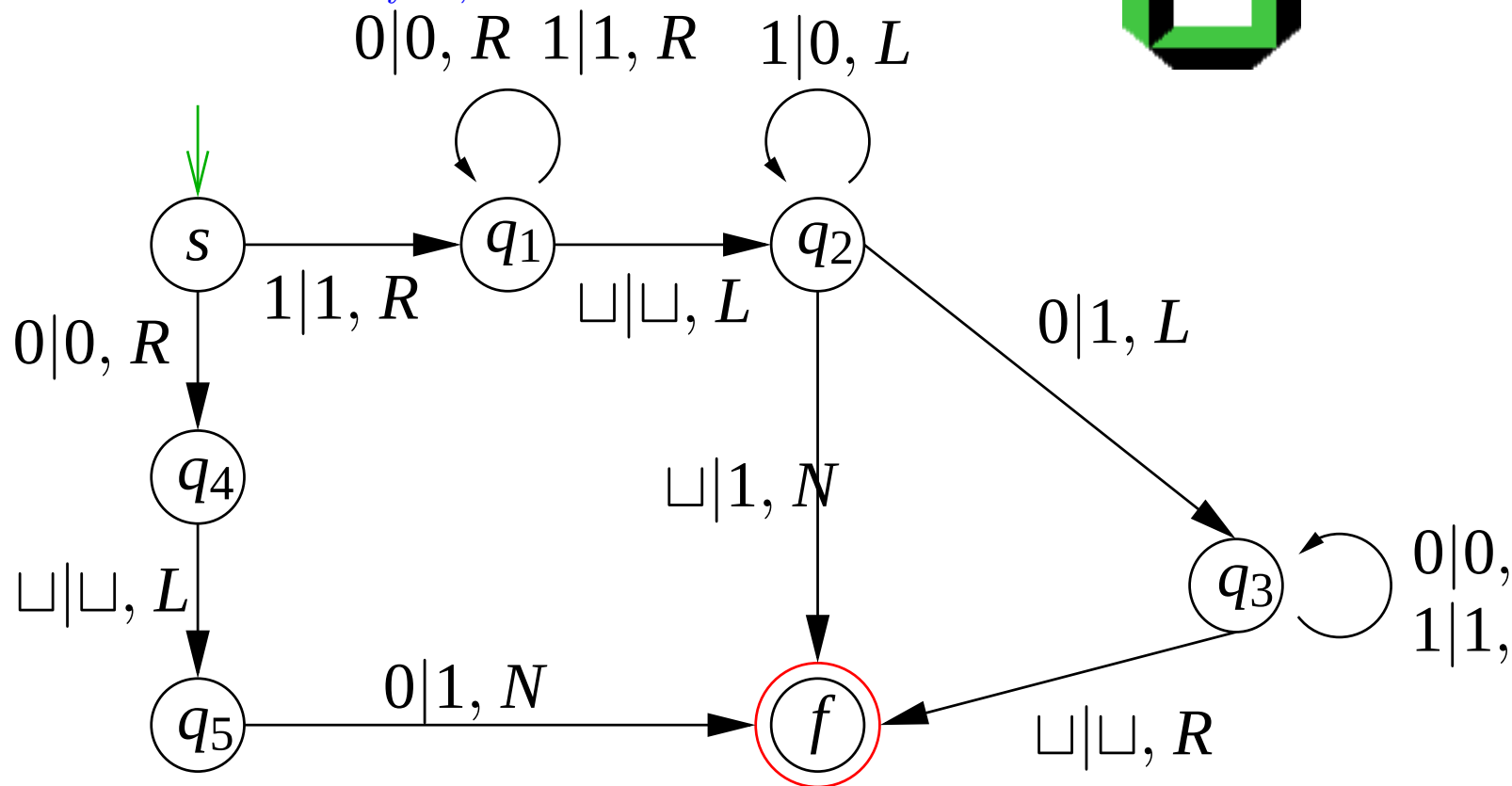


$$f(w) = \begin{cases} w + 1 & \text{ако } w \in 0 \cup 1(0 \cup 1)^*, \\ & w \text{ като двоично число} \\ \text{недефинирана} & \text{иначе} \end{cases}$$

Забележка: Непоказаните дъги са **безкрайни цикли**.



Пример



$(s)11 \vdash 1(q_1)1 \vdash 11(q_1) \vdash 1(q_2)1 \vdash (q_2)10 \vdash (q_2)\sqcup 00 \vdash (f)100$

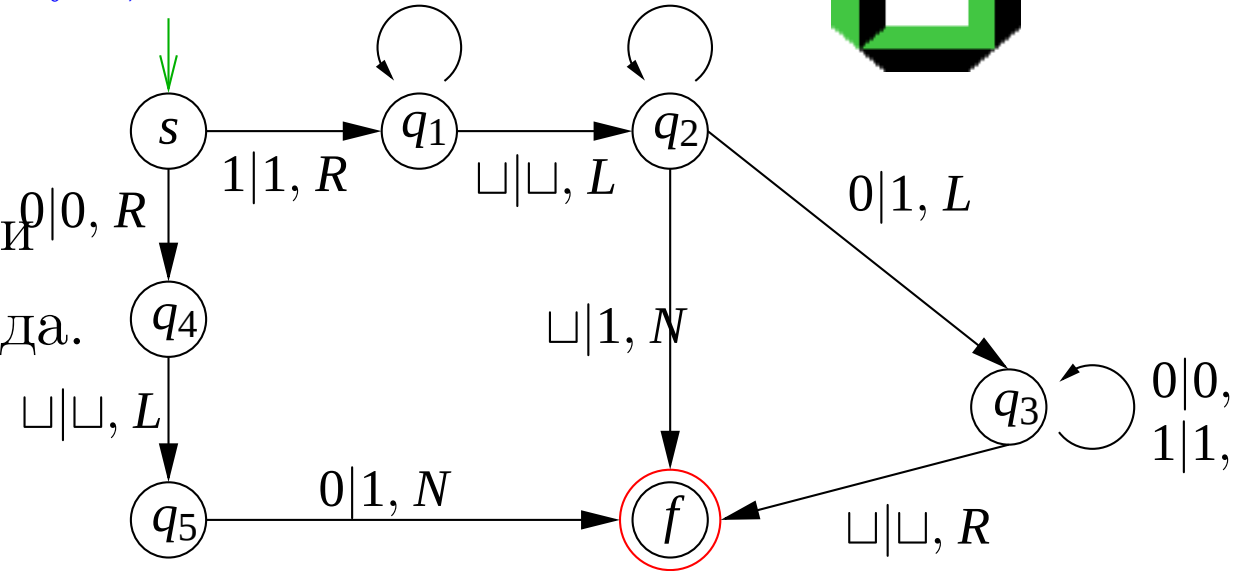


Разглеждане на случаи

по структурата на входа.

Нека $w \in \{0, 1\}^*$,

$a \in \{0, 1\}, n \geq 1$.



0: $(s)0 \vdash 0(q_4) \vdash (q_5)0 \vdash (f)1$

0aw: $(s)0aw \vdash 0(q_4)aw$ не е дефинирана

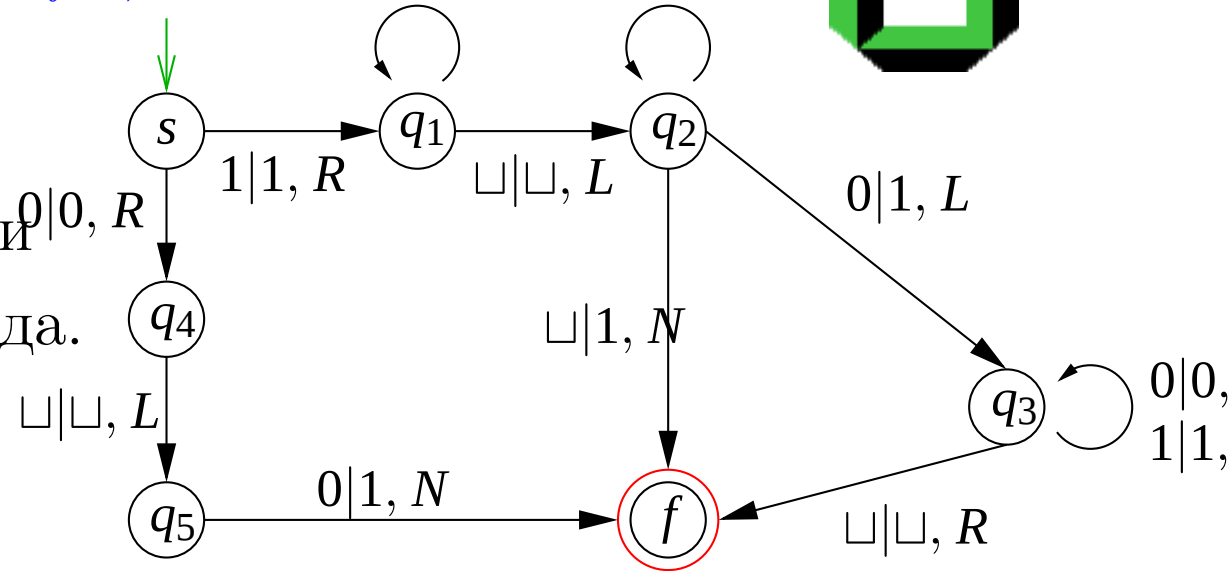


Разглеждане на случаи

по структурата на входа.

Нека $w \in \{0, 1\}^*$

$a \in \{0, 1\}, n \geq 1$.



$$1^n: (s)1^n \vdash 1(q_1)1^{n-1} \vdash 1^n(q_1) \vdash 1^{n-1}(q_2)1 \vdash (q_2)\sqcup 0^n \vdash (f)10^n$$

$$1w0: (s)1w0 \vdash 1(q_1)w0 \vdash 1w0(q_1) \vdash 1w(q_2)0 \vdash 1w(q_3)1 \vdash (q_3)\sqcup 1w1 \vdash (f)1w1$$

$1w01^n:$

$$(s)1w01^n \vdash 1(q_1)w01^n \vdash 1w01^n(q_1) \vdash 1w01^{n-1}(q_2)1 \vdash 1w(q_2)00^n \vdash 1w(q_3)10^n \vdash (q_3)\sqcup 1w10^n \vdash (f)1w10^n$$



Пример: Никъде недефиниранта функция

$$T = (\{s\}, \Sigma, \Gamma, \delta, s, \{\})$$

$$\forall a \in \Gamma : \delta(s, a) = (s, a, R)$$



Програмни техники и конструкции на машини на Тюринг

- ☐ Запомняне на символ
- ☐ Основни машини
- ☐ Комбиниране на машини - композиция и разклонение
- ☐ While-loops



Запомняне на символ

Нека $x \in A \subseteq \Gamma$, ($|A| < \infty$!):

$$Q \rightsquigarrow Q \times A.$$

Пример M е МТ, която запаметява първия символ на входната дума и завършва успешно, ако той не се среща на друго място в думата.

$$\delta([s, \sqcup], 0) = ([q, 0], 0, R) \quad \delta([s, \sqcup], 1) = ([q, 1], 1, R)$$

$$\delta([q, 0], 1) = ([q, 0], 1, R) \quad \delta([q, 1], 0) = ([q, 1], 0, R)$$

$$\delta([q, 0], \sqcup) = (f, \sqcup, N) \quad \delta([q, 1], \sqcup) = (f, \sqcup, N)$$



Композиция и разклонение

Основни машини (за 1 стъпка): за всяко $b \in \Gamma$:

$$a : \delta(s, b) = (f, a, N)$$

$$L : \delta(s, b) = (f, b, L)$$

$$R : \delta(s, b) = (f, b, R)$$

Композиция

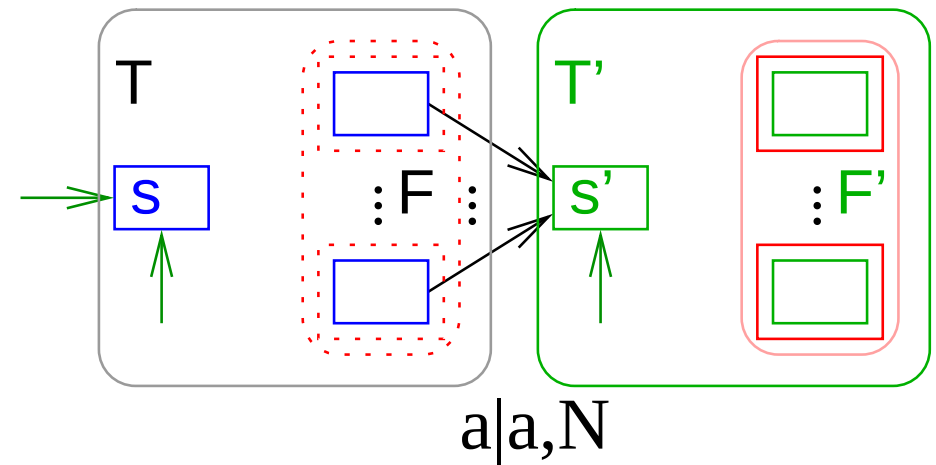
Дадено: $T = (Q, \Sigma, \Gamma, \delta, s, F)$ и $T' = (Q', \Sigma, \Gamma', \delta', s', F')$.

Изход: Машина на Тюринг $T^\circ = (Q^\circ, \Sigma, \Gamma^\circ, \delta^\circ, s, F')$ за

$$f_{T'}(f_T(x)): Q^\circ = Q \dot{\cup} Q' \quad \Gamma^\circ = \Gamma \cup \Gamma'$$



$$\delta^\circ(q, a) = \begin{cases} \delta(q, a) & \text{ако } q \in Q \setminus F \\ (s', a, N) & \text{ако } q \in F \\ \delta'(q, a) & \text{ако } q \in Q' \end{cases}$$





Разклонение

Дадено: $T = (Q, \Sigma, \Gamma, \delta, s, F)$, $T' = (Q', \Sigma, \Gamma', \delta', s', F')$ и $T'' = (Q'', \Sigma, \Gamma'', \delta'', s'', F'')$.

Исход: Машина на Тюринг $T^\circ = (Q^\circ, \Sigma, \Gamma^\circ, \delta^\circ, s, F' \cup F'')$

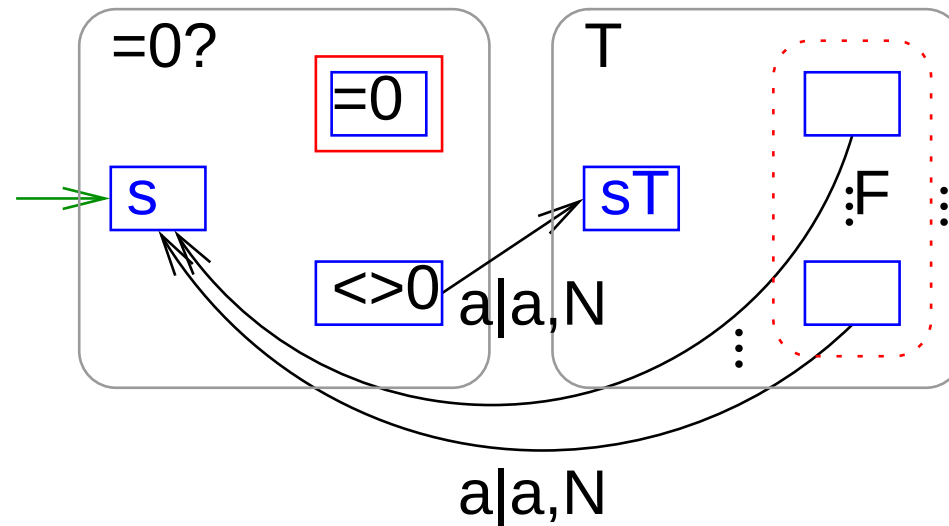
$$Q^\circ = Q \dot{\cup} Q' \dot{\cup} Q'', \Gamma^\circ = \Gamma \cup \Gamma' \cup \Gamma''$$

$$f_{T^\circ}(x) = \begin{cases} f_{T'}(f_T(x)) & \text{ако } f_T(x) = a \\ f_{T''}(f_T(x)) & \text{ако } \downarrow f_T(x) \neq a \end{cases}.$$

$$\delta^\circ(q, b) = \begin{cases} \delta(q, b) & \text{ако } q \in Q \setminus F \\ (s', b, N) & \text{ако } q \in F \text{ \& } b = a \\ (s'', b, N) & \text{ако } q \in F \text{ \& } b \neq a \\ \delta'(q, b) & \text{ако } q \in Q' \\ \delta''(q, b) & \text{ако } q \in Q'' \end{cases}$$



While-loops: While $i \neq 0$ Do $\text{tape} := f_T(\text{tape})$



Примери: $R_{\sqcup}$ - сканира надясно докато намери $\sqcup$
 (аналогично $L_{\sqcup}$)

Copy: $(q)w\sqcup \vdash (f)w\sqcup w$ - копира думата

Shift R: $(q)\sqcup w\sqcup \vdash (f)w\sqcup w$ - премества думата надясно



Многолентови машини на Тюринг

k - лентова машина на Тюринг (k - глави):

$T = (Q, \Sigma, \Gamma, \delta, s, F)$, където

$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k, \{L, R, N\}^k$.

$\delta(q, (a_1, \dots, a_k)) = (p, (b_1, \dots, b_k), (C_1, \dots, C_k)),$
 $C_1, \dots, C_k \in \{L, R, N\}.$

Теорема За всяка k -лентова машина на Тюринг M съществува еднолентова машина на Тюринг M' , такава че за всяка дума x на първата лента $M(x)$ завършва с резултат y на първата лента $\iff M'$ завършва над x с резултат y .



Идея

$$\Sigma' = \Sigma \cup (\Sigma \times \{0, 1\})^k.$$

На i -та позиция единица показва главата на i -тата лента.

- Фаза 1. Замества всеки символ a_i с $(a_i, 0, \sqcup, 0, \dots, \sqcup, 0)$.
Отпред $(\sqcup, 1, \sqcup, 1, \dots, \sqcup, 1)$.
- Фаза 2. Симулира изчислението на M върху M'
докато M завърши
 - (а) сканиране надясно : запомня къде са главите
 - (б) сканиране наляво и надясно за симулиране на M
- Фаза 3. Преобразува всеки k -местен символ в унарен символ: $(b_1, 0, \dots, b_k, 0) \rightsquigarrow b_1$. Игнорира другите ленти без първата.



Варианти на машини на Тюринг

k глави: $\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R, N\}^k$

k ленти: т.е. по една глава за лента

d -размерна лента: например $d = 2$,

$$\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R, U, D, N\}$$

вероятностни: допълнително инструкции за движение на главата с рандом бит

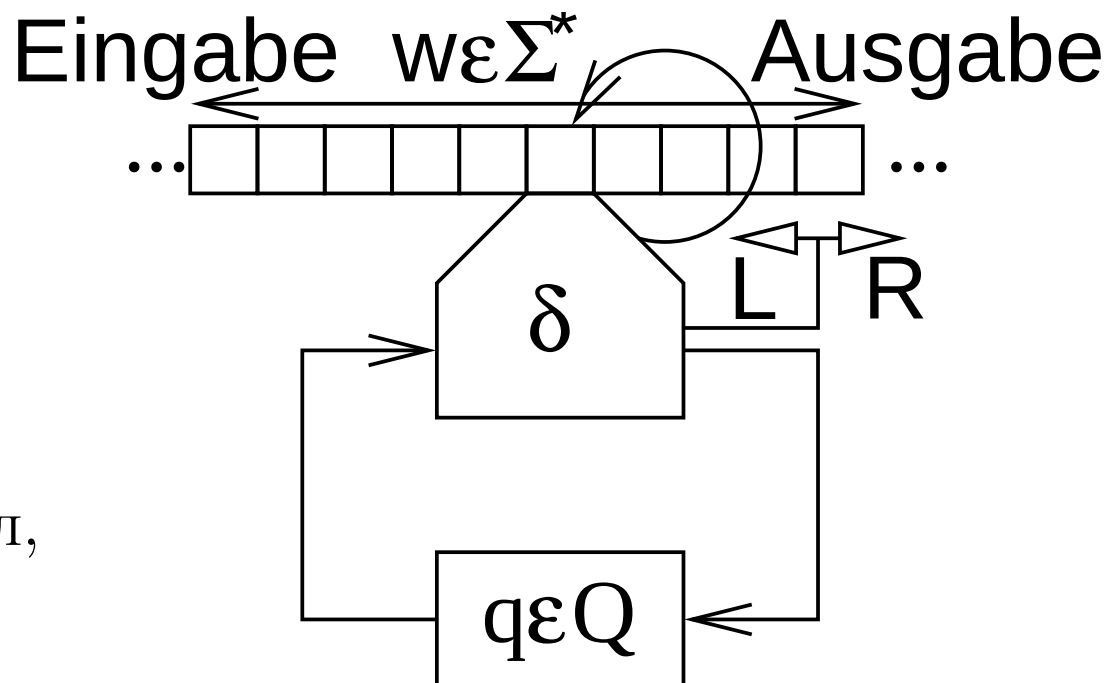
Времето T за k -лентова-МТ $\rightarrow \mathcal{O}(T^2)$ за едно-лентова-МТ



Недетерминистични машини на Тюринг (НМТ)

$T = (Q, \Sigma, \Gamma, \delta, s, F)$:

- Q , състояния
- Σ , входна азбука
- Γ азбука на лентата,
 $\sqcup \notin \Sigma$: празен символ,
 $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $\delta : Q \times \Gamma \rightarrow 2^{Q \times \Gamma \times \{L, R, N\}}$,
 функция на прехода;
- $s \in Q$, начално състояние
- $F \subseteq Q$, крайни състояния

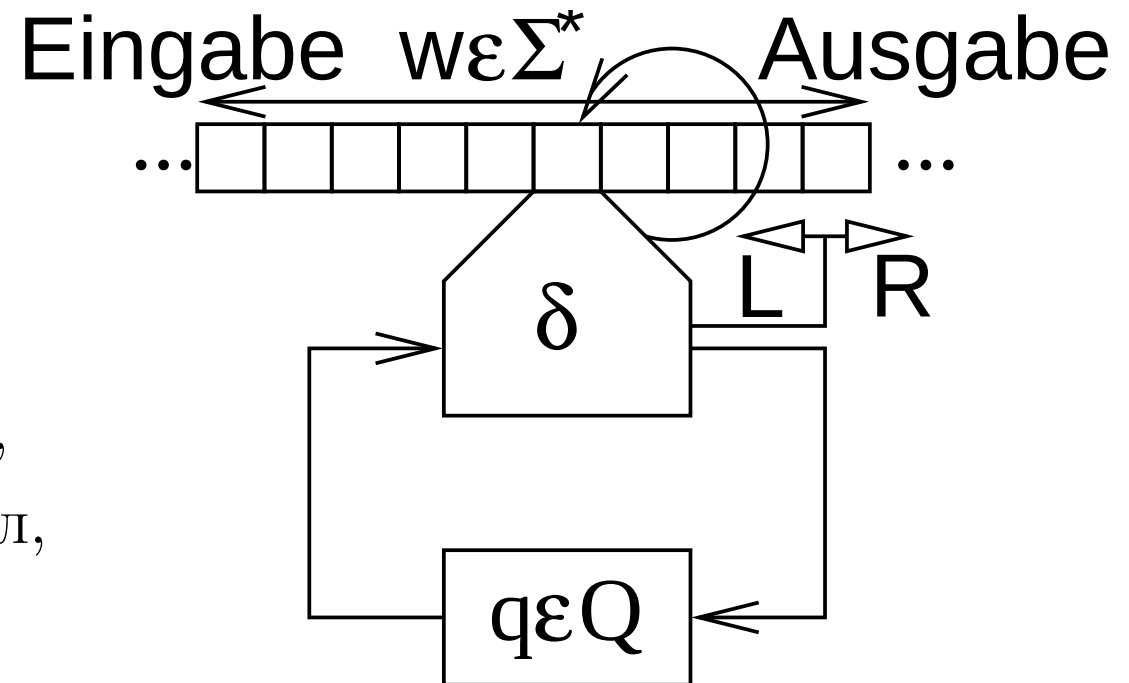




Недетерминистични машини на Тюринг (НМТ)

$T = (Q, \Sigma, \Gamma, \delta, s, F)$:

- Q , състояния
- Σ , входна азбука
- Γ азбука на лентата,
 $\sqcup \notin \Sigma$: празен символ,
 $\Sigma \cup \{\sqcup\} \subseteq \Gamma$
- $\delta \subseteq Q \times \Gamma \times Q \times \Gamma \times \{L, R, N\}$,
релация на прехода;
- $s \in Q$, начално състояние
- $F \subseteq Q$, крайни състояния





Как работи една НМТ

$$wa(q)bcv \quad \begin{array}{c} (q', b', N) \in \delta(q, b) \\ \vdash \end{array} \quad wa(q')b'cv$$

$$wa(q)bcv \quad \begin{array}{c} (q', b', L) \in \delta(q, b) \\ \vdash \end{array} \quad w(q')ab'cv$$

$$wa(q)bcv \quad \begin{array}{c} (q', b', R) \in \delta(q, b) \\ \vdash \end{array} \quad wa b'(q')cv$$

Възможни преходи между конфигурациите



Кога **завършва** една **НМТ**?

T завършва при конфигурация $w(q)av$ ако $\delta(q, a) = \{\}$.

Конвенция:

$$\forall q \in F : \forall a \in \Gamma : \delta(q, a) = \{\}$$

Разлики **ДМТ** versus **НМТ**:

δ **определя** преходите между конфигурации versus

δ **допуска** преходите между конфигурации.



Как работи една НМТ?

T приема $w \Leftrightarrow$

$\exists \alpha, \beta \in \Gamma^*, f \in F : (s)w \vdash^* \alpha f \beta$

$L(T) := \{w \in \Sigma^* : T \text{ приема } w\}.$

Дефиниция:

T изчислява функцията f ако

за всяка дума w : $f(w)$ е дефинирана т.т.к. T завършва над w

и $f(w) = v$ ако $(s)w \rightarrow \dots \rightarrow x(f)v$ с $f \in F$.



НМТ и ДМТ

Теорема Ако M е НМТ, то има ДМТ M' , такава че за всяка дума w

M завършва над w т.т.к. M' завършва над w и

$$(s)w \vdash_M^* u(f)v \Leftrightarrow (s')w \vdash_{M'}^* u(f')v.$$

Д-во: Нека $M = (Q, \Sigma, \Gamma, \delta, s, F)$

Нека $r = \max |\delta(q, a)| \forall q \in Q \ \& \ a \in \Gamma$.

Номерираме стойностите на $\delta(q, a)$ с $1, 2, \dots, r$, ако са по-малко повтаряме последния.

Тъй като M е недетерминистична, няма как да разберем на всяка стъпка, кой от тези r прехода избира M
Но, ние ще помогнем.



НМТ и ДМТ

Машина M_d - детерминистична с 2 ленти:

I лента: w -входната дума

II лента: $i_1, \dots, i_n$ - редица на преходите $i_j \leq r$.

M_d имитира M с избор на преходите, посочен от II лента.

Машина N - генерира последователно всички редици

$i_1, \dots, i_n$, за $i_j \leq r$, лексикогарфски:

$1, 2, \dots, r, 11, 12, 13, \dots, rr, 111, \dots$



НМТ и ДМТ

Машина M' : 3

I лента: w - входа, не се променя

II лента: w

III лента: $i_1, \dots, i_n$ -редица на преходите $i_j \leq r$,

1. Копира входа от I-та лента на II-та.
2. Пуска M_d на II-та и III-та лента, като след всяка симулация на M_d генерира редица от преходи с машината N на III-та и възстановява w на II-та лента.
3. Когато M_d стигне до заключително състояние, копира съдържанието от II-та на I-та.



НМТ и ДМТ

$$\downarrow M(w) \iff \downarrow M'(w)$$

Ако $\downarrow M(w)$, то след редица неуспешни опити ще намерим успешната редица от преходи и $\downarrow M'(w)$.

Ако $\downarrow M'(w)$, то $\downarrow M_d(w)$ и следователно $\downarrow M(w)$.

Но ако M завърши за n стъпки, то M' завършва за $\exp(n)$ стъпки.



Граматики от **тип 1**

Пример: $L = \{a^n b^n c^n : n \in \mathbb{N}\}$

$$G = (V, \Sigma, P, S), \quad V = \{S, A, B, C, T_a, T_b, T_c\}, \quad \Sigma = \{a, b, c\}$$

R:

$$S \longrightarrow ABCS \quad CT_c \longrightarrow T_c c$$

$$S \longrightarrow T_c \quad CT_c \longrightarrow T_b c$$

$$BA \longrightarrow AB \quad BT_b \longrightarrow T_b b$$

$$CB \longrightarrow BC \quad BT_b \longrightarrow T_a b$$

$$CA \longrightarrow AC \quad AT_a \longrightarrow T_a a$$

$$T_a \longrightarrow \varepsilon$$



Граматика от **тип 0** (от общ тип)

Пример: $L = \{a^{2^n} : n \in \mathbb{N}\}$

$$G = (V, \Sigma, P, S), \quad V = \{S, A, B, C, D, E\}, \quad \Sigma = \{a\}$$

R:

$$S \longrightarrow ACaB$$

A, B маркери

$$Ca \longrightarrow aaC$$

удвоява a

$$CB \longrightarrow DB$$

докато стигне B , $C \leftrightarrow D$

$$CB \longrightarrow E$$

за край

$$aD \longrightarrow Da$$

при D , наляво

$$AD \longrightarrow AC$$

докато стигне A , $D \leftrightarrow C$

$$aE \longrightarrow Ea$$

при D , наляво

$$AE \longrightarrow \varepsilon$$

E маха границите.



Твърдение: $\forall$ тип 0 език $L : \exists \text{MT } T : L(T) = L$

Д-во: Нека $G = (V, \Sigma, P, S)$ -граматика от тип 0 с $L(G) = L$.

Да разгледаме НМТ $T = (Q, \Sigma, (\Sigma \cup V), \delta, s, F)$:

```

Procedure inL(z)           // начална конфигурация (s)z
    invariant “съдържанието на лентата”  $\xRightarrow{*} z$ 
    while tape  $\neq S$  do
        if  $\exists w \rightarrow \alpha \in P : \text{tape} = x\alpha y$  then // 2× недетрм. избор!
            tape := xwy           // заместващо правило!
        else reject z
    accept z
    
```

приемащо изчисление $\xrightarrow{Inv.} \exists$ извод.

$S \xRightarrow{*} z \longrightarrow \exists$ съответстващо успешно изчисление .



Твърдение: $\forall L : \exists \text{НМТ } T : L(T) = L \rightarrow L$ е език от тип 0.

Д-во: Нека $T = (Q, \Sigma, \Gamma, \delta, s, F)$ е НМТ и $L(T) = L$.

Да разгледаме граматика от тип 0

$$G = (V = \{S\} \cup (\Gamma \times \Sigma) \cup (Q \times \Gamma \times \Sigma), \Sigma, P, S).$$

Идея: МТ-конфигурация $\alpha(q)a\beta \rightsquigarrow$ твърдение за $\alpha(q,a)\beta$ плюс екстра информация за оригиналния вход.

3 фази на извода:

1. **генерираме** дума от Σ^* .
2. **симулираме** изчислението на МТ.
3. след успешно приемане **възстановяваме** входната дума



Фаза 1: генериране на дума от Σ^*

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ НМТ и $L(T) = L$.

$G = (\mathbf{V} = \{\mathbf{S}\} \cup (\mathbf{\Gamma} \times \mathbf{\Sigma}) \cup (\mathbf{Q} \times \mathbf{\Gamma} \times \mathbf{\Sigma}), \Sigma, P, S)$.

$\{S \rightarrow S(a, a) : a \in \Sigma\} \subseteq P$

$\{S \rightarrow (s, a, a) : a \in \Sigma\} \subseteq P$

край на фаза 1.

(и специални грижи ако $\varepsilon \in L$)

Пример: $S \Rightarrow S(c, c) \Rightarrow S(b, b)(c, c) \Rightarrow (s, a, a)(b, b)(c, c)$

отговаря на началната конфигурация $(s)abc$



Фаза 2: **симулиране** на изчислението на МТ

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ НМТ с $L(T) = L$.

$G = (\mathbf{V} = \{\mathbf{S}\} \cup (\mathbf{\Gamma} \times \mathbf{\Sigma}) \cup (\mathbf{Q} \times \mathbf{\Gamma} \times \mathbf{\Sigma}), \Sigma, P, S)$.

$$P := P \cup \left\{ \begin{aligned} &(q, a, c) \rightarrow (q', a', c) && : (q', a', \mathbf{N}) \in \delta(q, a), c \in \Sigma \\ &\cup \{ (b, c')(q, a, c) \rightarrow (q', b, c')(a', c) && : (q', a', \mathbf{L}) \in \delta(q, a), c \in \Sigma \\ &\cup \{ (q, a, c)(b, c') \rightarrow (a', c)(q', b, c') && : (q', a', \mathbf{R}) \in \delta(q, a), c \in \Sigma \end{aligned} \right.$$

Пример: $(s, a, a)(b, b)(c, c) \xRightarrow{*} (x, a)(f, y, b)(z, c)$

отговаря на редицата от конфигурации

$(s)abc \vdash \cdots \vdash x(f)yz$



Фаза 3: **ВЪЗСТАНОВЯВА** ВХОДНАТА дума

$T = (Q, \Sigma, \Gamma, \delta, s, F)$ НМТ с $L(T) = L$.

$G = (V = \{S\} \cup (\Gamma \times \Sigma) \cup (Q \times \Gamma \times \Sigma), \Sigma, P, S)$.

$\{(f, a, c) \rightarrow c : f \in F, a \in \Gamma, c \in \Sigma\} \subseteq P$

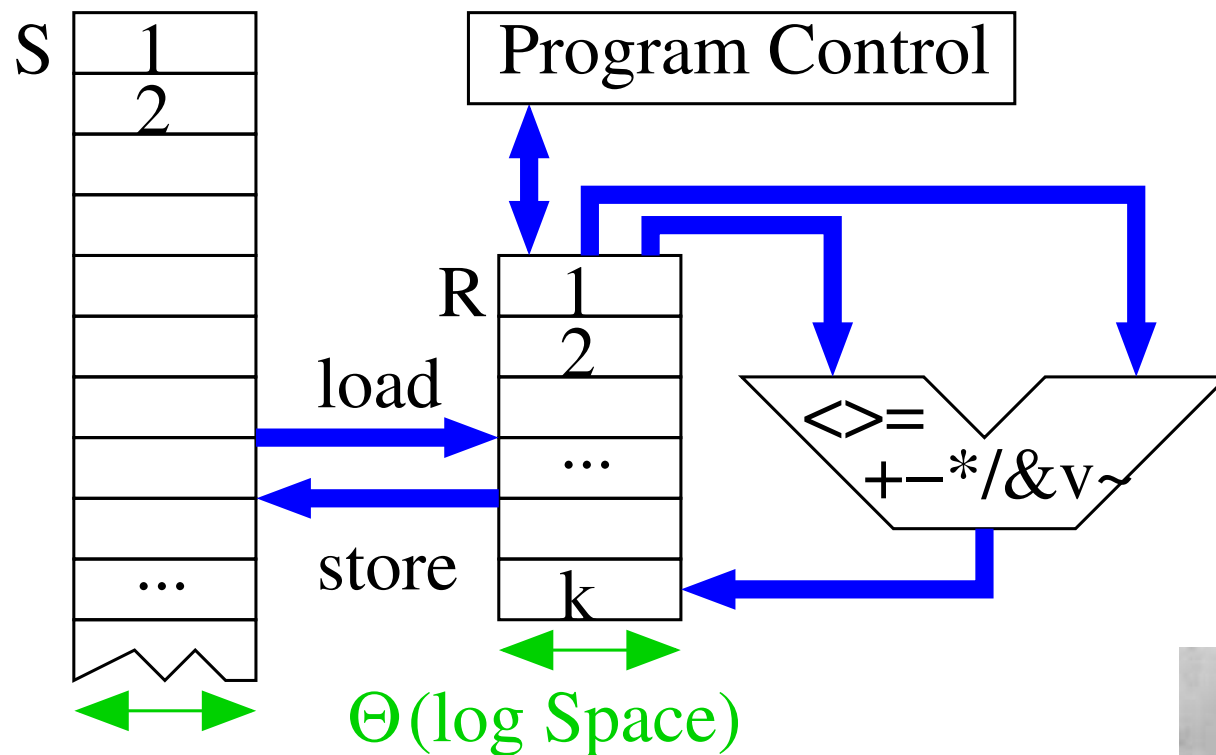
Приемане

$\{(a, b) \rightarrow b : a \in \Gamma \wedge b \in \Sigma\} \subseteq P$

Пример: $(x, a)(f, y, b)(z, c) \Rightarrow (x, a)b(z, c) \Rightarrow ab(z, c) \Rightarrow abc$



RAM: Машины с директен достъп до паметта (random access memory)

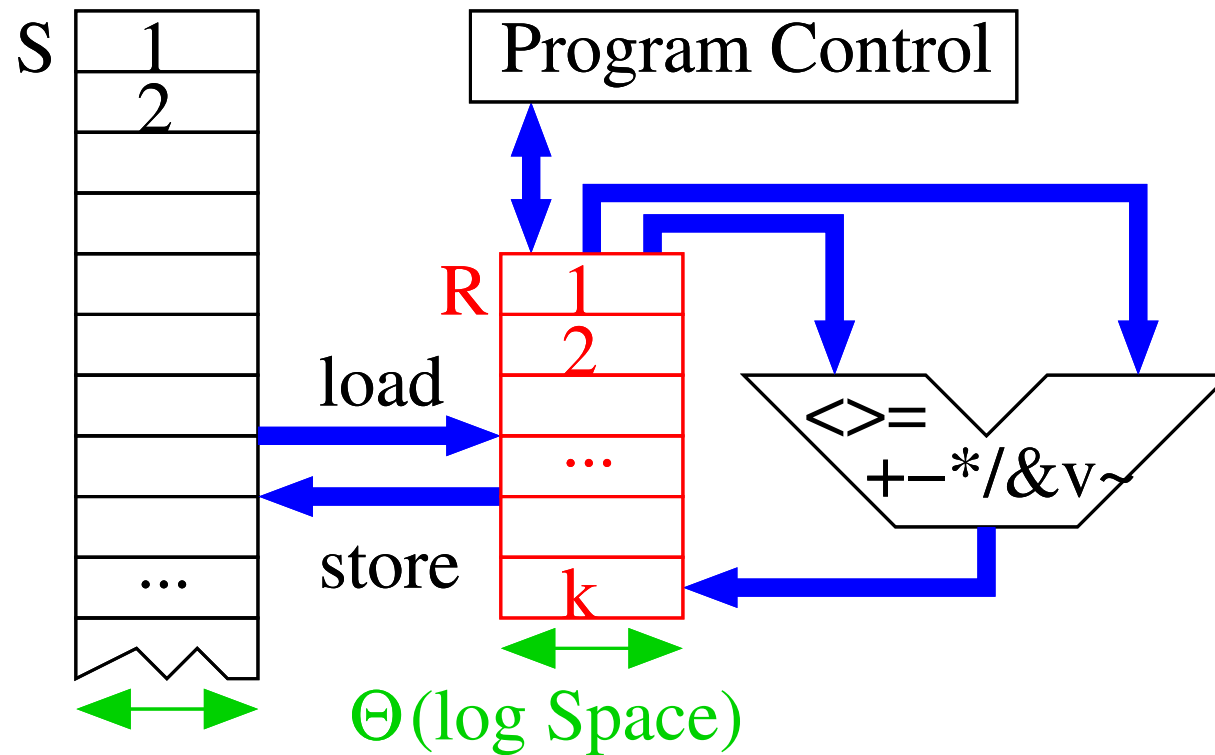


Модерна (RISC) адаптация
на фон Ноймановия модел [1945]





Регистри



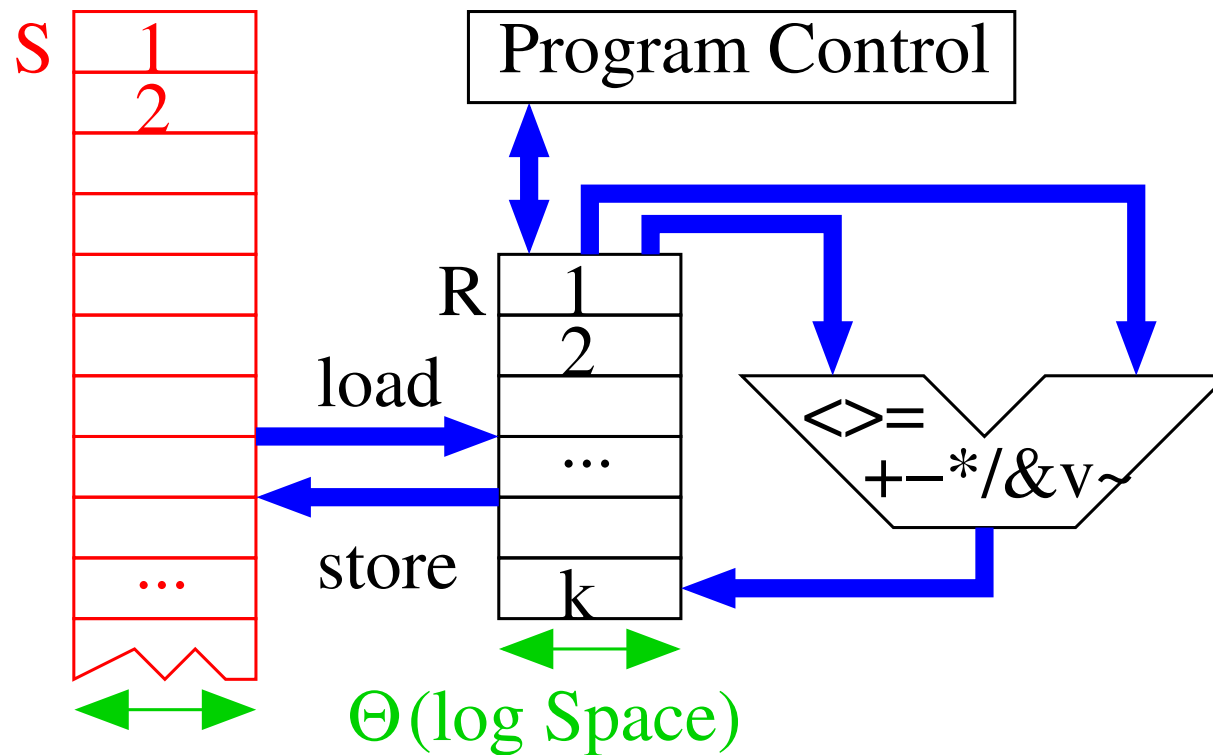
k (всяка константа) регистри

$R_1, \dots, R_k$ за

(малки) числа



Основната памет

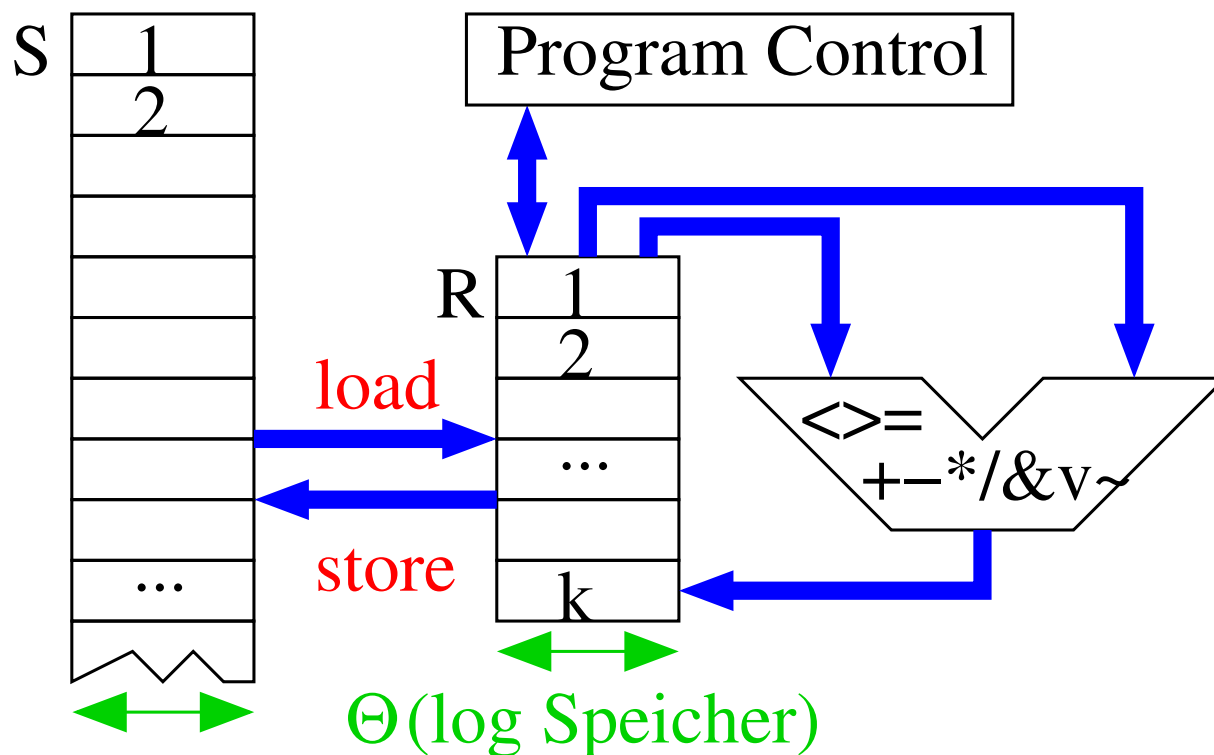


Неограничен брой клетки

$S[1], S[2] \dots$ за
(малки) числа



Достъп до паметта

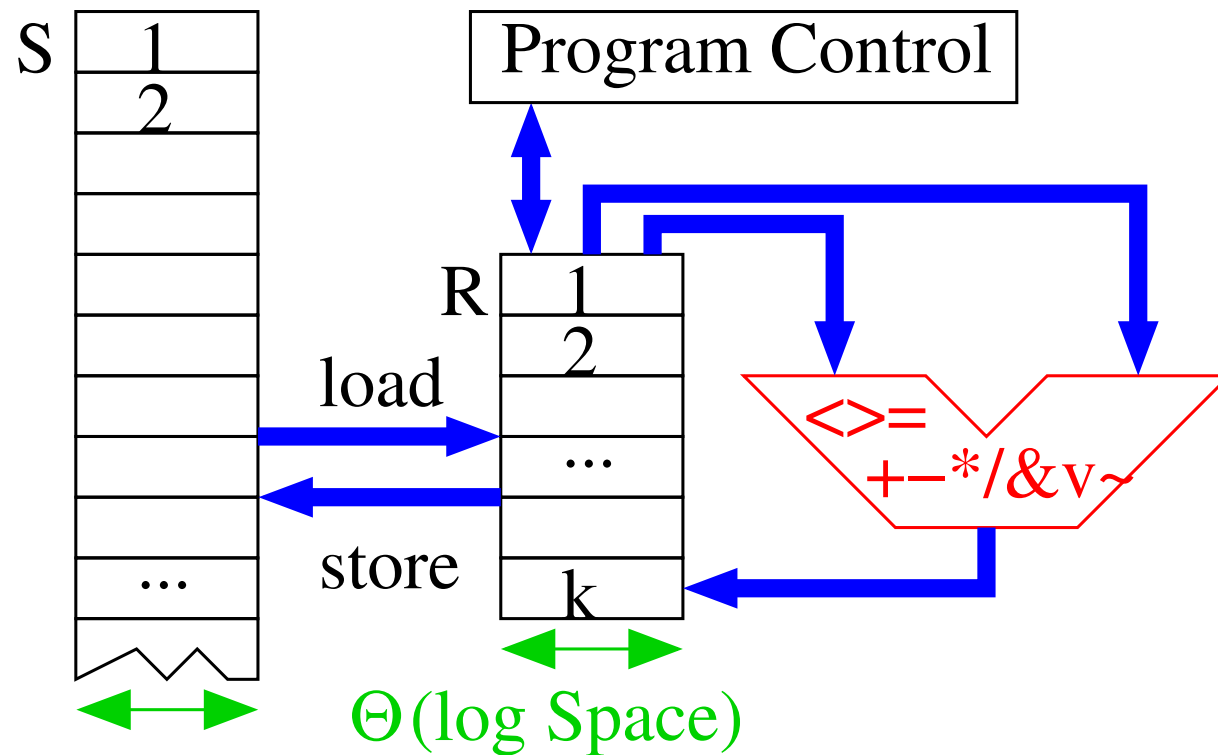


$R_i := S[R_j]$ **зарежда** съдържанието на клетката от паметта $S[R_j]$ в регистър R_i .

$S[R_j] := R_i$ **запомня** регистъра R_i в клетката $S[R_j]$.



Изчисление

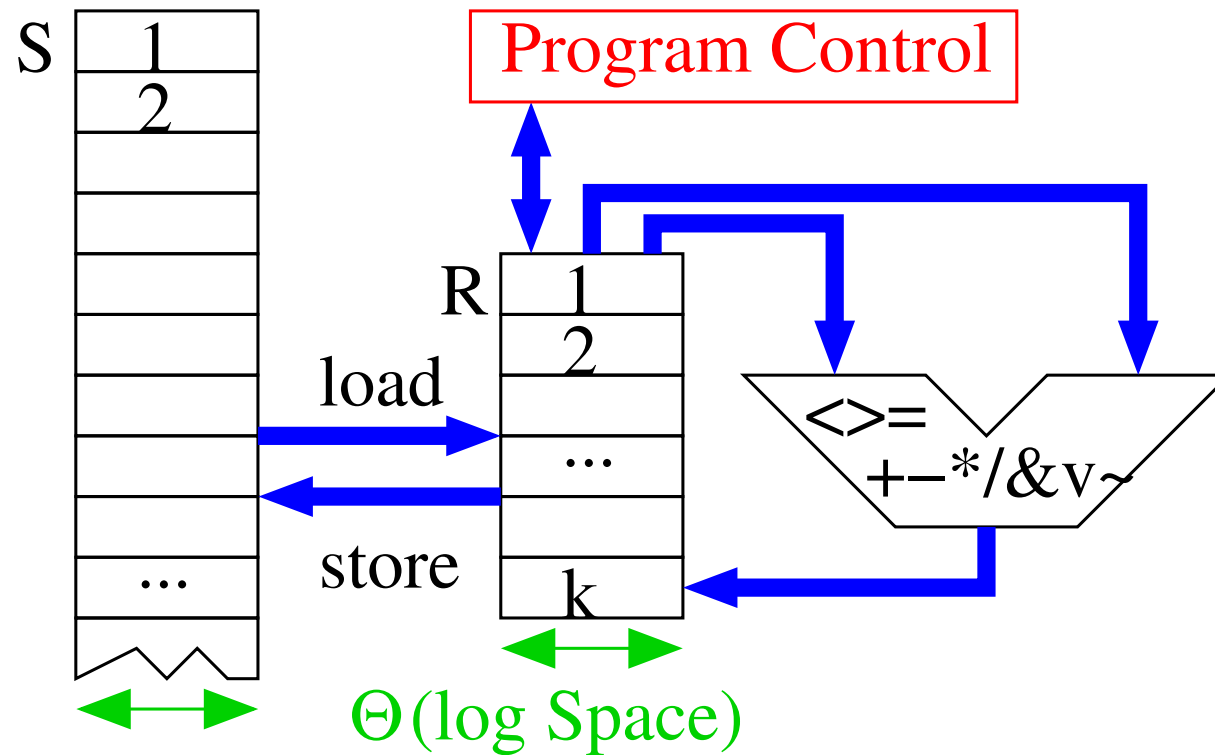


$R_i := R_j \odot R_\ell$ Регистрова аритметика.

‘ $\odot$ ’ означава голямо количество от операции
аритметични, сравнения, логически



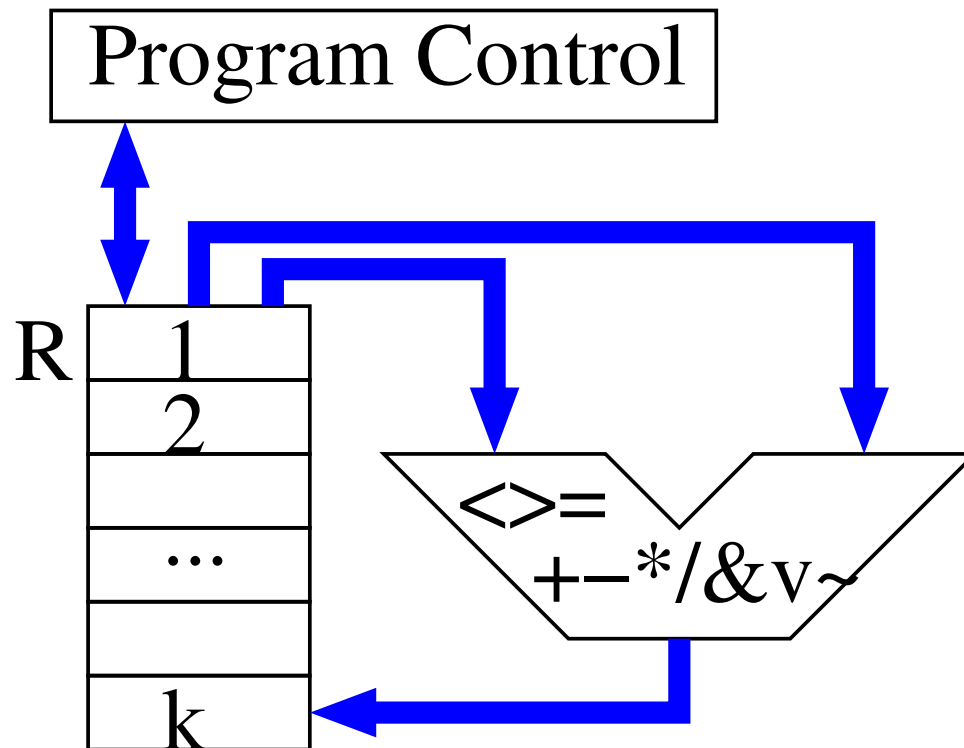
Условен преход



$JZ\ j, R_i$ продължава изпълнението на програмата с команда с етикет j (goto j) ако $R_i = 0$



Машини с Неограничени Регистри (МНР)
 $\approx$ RAM – памет + произволно големи числа +
произволен (краен) брой (Шефердсон, Стържиц, 1963)





МНР-изчислимост

Конфигурация: $(q, R_1, \dots, R_k)$

q е брояч на програмните команди

" $\vdash^*$ " дефинирахме.

$f : \mathbb{N}^{k'} \rightarrow \mathbb{N}$, $k' \leq k$ е **МНР** изчислима $\Leftrightarrow$

$\exists$ МНР $M : \forall n_1, \dots, n_{k'}, m \in \mathbb{N} :$

$$f(n_1, \dots, n_{k'}) = m \Leftrightarrow$$

$$(1, n_1, \dots, n_{k'}, 0^{k-k'}) \vdash^* (q, f(n_1, \dots, n_k), \dots)$$

with PROGRAM[q] = HALT



РАМ-изчислимоСТ

Конфигурация: $(q, R_1, \dots, R_k, S)$

Нека M е РАМ:

вход: $w \in \Sigma^n$ в $S[1], \dots, S[n]$

изход: $f_M(w)$ в $S[1], \dots, S[|f_M(w)|]$

когато HALT-командата се изпълни.

Рзаглеждаме естествени числа и трябва да ги кодираме!

Аналогично като МТ



Езици за програмиране от високо ниво

Java, C/C++, Pascal,...

ML, Lisp,...

Prolog, Oz,...

...

са най-популярните програмни модели за нас.

Компилаторите транслират програмата в RAM-код.



МНР емулира RAM

Идея: допълнителен регистър R_S представящ паметта:

$$R_S = \sum_i S[i] \cdot 2^{bi}$$

с b = броят на RAM битовете

$S[i]$ в R_j **зареждане**:

$$R_j := \frac{R_S}{2^{bi}} \bmod 2^b .$$

$S[i] := 0$:

$$R_S := R_S - \left(\frac{R_S}{2^{bi}} \bmod 2^b \right) 2^{bi}$$

R_j в $S[i]$ **запазване**:

$$S[i] := 0; R_S := R_S + R_j \cdot 2^{bi}$$



Квантов компютър

- ☐ Един **Qubit** запаметява суперпозициите 0 и 1 (и 0 и 1- едновременно).
- ☐ Изчисления с **n** Qubit-а дава суперпозицията на **2^n** класически изчисления
- ☐ Квантовият компютър може за полиномиално време да **факторизира** и може да получи **дискретни логаритми**
- ☐ Това дава възможност много криптографски алгоритми да се реализизрат бързо



Квантов компютър: теория на изчислимостта и сложността

- Предположения от теория на сложността:
 - Факторизирането, DLog **не са в P** (същото и с рандомизацията)
 - Факторизирането, DLog не са NP.
- Машините на Тюринг могат да симулират квантовия компютър

Резултат: Квантовите компютри са по-бързи, но не по-мощни от класическите компютри



2.3 Примитивно-рекурсивни функции

Разглеждаме функции в множеството на естествените числа $\mathbb{N}$.

Основни функции

$$\square \quad O(x) = 0$$

$$\square \quad S(x) = x + 1$$

$$\square \quad I_k^n(x_1, \dots, x_n) = x_k, \quad k \leq n$$



2.4 Примитивно-рекурсивни функции

Основни операции

- Суперпозиция

$$h(x_1, \dots, x_n) = f(g_1(x_1, \dots, x_n), \dots, g_k(x_1, \dots, x_n))$$

- Примитивна рекурсия

$$h(x_1, \dots, x_n, 0) = f(x_1, \dots, x_n)$$

$$h(x_1, \dots, x_n, y + 1) = g(x_1, \dots, x_n, y, h(x_1, \dots, x_n, y))$$



Примитивно-рекурсивни функции

Една функция е **примитивно рекурсивна**, ако се получава от основните с помощта на операциите суперпозиция и примитивна рекурсия, приложени краен брой пъти.

Примери:

$$\square \quad x + 0 = x$$

$$x + (y + 1) = (x + y) + 1$$

$$\square \quad x \cdot 0 = 0$$

$$x \cdot (y + 1) = x \cdot y + x$$

$$\square \quad x^0 = 1$$

$$x^{y+1} = x^y \cdot x$$



Примитивно-рекурсивни функции

Примери:

$$\square \quad x \dot{-} 1 = 0, \text{ ако } x = 0 ; x \dot{-} 1 = x - 1 \text{ ако } x > 0.$$

$$\square \quad x \dot{-} y = 0, \text{ ако } x < y ; x \dot{-} y = x - y \text{ ако } x \geq y.$$

$$\square \quad sg(x) = 0, \text{ ако } x = 0 ; sg(x) = 1 \text{ ако } x > 0.$$

$$\square \quad \bar{sg}(x) = 1, \text{ ако } x = 0 ; \bar{sg}(x) = 0 \text{ ако } x > 0.$$

$$\square \quad |x - y| = (x \dot{-} y) + (y \dot{-} x).$$

$$\square \quad x \leq y \Leftrightarrow x \dot{-} y = 0$$



Операции, запазващи примитивната рекурсивност

□ if then else

$$h(x) = \begin{cases} f(x) & \text{ако } p(x) = 0 \\ g(x) & \text{иначе} \end{cases}$$

$$h(x) = \bar{s}g(p(x)).f(x) + sg(p(x)).g(x)$$

□ Ограничено сумиране

$$g(x, y) = \Sigma_{z < y} f(x, z)$$

□ Ограничена минимизация

$$g(x, y) = \begin{cases} \mu_{z < y} [f(x, z) = 0] & \text{ако има такова} \\ y & \text{иначе} \end{cases}$$



Примитивно-рекурсивни функции

□ $x \bmod y$, като $x \bmod 0 = x$

$$0 \bmod y = 0$$

$$x + 1 \bmod y = \begin{cases} x \bmod y + 1 & \text{ако } x \bmod y + 1 \neq y \\ 0 & \text{иначе} \end{cases}$$

□ x/y , $x/0 = 0$.

$$\square \text{ div } (x, y) = \begin{cases} 0 & \text{ако } x \bmod y = 0 \\ 1 & \text{иначе} \end{cases}$$

□ $D(x) = \sum_{z < x+1} \overline{\text{sg}}(\text{div}(x, z))$ броят на делителите на x .



Примитивно-рекурсивни функции

□ $pr(x) = \text{sg}|D(x) - 2|$ x е просто число

□ $p(x)$ = простото число с номер x $p(0) = 2, p(1) = 3, \dots$

$$p(0) = 2$$

$$p(x+1) = \mu z_{\leq 2^{2^x}} [z > p(x) \ \& \ pr(z) = 0].$$

□ $(x)_y$ = степента, с която пр. число с ном. y участва в разлагането на x .

$$(x)_y = \mu t_{\leq x} [\text{div}(p(y)^{t+1}, x) = 1].$$



Примитивно-рекурсивно кодиране

- $\pi(x, y) = 2^x(2y + 1) - 1$ е примитивно рекурсивна
- $L(\pi(x, y)) = x$ и $R(\pi(x, y)) = y$ декодиращи функции
$$L(z) = (z + 1)_0$$
$$R(z) = ((z + 1) / 2^{(z+1)_0}) / 2.$$
- Всяко число е код на наредена двойка
- Кодирането е еднозначно.



Примитивно-рекурсивни и μ -рекурсивни функции

μ операция:

$$f(x_1, \dots, x_n) = \mu z [g(x_1, \dots, x_n, z) = 0] \Leftrightarrow$$

$$(\forall y < z) (g(x_1, \dots, x_n, y) > 0) \ \& \ g(x_1, \dots, x_n, z) = 0$$

Една функция е **μ -рекурсивна**, ако се получава от основните с помощта на операциите суперпозиция, примитивна рекурсия и μ -операция, приложени краен брой пъти.

Пример: никъде недефинираната функция

$$\emptyset(x) = \mu z [S(x) = 0].$$

Теорема μ -рекурсивни функции съвпадат с изчислимите с МТ.



2.5 Функция на Акерман

[Акерман 1928, Хермес]

Function $a(x, y)$

if $x = 0$ then return $y + 1$

if $y = 0$ then return $a(x - 1, 1)$

return $a(x - 1, a(x, y - 1))$



Твърдение: a е тотална, МТ-изчислима функция

Д-во:

рекурсия $\rightsquigarrow$ стек с RAM $\rightsquigarrow$ МТ



Тоталност на функцията на Акерман

Д-во: Индукция по **лексикографската наредба** на (x, y) :

База: $a(0, y) = y + 1$

Индукционна стъпка за $y = 0$:

$$a(x, 0) = a(x - 1, 1),$$

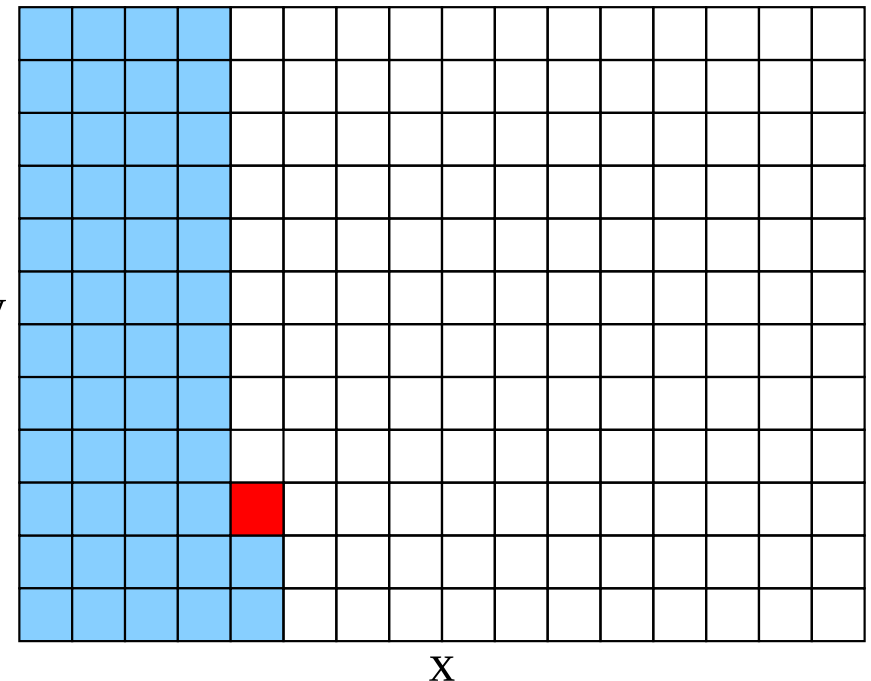
завършва по ИП $(x - 1, 1) < (x, 0)$

Индукционна стъпка за $x, y > 0$:

$a(x - 1, a(x, y - 1))$ завършва

$(x, y - 1) < (x, y)$ и

$(x - 1, a(x, y - 1)) < (x, y)$





Функцията на Акерман **не** е примитивно рекурсивна

Д-во: Да допуснем, че a е примитивно рекурсивна.

$\longrightarrow a(n, n) = g(n)$ е примитивно рекурсивна

Но ние ще покажем:

Лема Е: $\forall$ пр. р. $h : \exists k : \forall \bar{x} \in \mathbb{N}^n : h(\bar{x}) < a(k, \max(\bar{x}))$.
--

Тогава $(\exists k) a(n, n) < a(k, n)$ за всяко $n \Rightarrow a(k, k) < a(k, k)$.

Противоречие.



Пример

$$a(0, y) = y + 1$$

$$\begin{aligned} a(1, y) &= a(0, a(1, y - 1)) = a(1, y - 1) + 1 = \\ &= a(0, a(1, y - 2)) + 1 = a(1, y - 2) + 2 = \dots = \\ &= a(1, 0) + y = y + a(0, 1) = y + 2 \end{aligned}$$

$$\begin{aligned} a(2, y) &= a(1, a(2, y - 1)) = 2 + a(2, y - 1) = \dots \\ &= 2y + a(2, 0) = 2y + a(1, 1) = 2y + 3 \end{aligned}$$



Пример

$$a(\textcolor{red}{2}, y) = 2y + 3$$

$$a(\textcolor{red}{3}, y) = a(2, a(3, y - 1)) = 2a(3, y - 1) + 3$$

$$= 2a(2, a(3, y - 2)) + 3 = 4a(3, y - 2) + 3(1 + 2)$$

$$= 4a(2, a(3, y - 3) + 3(1 + 2)) = 8a(3, y - 3) + 3(1 + 2 + 4)$$

$$= \dots = 2^y \underbrace{a(3, 0)}_{=5} + 3 \underbrace{(1 + 2 + \dots + 2^{y-1})}_{=2^y - 1}$$

$$= \textcolor{red}{2^{y+3}} - 3$$



Пример

$$a(\textcolor{red}{3}, y) = 2^{y+3} - 3$$

$$a(\textcolor{red}{4}, y) = a(3, a(4, y-1)) = 2^{a(4, y-1)+3} - 3$$

$$= 2^{a(3, a(4, y-2))+3} - 3 = 2^{2^{a(4, y-2)+3}-3+3} - 3$$

$$= 2^{2^{a(3, a(4, y-3))+3}} - 3 = 2^{2^{2^{a(4, y-3)+3}-3+3}} - 3$$

$$= \dots = 2^{\overbrace{a(4, 0)}^{=a(3, 1)=2^{1+3}-3} + 3} - 3 = \textcolor{red}{2}^{\textcolor{red}{2^{16}}} - 3$$

$$a(4, 2) = 2^{2^{16}} - 3 = 2^{65536} - 3$$



Моноотонност на функцията на Акерман

Лема А: $y < a(x, y)$

Лема В: $a(x, y) < a(x, y + 1)$

Лема С: $a(x, y + 1) \leq a(x + 1, y)$

Лема D: $a(x, y) < a(x + 1, y)$

Лема BD: $a(x, y) \leq a(x', y')$ ако $x \leq x'$ и $y \leq y'$

Лема E: $a(x, 2y + 1) \leq a(x + 1, y)$

Д-во: Упражнение. Индукция,...



Лема Е: $\forall$ пр. рек. функция

$$h : \exists k : \forall n = \max(x_1, \dots, x_n) : h(x_1, \dots, x_n) < a(k, n).$$

Д-во: Индукция по дефиницията на пр. рек. функции.

База на индукцията:

$$O(n) = o < n + 1 = a(\mathbf{0}, n).$$

$$S(n) = n + 1 < 2n + 3 = a(\mathbf{2}, n).$$

$$I_i^k(x_1, \dots, x_k) = x_i < n + 1 = a(\mathbf{0}, n), \quad n = \max x_1, \dots, x_k.$$



Индукционна стъпка за $f = f_2 \cdot f_1$ (за простота),
 $n = \max(\bar{x})$:

По ИП $\exists k_1, k_2 : f_1(n) < a(k_1, n) \wedge f_2(m) < a(k_2, m)$.

Нека $k_3 = \max\{k_1 - 1, k_2\}$. Тогава:

$f(\bar{x}) \leq f_2(f_1(n))$	Деф. f
$< a(k_2, f_1(n))$	ИП
$< a(k_2, a(k_1, n))$	ИП, монотонност
$\leq a(k_3, a(k_3 + 1, n))$	монотонност
$= a(k_3 + 1, n + 1)$	Деф. a
$\leq a(k_3 + 2, n)$	Лема В



Индукционна стъпка за $f = g(f_1 \dots f_n)$, $n = \max(\bar{x})$:

По ИП

$\exists k, k_1 \dots k_n : g(\bar{x}) < a(k, n), f_1(\bar{x}) < a(k_1, n) \wedge f_2(\bar{x}) < a(k_n, n).$

Нека $m = \max\{k, k_1, \dots, k_n\}$. Тогава: $f_i(\bar{x}) < a(m+1, n).$

$$f(\bar{x}) \leq a(m, \max(f_1(\bar{x}), \dots, f_n(\bar{x}))) \quad \text{ИП } g$$

$$< a(m, a(m+1, n)) \quad \text{ИП}$$

$$= a(m+1, n+1) \quad \text{Деф. } a$$

$$\leq a(m+2, n) \quad \text{Лема В}$$



Индукционна стъпка за

$$h(\bar{x}, 0) = f(\bar{x}), \quad h(\bar{x}, y+1) = g(\bar{x}, y, h(\bar{x}, y)), \quad n = \max(\bar{x}):$$

По ИП $\exists k_1 : f(n) < a(k_1, n)$,

$\exists k_2 : g(n, y, z) < a(k_2, \max(n, y, z))$.

Нека $k = \max(k_1, k_2)$.

С индукция по $y : h(\bar{x}, y) \leq a(k+1, y + \max(\bar{x}))$:

$$h(\bar{x}, 0) = f(\bar{x}) \leq a(k+1, \max(\bar{x}))$$

$$h(\bar{x}, y+1) = g(\bar{x}, y, h(\bar{x}, y))$$

$$< a(k, \max(\bar{x}, y, h(\bar{x}, y)))$$

$$\leq a(k, (a(k+1, y + \max(\bar{x})))$$

$$\leq a(k+1, y+1 + \max(\bar{x})).$$



Индукционна стъпка за

$$h(\bar{x}, 0) = f(\bar{x}), \quad h(\bar{x}, y+1) = g(\bar{x}, y, h(\bar{x}, y)):$$

$$\begin{aligned} h(\bar{x}, y) &\leq a(k+1, y + \max(\bar{x})) \\ &\leq a(k+1, 2 \max(\bar{x}, y)) \\ &\leq a(k+1, 2 \max(\bar{x}, y) + 1) \\ &\leq a(k+2, \max(\bar{x}, y)) \end{aligned}$$



Busy Beaver

$\Sigma(n)$: $\max_{\delta} \#$ **единици**, които са на лентата след като ДМТ $(\{1, \dots, n, Z\}, \emptyset, \{0, 1\}, \delta, 1, \{Z\})$ завършва (празен вход).

$S(n)$: $\max_{\delta} \#$ **преходи**, които една ДМТ $(\{1, \dots, n, Z\}, \emptyset, \{0, 1\}, \delta, 1, \{Z\})$ прави преди да завърши (празен вход).



Busy Beaver

Да допуснем, че $S(n)$ е изчислима функция.

EvalS : MT, остойносттава $S(n)$

Clean изтрива 1-те на лентата

Double $n + n$

Double | **EvalS** | **Clean** - с n_0 състояния

Create n_0 - създава n_0 1-ци на празна лента

BadS : **Create** n_0 | **Double** | **EvalS** | **Clean** - $N = n_0 + n_0$ с.

От празна лента първо пише n_0 1-ци и след това ги удвоява, пишейки N 1-ци. Тогава **BadS** пише $S(N)$ 1-ци на лентата, и накрая изтрива 1'ците и завършва. Но изтриването прави поне $S(N)$ стъпки, така времето на **BadS** е $>$ от $S(N)$, противоречие с деф. на $S(n)$.



Busy Beaver

$\Sigma(n)$, $S(n)$ са тотални **НЕ ИЗЧИСЛИМИ** функции.

n	$\Sigma(n)$	$S(n)$
1	1	1
2	4	6
3	6	21
4	13	107
5	$\geq 4\,098$	$\geq 47\,176\,870$
6	$> 1.29 \cdot 10^{865}$	$> 3 \cdot 10^{1730}$

[<http://www.drb.insel.de/~heiner/BB/>],

[<http://www.logique.jussieu.fr/~michel/ha.html>]



Busy Beaver

n:	6	5	4	3	2	
q/in	0	1	0	1	0	1

A: B1R F0L; B1R C1L; B1R B1L; B1R Z1L; B1R B1L
 B: C0R D0R; C1R B1R; A1L C0L; B1L C0R; A1L Z1R
 C: D1L E1R; D1R E0L; Z1R D1L; C1L A1L;
 D: E0L D0L; A1L D1L; D1R A0R;
 E: A0R C1R; Z1R A0L; n=1:
 F: A1L Z1R; H1N ---



2.6 Проблемът за завършване, Неразрешимост, Сводимост

- ☐ Гьоделова номерация: МТ могат да оперират над себе си като вход
- ☐ Важен пример: Универсална МТ
- ☐ Диагонален метод: един неразрешим език
- ☐ Сводимост: показва, че и други проблеми са неразрешими.

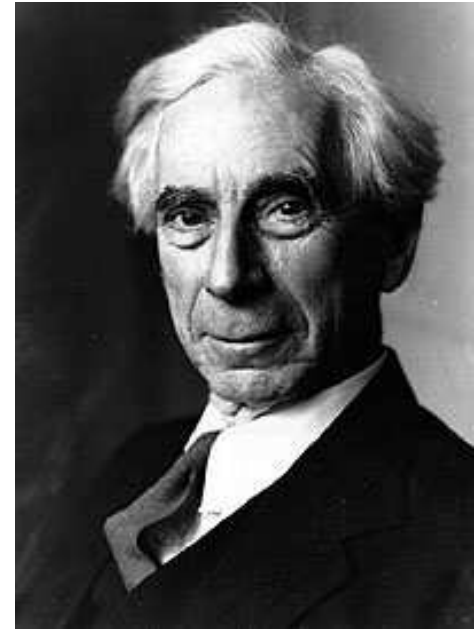


Парадокси и Self reference

Бръснарят в един малък град
бръсне тези и само тези хора,
които не се бръснят сами.

.

Кой бръсне бръснаря?





Парадокси и Self reference

Даниел Дйосентриб решил да изобрети една машина
КОЯТО ВСИЧКО ЗНАЕ.

Да Не

Задавате да/не въпрос и отговорът светва.

Дагоберт Дъг иска да купи машината.

Но е решил да провери дали работи правилно.

Задава въпрос на машината:

Ще ми отговориш ли с не?

Какво се случва?



Разрешимост

$A \subseteq \Sigma^*$ е (разрешимо), ако

характеристичната му функция χ_A е изчислима.

$$\chi_A(w) = \begin{cases} 1 & \text{ако } w \in A \\ 0 & \text{ако } w \notin A \end{cases}$$



Полуразрешимост

$A \subseteq \Sigma^*$ е **полу**разрешимо, ако

"**полу**" характеристичната му функция χ_A е изчислима.

$$\chi_A(w) = \begin{cases} 1 & \text{ако } w \in A \\ \perp & \text{ако } w \notin A \end{cases}$$



Свойства на разрешимите и полуразрешимите множества

Твърдение: $A \subseteq \Sigma^*$ разрешимо $\Leftrightarrow$
 A и $\bar{A}$ са полуразрешими

Д-во: Нека МТ

M_A полуразрешава A и

$M_{\bar{A}}$ полуразрешава $\bar{A}$

for $s := 1$ to ∞ do

if M_A завършва за s стъпки then Accept

if $M_{\bar{A}}$ завършва за s стъпки then Reject



Свойства на разрешимите множества

Твърдение: $A, B \subseteq \Sigma^*$ разрешими $\Rightarrow$

$A \cup B,$

$A \cap B,$

$A \setminus B,$

$A \cdot B,$

A^* са разрешими.

Примери : $\Sigma^*, \emptyset,$

Всяко крайно множество е разрешимо.



Свойства на полуразрешимите множества

Твърдение: $A, B \subseteq \Sigma^*$ полуразрешими $\Rightarrow$
 $A \cup B$ и $A \cap B$ са полуразрешими.

Нека M_1 и M_2 полуразрешават A и B

$A \cup B$: for $j := 1$ to ∞ do

 if M_1 приема w след j стъпки then Accept

 if M_2 приема w след j стъпки then Accept

$A \cap B$: for $j := 1$ to ∞ do

 if M_1 приема w след j ст. & M_2 приема w след j ст. then Accept



Рекурсивна номеруемост

$A \subseteq \Sigma^*$ **рекурсивно номеруемо**, ако

$A = \emptyset$ или $\exists$ тотална изчислима функция $f : \mathbb{N} \rightarrow \Sigma^*$:

$$A = \{f(1), f(2), f(3), \dots\}$$

Твърдение: A е рекурсивно номеруемо $\Leftrightarrow A$ е
полуразрешимо



Рекурсивна номеруемост $\longrightarrow$ полуразрешимост

Нека A е рекурсивно номеруемо с помощта на f .

Function $\chi'_A(x)$

for $n := 0$ to ∞ do

if $f(n) = x$ then return 1



Рекурсивна номеруемост $\longrightarrow$ полуразрешимост

- Нека $\pi(m, k) = 2^m(2k + 1) - 1$ - е кодираща функция на всички двойки естествени числа.
- Всяко естествено число n е код на точно една двойка числа $n = \pi(m, k)$.
- Нека $L(\pi(m, k)) = m$ и $R(\pi(m, k)) = k$ са декодиращите функции.
- π, L, R са изчислими функции.
- Да разгледаме редицата на всички думи в Σ^* :
 $\alpha_0, \alpha_1, \dots, \alpha_i, \dots$, наредени така: $|\alpha_i| < |\alpha_{i+1}|$ или
 $|\alpha_i| = |\alpha_{i+1}|$ и α_i е лексикографски по-малко от α_{i+1} .
- Например: $a, b, aa, ab, ba, bb, \dots$



Полуразрешимо $\longrightarrow$ рекурсивно номеруемо

Нека M е МТ полуразрешаваща A .

Ако $A = \emptyset$: тривиално.

Иначе дефинираме функция $f : \mathbb{N} \rightarrow \Sigma^*$ с област на стойностите A . Нека a е фиксиран елемент на A .

$$f(n) = \begin{cases} \alpha_{L(n)} & \text{ако } M(\alpha_{L(n)}) \downarrow \text{ за } R(n) \text{ стъпки,} \\ a & \text{иначе.} \end{cases}$$

Function $f(n)$

Интерпретираме n като код на двойката $n = \pi(m, k)$

Да разгледаме думата с номер m : $u = \alpha_m$

if M завършва при u за $\leq k$ стъпки then return u

else return a



Полуразрешимо $\longrightarrow$ рекурсивно номеруемо

- ☐ f е тотална
- ☐ f приема само стойности от A
- ☐ $\forall u \in A \exists k : M$ приема u за k стъпки
- ☐ $f(\pi(m, k)) = \alpha_m$



Задача: Да се докаже, че ако $A \subseteq \Sigma^*$ е безкрайно, то A е разрешимо тогава и само тогава, когато съществува тотална изчислима функция $f : \mathbb{N} \rightarrow \Sigma^*$:

$A = \{f(0) < f(1) < f(2) < \dots\}$ в лексикографски ред по големина.



Еквивалентни определения

$$A \subseteq \Sigma^*$$

- ☐ A е рекурсивно номеруемо
- ☐ A е полуразрешимо
- ☐ A е от Чомски тип 0
- ☐ $A = L(M)$ за МТ M
- ☐ χ'_A е Тюринг, МНР, RAM, ... изчислима
- ☐ A е дефиниционна област на една (частична) изчислима функция



2.7 Неразрешими Проблемы



Номерация на машините на Тюринг

Машината на Тюринг $M = (Q, \Sigma, \Gamma, \delta, s, F)$ наричаме стандартизирана, ако

- $\square Q = \{1, \dots, n\}$
- $\square \Sigma = \{0, 1\}$
- $\square \Gamma = \{0, 1, \sqcup\}, (\sqcup = 2)$
- $\square s = 1$
- $\square F = \{2\}$

за подходяща константа n



Гьоделов номер $\langle M \rangle$ на машина на Тюринг M

Дефинираме следните изрази в $\{0, 1\}$:

Кодиране $\delta(q, a) = (r, b, d)$ by $0^q 1 0^{a+1} 1 0^r 1 0^{b+1} 1 0^d$

където d е кодът на посоките: $N = 1, L = 2, R = 3$.

Машините на Тюринг ще кодираме с двоични числа:

$$111\text{code}_1 11\text{code}_2 11 \dots 11\text{code}_z 111,$$

code_i for $i = 1, \dots, z$: всички стойности на функцията δ .

Конвенция:

n не е Гьоделов номер на някоя МТ,

$\rightarrow n$ описва една МТ, която приема $\emptyset$



Гьоделова номерация

Наблюдение

Гьоделовата номерация дава едно
инективно изображение на **стандартизираните** МТ в
естествените числа



Пример

Нека $M = (\{1, 2, 3\}, \{0, 1\}, \{0, 1, \sqcup\}, \delta, 1, \{2\})$, с

$$\delta(1, 1) = (3, 0, R)$$

$$\delta(3, 0) = (1, 1, R)$$

$$\delta(3, 1) = (2, 0, R)$$

$$\delta(3, \sqcup) = (3, 1, L)$$

$\langle M \rangle$ е тогава:

11101001000101000110001010100100011000100100101000
1100010001000100100111



Универсална машина на Тюринг

$$U = (Q_u, \{0, 1\}, \{0, 1, \sqcup\}, \delta_u, s_u, F_u)$$

ВХОД: $\langle M \rangle w$

M е симулираната МТ, w е двоично кодираният вход.

U симулира M над w .

U приема $\langle M \rangle w$, ако M приема w



Универсална машина на Тюринг

3 ленти:

1. $\langle M \rangle$
2. кодът на състоянието q_M на M (унарно кодирано)
3. съдържанието на лентата за w на M



Универсална машина на Тюринг

```

if prefix  $v$  на  $w$  е код на една МТ then      // 111tuple111
    преместваме  $v$  на лентата  $\langle M \rangle$ 
     $q_M := 1$                                 // началното състояние на  $M$ 
    while  $q_M \neq 2$  do                        // крайно състояние на  $M$ 
        преместваме се до началото на  $\langle M \rangle$ 
        foreach  $(q, a, r, b, d) \in \langle M \rangle$  do // поле по поле
            if  $q = q_M$  then                    // сравняваме с  $q_M$ 
                if входният символ на лента 3 =  $a$  then
                     $q_M := r$                 // копираме на лента 2
                    слагаме  $b$  на лента 3
                    Движението по лента 3 е съгласно  $d$ 

```



Диагонален език L_d

Нека M_i е МТ с код $\langle M_i \rangle = i$.

Нека w_i двоичното представяне на i .

$L_d := \{w_i : M_i \text{ не приема (не завършва над) } w_i\}$



Твърдение: L_d не е полуразрешим

Д-во:

Да допуснем:

$L_d = \{w_i : M_i \text{ не приема (не завършва над) } w_i\}$ е полуразрешим.

Деф. "полуразрешим" $\rightarrow \exists M_i : M_i \text{ приема } L_d$.

Как работи M_i над w_i ?

$w_i \in L_d \xrightarrow{\text{Деф. } M_i} w_i \text{ ще се приеме от } M_i. \xrightarrow{\text{Деф. } L_d} w_i \notin L_d$

$w_i \notin L_d \xrightarrow{\text{Деф. } M_i} w_i \text{ няма да се приеме от } M_i. \xrightarrow{\text{Деф. } L_d} w_i \in L_d$

И в двата случая имаме **противоречие**.



Следствие:

$\bar{L}_d = \{w_i : M_i \text{ приема } w_i\}$ е неразрешим

Забележете, че $\bar{L}_d$ е полуразрешим:

изпълняваме универсалната машина над $\langle M_i \rangle w_i$.

Да допуснем: $\bar{L}_d$ е разрешим.

$\rightarrow \exists M : M$ разпознава $\bar{L}_d$

Променяме $M \rightsquigarrow M'$ така M' разпознава L_d

(Заменяме: приема/не приема за заключителните състояния).

Противоречие.



Следствие:

$\overline{L_d}$ е полуразрешим и не е разрешим.

Полуразрешимите езици не са затворени относно допълнение.



Неразрешими проблеми

Не съществува програма P , такава че

$$\text{halts}(\langle P \rangle, X) = \begin{cases} \text{yes} & \text{ако } P(X) \text{ завършва} \\ \text{no} & \text{иначе} \end{cases}$$

Ако такава програма съществува, то:

$D(X) = \text{if halts}(X, X) \text{ then loop}(X) \text{ else halt}$

$D(\langle D \rangle) = \text{if halts}(\langle D \rangle, \langle D \rangle) \text{ then loop}(\langle D \rangle) \text{ else halt}$

Ако $\text{halts}(\langle D \rangle, \langle D \rangle) = \text{yes}$, то $\downarrow D(\langle D \rangle)$, но $\uparrow D(\langle D \rangle)$.

Ако $\text{halts}(\langle D \rangle, \langle D \rangle) = \text{no}$, то $\uparrow D(\langle D \rangle)$, но $\downarrow D(\langle D \rangle)$.



Неразрешими проблеми

Не съществува програма **is-safe**, която не е вирус и:

$$\text{is-safe}(\langle P \rangle, X) = \begin{cases} \text{yes} & \text{ако } P(X) \text{ не пуска вирус} \\ \text{no} & \text{иначе} \end{cases}$$

$D(X) = \text{if is-safe}(X, X) \text{ then virus}(X) \text{ else print "Hello"}$

$D(\langle D \rangle) = \text{if is-safe}(\langle D \rangle, \langle D \rangle) \text{ then virus}(\langle D \rangle) \text{ else print "Hello"}$

Ако $\text{is-safe}(\langle D \rangle, \langle D \rangle) = \text{yes}$, то $D(\langle D \rangle)$ не е активирала вирус, но $\text{virus}(\langle D \rangle)$ е пуснат.

Ако $\text{is-safe}(\langle D \rangle, \langle D \rangle) = \text{no}$, то вирус е активиран $\Rightarrow$ **is-safe** е пуснала вирус.



Проблемът за завършване (Стоп проблемът)

$H := \{w_i v : M_i \text{ завършва при } v\}$

Твърдение: H не е разрешим.

Д-во: Да допуснем, че H е разрешим.

Тогава има МТ M_i , която разпознава H .

$w_i \in \bar{L}_d$?

$\Leftrightarrow M_i$ приема w_i .

$\Leftrightarrow w_i w_i \in H$.

Така $\bar{L}_d$ също излиза разрешим. Противоречие.

Защо L_d се нарича диагонален?

$(u, v) \in R \Leftrightarrow u = \langle M \rangle$ (МТ) и M приема v

$R_u = \{v \mid (u, v) \in R\}$

$D = \{u \mid (u, u) \notin R\}$. Следователно $D \neq R_u$ за всяко u .



Ограниченият проблем за завършване

Твърдение:

$\{w_i v \# w_j : M_i \text{ завършва при } v \text{ за най-много } j \text{ стъпки}\}$
е разрешимо.

Д-во:

Пускаме универсалната МТ U да работи над $w_i v$
 j симулиращи стъпки.



Други неразрешими проблеми

Дадено: машини на Тюринг T , T'

☐ $L(T) = \emptyset?$

празнота

☐ $|L(T)| = \infty?$

безкрайност

☐ $L(T) = \Sigma^*?$

ПЪЛНОТА

☐ $L(T) = L(T')?$

ЕКВИВАЛЕНТНОСТ



Неразрешимост на празнотата

Да допуснем, че $\{i : L(M_i) = \emptyset\}$ е разрешим.

Ще покажем, че $\bar{L}_d$ ще излезе разрешим.

$\bar{L}_d = \{w_i : M_i \text{ приема } w_i\}$?

Конструираме машина на Тюринг $T(i)$:

изтрий входа

изпълни M_i над w_i

if state(M_i) $\neq 2$ then безкраен цикъл

Така $L(T(i)) \neq \emptyset$, ако и само ако $w_i \in \bar{L}_d$.

Следователно $\bar{L}_d$ е разрешим.

Противоречие.



Неразрешимост на пълнотата

$$L(T) = \Sigma^*?$$

Същото доказателство като празнотата! Тук $T(i)$ игнорира входа си!

$$L(T(i)) = \Sigma^*, \text{ ако и само ако } w_i \in \bar{L}_d.$$



Метапрограмиране

Доказателството на празнотата взима една програма и я трансформира в друга.

Важна техника за програмиране.

Деф. Нека L_1 и L_2 са езици в Σ^* . Казваме, че $L_1 \leq L_2$ (L_1 се свежда до L_2), ако има тотална изчислима функция f за която

$$u \in L_1 \Leftrightarrow f(u) \in L_2.$$

Ако L_2 е разрешим (полуразрешим), то L_1 е разрешим (полуразрешим).

Ако L_1 не е разрешим (полуразрешим), то L_2 не е разрешим (полуразрешим).



Други неразрешими проблеми

Твърдение. Следните проблеми не са разрешими:

1. По дадена МТ M дали M завършва над празна лента— $\{w_i \mid M_i \text{ halts on } \varepsilon\}$.

Дво 1. $E = \{w_i \mid M_i \text{ halts on } \varepsilon\}$ не е разрешим.

Идея: Ще построим изчислима функция f :

$$\langle M \rangle v \in H \Leftrightarrow f(\langle M \rangle, v) \in E.$$

Конструираме машина на Тюринг $T_{\langle M \rangle v}$:

пише v на празната лента

изпълнява M над v

Така f е функцията, която по дадени $\langle M \rangle, v$ дава кода на $T_{\langle M \rangle v}$.



2. По дадена МТ M дали M завършва над всеки вход.

— $A = \{w_i \mid \forall v (M_i \text{ halts on } v)\}$.

Д-во: $\langle M \rangle \in E \Leftrightarrow f(\langle M \rangle) \in A$.

Конструираме машина на Тюринг $T_{\langle M \rangle}$:

изтрива входа

изпълнява M над празна лента

Така f е функцията, която по дадено $\langle M \rangle$, дава кода на МТ $T_{\langle M \rangle}$.



3. По дадени МТ M_1 и M_2 дали те завършват над един и същи вход.

$$B = \{w_i w_j \mid \forall v (\downarrow M_i(v) \iff \downarrow M_j(v))\}.$$

Д-во: Нека y е код на МТ, която завършва при всеки вход- например идентитета.

Дефинираме $f(w_i) = w_i y$. Тогава:

$$\langle M \rangle \in A \iff f(\langle M \rangle) \in B.$$



Теорема на Райс-Успенски

Нека $\mathbf{R}$ е класът от изчислимите функции.

Теорема Нека $\mathbf{S}$ е не тривиален клас от изчислими функции. $(S \neq \emptyset, S \neq R)$. Тогава множеството

$$C(S) = \{w_i \mid M_i \text{ изчислява функция } \in S\}$$

не е разрешим.

Д-во:

Да допуснем, че $C(S)$ е разрешим.

Случай 1. $\emptyset \notin S$ и $f \in S$. Тогава има машина на Тюринг M_f която изчислява f .



Нека M е МТ и w е дума.

$$T_{M,w}(x) = \begin{cases} M_f(x) & \text{if } \downarrow M(w) \\ \perp & \text{иначе} \end{cases}$$

Тогава:

ако $\downarrow M(w) \Rightarrow (\forall x) T_{M,w}(x) = f(x)$

ако $\uparrow M(w) \Rightarrow T_{M,w}(x) = \emptyset$.

$$T_{M,w} \in S \Leftrightarrow \downarrow M(w).$$

$$\langle T_{M,w} \rangle \in C(S) \Leftrightarrow \langle M \rangle w \in H.$$

Противоречие.

Случай 2. $\emptyset \in S$. Разглеждаме: $R \setminus S$ не е тривиален.

Тогава $C(S)$ не е разрешимо, и следователно $C(S)$ е неразрешимо.



Теорема на Райс-Успенски

Нека $\mathbf{R}$ е класът на всички полуразрешими езици в Σ^* .

Следствие Ако $\mathbf{S}$ е нетривиален клас от полуразрешими множества в Σ^* ($S \neq \emptyset, S \neq R$). Тогава множеството

$$C(SL) = \{w_i \mid M_i \text{ приема (полуразрешава) множество от } \in S\}$$

не е разрешимо.

Д-во:

Да допуснем, че е. Тогава

$$C(S) = \{w_i \mid M_i \text{ изчислява функция с } dom \text{ от } \in S\}$$

е разрешимо. Противоречие.



Self-reference

Да построим МТ, която при всеки вход, изтрива входа и печата собствения си описане (код). Тази машина наричаме **SELF**.

Лема : Съществува изчислима функция $q(w) = \langle P_w \rangle$ -кодът на P_w и завършва, където $\forall x P_w(x) = w$.

(а) За всеки вход w построяваме сл. машина

P_w :

1. Изтрива входа.
2. Пише w на лентата.
3. Завършва.

(b) Изход : $\langle P_w \rangle$.



Self-reference

Конструкция на **SELF**: $\langle SELF \rangle = \langle AB \rangle$.

Работата на A : печатай $\langle B \rangle$

Работата на B : печатай $\langle AB \rangle$

$A = P_{\langle B \rangle} [\Rightarrow q(\langle B \rangle) = \langle A \rangle]$.

$B = :$ При вход $\langle M \rangle$:

1. Пресмята $q(\langle M \rangle) = \langle M' \rangle$.
2. Комбинираща резултата с $\langle M \rangle$ за да получи $\langle M'M \rangle$.
3. Печата и завършва.

Тогава **SELF**:

1. $A : \langle B \rangle$;
2. $B : q(\langle B \rangle) = \langle A \rangle$;
3. $B : \langle A \rangle \langle B \rangle \rightsquigarrow \langle AB \rangle$



Теорема за рекурсията

Теорема. Нека $t : \Sigma^* \times \Sigma^* \rightarrow \Sigma^*$ е изчислима функция.

Съществува МТ R , която изчислява функцията $r(w) = t(\langle R \rangle, w)$.

Например $t(a, w) = a$. $\exists R(R(w) = \langle R \rangle)$.

Д-во: Нека T е МТ изчисляваща функцията t .

Построяваме МТ $\langle R \rangle = \langle ABT \rangle$:

$A: P_{\langle BT \rangle} (\langle A \rangle = q(\langle BT \rangle))$.

$B: input \langle BT \rangle$;

$q(\langle BT \rangle) = \langle A \rangle \Rightarrow (\langle ABT \rangle, w)$.

$R(\langle ABT \rangle, w) = r(w)$.



Втора теорема за рекурсията

Следствие. Нека $t : N \rightarrow N$ е тотална изчислима функция
и за всяко $w \in \Sigma^*$

$$\theta(a, w) = U(t(a), w).$$

Съществува МТ R , за която

$$R(w) = \theta(\langle R \rangle, w) = U(t(\langle R \rangle), w) = t(\langle R \rangle)(w).$$



Post Correspondence Problem (PCP)

Дадено: карйни редици от двойки думи

$$K = (x_1, y_1) \cdots (x_n, y_n) \in (\Sigma^+ \times \Sigma^+)^*$$

Въпрос:

$$\exists i_1, \dots, i_k \in \{1, \dots, n\} : x_{i_1} \cdots x_{i_k} = y_{i_1} \cdots y_{i_k}$$

?



Пример

□ $K = ((1, 11), (10111, 10), (10, 0))$ има решение
(2, 1, 1, 3), имаме:

$$x_2 x_1 x_1 x_3 = 101111110 = 101111110 = y_2 y_1 y_1 y_3$$

□ $K = ((10, 101), (011, 11), (101, 011))$

няма решение:

(133...)



Пример [Mirko Rahn]

□ $K = ((0, 011), (001, 1), (1, 00), (11, 110))$

има на-кисо решение с дължина 595:

12112121121121211212032121121303212033112131112120312121211213121
03212112103212130321202111120331121321212121311121211211112032031
21203212112121212131312032130321203203210312130331121313021032011
12121121112002101212121212032121121212120212032032130321211203213
31303303212130303121131130320010321211121211312123032121203212103
01103213032302121230331011203131021212131210203203120213131213211
10321111212120211112121212032132121212112032130311203211212130331
21311212112033103203121203212131021010321303210202123033021321011
30212122113203121210103202123132110311212312120303213303003



PCP е полуразрешим

Алгоритъм:

```
Procedure PCP( $(x_1, y_1) \cdots (x_n, y_n)$ )  
  for  $k := 1$  to  $\infty$  do  
    foreach  $i_1 \cdots i_k \in \{1..n\}^k$  do  
      if  $x_{i_1} \cdots x_{i_k} = y_{i_1} \cdots y_{i_k}$  then  
        output  $i_1 \cdots i_k$   
  return
```



РСР е неразрешим

Идея: допускаме, че е разрешим $\rightarrow$ проблемът за завършване става разрешим

$$x_{i_1} \dots x_{i_k} = y_{i_1} \dots y_{i_k} = (s)w\# \dots \# u(f)v$$

описва една приемаща последователност от МТ-конфигурации



10. проблем на Хилберт — Диофантови уравнения

Дадено:

полином на **на много променливи p**
с цели коефициенти.

Въпрос [Hilbert 1900]:



$$\exists x_1, \dots, x_n \in \mathbb{Z} : p(x_1, \dots, x_n) = 0?$$

[Matiyasevich 1970]: Проблемът е неразрешим.



Затвореност на разрешимите езици

Затворени относно

$\cup$

$\cap$

$\cdot$



Затвореност на **полу**разрешимите езици

Затворени относно

$\cup$

$\cap$

Не са затворени относно

$\bar{}$



Незатвореност на полуразрешимите езици относно допълнение

Да допуснем : че са затворени относно допълнение.

Нека M полуразрешава L_d , $\bar{M}$ полуразрешава $\bar{L}_d$

Function isInLd(w)

for $j := 1$ to ∞ do

if M приема w след j стъпки then return true

if $\bar{M}$ приема w след j стъпки then return false

Поне едната от двете завършва.

$\rightarrow L_d$ разрешимо.

Противоречие.