Глава 8

Машини на Тюринг

8.1 Еднолентови машини на Тюринг

Машините на Тюринг са абстрактни изчислителни устройства, чиято цел е да могат да симулират алгоритмичните формални изчисления, извършвани от човек. Всяка машина на Тюринг разполага с неограничено количество памет във формата на безкрайна лента (разделена на клетки), четяща глава и крайна програма. Всяка клетка на лентата е или празна или съдържа един символ от от фиксирана азбука. Машината работи на тактове (равни интервали от време). На всеки такт от работата на машината само краен брой клетки от лентата не са празни. Четящата глава на машината разглежда точно една клетка от лентата, като разпознава дали клетка е празна или в нея има разположен символ от азбуката. За да опростим разглежданията ще считаме, че цялата лента е запълнена със символи от крайна азбука Γ, съдържаща съмивол $\sqcup$, маркиращ празна клетка, като във всеки един момент от изпълнението само краен брой клетки съдържат символ, различен от $\sqcup$. *Програмата* на машината на Тюринг е организирана с помощта на *състояния*, като на всеки такт действието на машината се определя еднозначно от състоянието, в което се намира, и това което вижда четящата глава. На всеки такт машината извършва точно едно от следните три действия:

- записва дадена буква в клетката, над която се намира четящата глава;
- мести четящата глава наляво;
- мести четящата глава надясно.

За допълнително улеснение ще считаме, че лентата е неограничена само надясно, т.е. има ляв край. Този ляв край на лентата ще бележим със символа $\triangleright$, като считаме, че той стои в най-лявата клетка на лентата.

По-формално, *машина на Тюринг* наричаме всяка наредена петорка $\mathcal{M} = (K, \Gamma, \delta, s, H)$, където

- K е крайно множество (състояния на машината);
- Γ е крайна азбука, съдържаща $\sqcup$, но *не* съдържаща символите $\triangleright$, $\rightarrow$ и $\leftarrow$;
- δ е изображение от $(K - H) \times (\Gamma \cup \{\triangleright\})$ в $K \times (\Gamma \cup \{\rightarrow, \leftarrow\})$;
- за всеки две състояния q и q', ако $\delta(q, \triangleright) = (q', y)$, то y е $\rightarrow$;
- $s \in K$ — начално състояние на $\mathcal{M}$;
- $H \subseteq K$ — множество от „стоп“ състояния.

Изображението δ е дефинирано за всяка двойка (q, x), дъдето $q \in K$ не е „стоп“ състояние, а x и символ, който може да бъде разположен на върху лентата на машината, като то изобразява (q, x) в (q', x'), където q' може да бъде произволно състояние, а смисълът на x' е следния: ако x е от Γ, то машината записва x в клетката, над която се намира четящата глава; ако x е $\rightarrow$, то машината премества четящата глава надясно; ако x е $\leftarrow$, то машината премества четящата глава наляво. При това, ако четящата глава чете символа $\triangleright$, единственото, което може да направи машината, е да премести четящата глава надясно. В случай, че машината достигне до състояние от H, то машината спира изпълнението.

Всяко изпълнение на машината се определя еднозначно от състоянието, в което машината се намира, символите, записани на лентата, и позицията на четящата глава. Тъй като нашата лента е ограничена отляво чрез символа $\triangleright$ и тя може да съдържа само краен брой символи, различни от $\sqcup$, то достатъчно е да знаем символите, които са записани между символа $\triangleright$ и последния символ, различен от $\sqcup$, или позицията на четящата глава, в случай че тя

се намира вдясно от последната непразна клетка. Тази информация представлява крайна дума σ, започваща с $\triangleright$. Позицията на четящата глава ще отбелязваме, подчертавайки символа от σ, който тя вижда в момента.

Конфигурация ще наричаме всяка двойка (q, σ), където q е състояние, а σ е дума от вида $\triangleright u$, $u \in \Gamma^*$, в която точно един символ е подчертан, като в случай че последния символ е $\sqcup$, то именно той е подчертаният. където $u \in \Gamma^*$, а v е или празна, или завършва на символ различен от $\sqcup$. Ако $q \notin H$, ще казваме че конфигурацията е нефинална, а ако $q \in H$, че конфигурацията е финална.

Всяка машина на тюринг $\mathcal{M}$ дефинира релацията $\vdash_{\mathcal{M}}$ (трансформация на конфигурации) между конфигурации по следния начин: Нека имаме нефинална конфигурация $(q, u\underline{x}v)$. Нека $\delta(q, x) = (q', x')$. Тогава

1. Ако $x' \in \Gamma$, то
$$(q, u\underline{x}v) \vdash_{\mathcal{M}} (q', u\underline{x'}v).$$
2. Ако x' е $\leftarrow$ и u е с най-десен символ y, т.е. $u = u'y$, то
$$(q, u'y\underline{x}v) \vdash_{\mathcal{M}} (q', u'\underline{y}xv).$$
3. Ако x' е $\rightarrow$ и v е непразна дума с най-ляв символ y, т.е. $v = yv'$, то
$$(q, u\underline{x}yv') \vdash_{\mathcal{M}} (q', ux\underline{y}v').$$
4. Ако $x' = \rightarrow$ и v е празната дума, то
$$(q, u\underline{x}) \vdash_{\mathcal{M}} (q', ux\underline{\sqcup}).$$

Тъй като δ е функция, то за всяка нефинална конфигурация (q, σ) съществува единствена конфигурация (q', σ'), такава че
$$(q, \sigma) \vdash_{\mathcal{M}} (q, \sigma').$$

С $\vdash^n_{\mathcal{M}}$ ще означаваме резултата от $n \geq 0$ последователни прилагания на релацията $\vdash_{\mathcal{M}}$. С други думи за две конфигурации C и C' е в сила $C \vdash^n_{\mathcal{M}} C'$ тогава и само тогава,
$$C \vdash_{\mathcal{M}} C_1 \vdash_{\mathcal{M}} \cdots \vdash_{\mathcal{M}} C_{n-1} \vdash_{\mathcal{M}} C'$$
като при $n = 0$ имаме
$$C \vdash^0_{\mathcal{M}} C' \iff C = C'.$$

С $\vdash^*_{\mathcal{M}}$ ще означаваме рефлексивното и транзитивно затваряне на релацията $\vdash_{\mathcal{M}}$. С други думи
$$C \vdash^*_{\mathcal{M}} C' \iff C \vdash^n_{\mathcal{M}} C' \text{ за някое } n \geq 0.$$

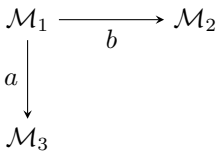
Пример. Да разгледаме машината $\mathcal{M}$, която има азбука $\{0, 1, \sqcup\}$, състояния $\{s, q, f\}$, като $H = \{f\}$, и преходи

$$\begin{aligned}
\delta(s, 0) &= (q, 1)\\
\delta(s, 1) &= (q, 0)\\
\delta(s, \sqcup) &= (f, \sqcup)\\
\delta(s, \triangleright) &= (s, \rightarrow)\\
\delta(q, x) &= (s, \rightarrow) \text{ за } x \in \{0, 1, \sqcup, \triangleright\}
\end{aligned}$$

Машината $\mathcal{M}$ преборазува конфигурацията $\triangleright 0s110$ по следния начин

$$\begin{aligned}
(s, \triangleright\underline{0}110) \quad &\vdash_{\mathcal{M}} \quad (q \triangleright \underline{1}110)\\
&\vdash_{\mathcal{M}} \quad (s, \triangleright 1\underline{1}10)\\
&\vdash_M c \quad (q, \triangleright 1\underline{0}10)\\
&\vdash_M c \quad (s, \triangleright 10\underline{1}0)\\
&\vdash_M c \quad (q, \triangleright 100\underline{0})\\
&\vdash_M c \quad (s, \triangleright 1000\underline{0})\\
&\vdash_M c \quad (q, \triangleright 1001\underline{1})\\
&\vdash_M c \quad (s, \triangleright 1001\underline{\sqcup})\\
&\vdash_M c \quad (f, \triangleright 1001\underline{\sqcup})
\end{aligned}$$

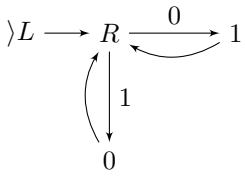
Пример. Елементарните машини на Тюринг над дадена фиксирана азбука Γ дефинираме по следния начин: за всяко $z \in \Gamma \cup \{\leftarrow, \rightarrow\}$ с $\mathcal{M}_z$ ще означаваме машината със две състояния, да речем s и h, като s е начално, а h е „стоп“ състояние, а преходите са $\delta(s, x) = (h, z)$ за всяко $x \in \Gamma$ и $\delta(s, \triangleright) = (s, \rightarrow)$. Машината $\mathcal{M}_a$ за $a \in \Gamma$ записва символа a върху клетката, която разглежда четящата глава. Тези машини ще означаваме с буквата, която записват. Машината $\mathcal{M}_\leftarrow$ премества четящата глава наляво и спира, а машината $\mathcal{M}_\rightarrow$ премества четящата глава надясно и спира. Тези машини ще означаваме съответно с L и R. Последователното изпълнение на елементарни машини ще записваме като дума над азбуката $\Gamma \cup \{L, R\}$. Така например с $RaRR \sqcup L$ ще означаваме машината, която премества четящата надясно, пише буквата a, премества четящата глава две клетки надясно, изтрива съдържанието на клетката и премества главата наляво. Освен чрез последователно изпълнение можем да свързваме и машини и с условни преходи зависещи от символа, който вижда четящата глава. Така например



изпълнява първо $\mathcal{M}_1$, като след като тя приключи, ако четящата глава вижда буквата a, то започваме да изпълняваме $\mathcal{M}_3$, а ако вижда b изпълняваме $\mathcal{M}_2$. Тази схема можем да реализираме по следния естествен начин: Нека $\mathcal{M}_i = (K_i, \Gamma, \delta_i, s_i, H_i)$ за $i = 1, 2, 3$, като K_1, K_2 и K_3 нямат общи елементи. На схемата описана по-горе съответвства машина $\mathcal{M} = (K, \Gamma, \delta, s, H)$, където

$$\begin{aligned}
K &= K_1 \cup K_2 \cup K_3 \cup \{h\}, \text{ където } h \notin K_1 \cup K_2 \cup K_3 \\
\delta &= \delta_1 \cup \delta_2 \cup \delta_3 \cup \\
&\quad \{((h_1, a), (s_3, a) \mid h_1 \in H_1\} \cup \\
&\quad \{((h_1, b), (s_2, b) \mid h_1 \in H_1\} \cup \\
&\quad \{((h_1, x), (h, x) \mid h_1 \in H_1,\ x \neq a, b\} \\
s &= s_1 \\
H &= H_2 \cup H_3 \cup \{h\}.
\end{aligned}$$

Машината от предния пример можем да опишем по следния начин



където в случай, че дадена стрелка няма етикет, то това означава, че правим този преход независимо от символа, който вижда четящата глава.

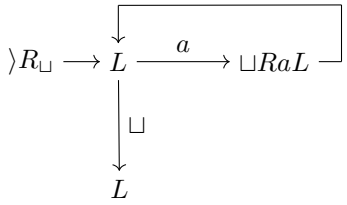
Пример. Машините

където преход с етикет $\overline{a}$ се извършват по всички символи, различни от a, преместват четящата глава съответно надясно и наляво, докато не срещнат символа a. Ще отбелязваме тези машини с R_a и L_a.

Машините

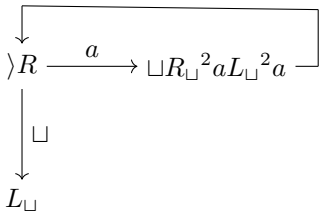
предвижват четящата глава съответно надясно и наляво, докато не достигнат до символ, различен от a.

Пример. Машината $S_{\rightarrow}$



преобразува $\sqcup w \sqcup$, където w е непразна дума, несъдържаща $\sqcup$, в $\sqcup \sqcup w \sqcup$, т.е. премества w с една позиция надясно.

Пример. Машината C



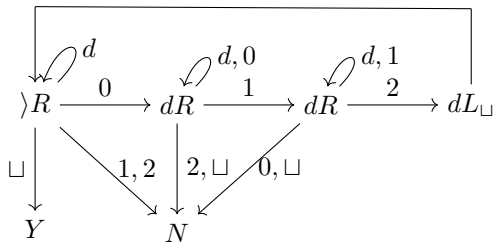
преобразува $\sqcup w \sqcup$, където w е непразна дума, несъдържаща $\sqcup$, в $\sqcup w \sqcup w \sqcup$, т.е. прави копие на w отдясно на w.

8.2 Разрешими езици. Изчислими функции.

Ще казваме, че машината на Тюринг $\mathcal{M} = (K, \Gamma, \delta, s, H)$ е *разпознавател*, ако $H = \{y, n\}$. Финална конфигурация от вида (y, σ) ще наричаме *приемаща*, а такава от вида (n, σ) — *отхвърляща*.

Нека Σ е крайна азбука, несъдържаща $\sqcup$. Ще казваме, че $L \subseteq \Sigma^*$ е *разрешим*, ако съществува разпознавател $\mathcal{M}$, такъв че за всяко $w \in \Sigma^*$, $\mathcal{M}$ преобразува $(s, \triangleright\underline{\sqcup}w)$ до приемаща конфигурация, в случай че $w \in L$, и до неприемаща конфигурация, в случай че $w \notin L$ (тук s е началното състояние на $\mathcal{M}$). С други думи, L е разрешим, ако съществува машина на Тюринг, която започвайки от конфигурацията $(s, \triangleright\underline{\sqcup}w)$ задължително в даден момент спира изпълнението си и или приема думата w в случай, че $w \in L$, или отхвърля w в случай, че $w \notin L$.

Пример. Нека с Y и N означим машините на Тюринг, чиято единствена работа да преминат съответно в заключително състояние y и заключително състояние n. Езикът $L = \{0^n1^n2^n \mid n \geq 0\}$ се разрешава от следния разпознавател

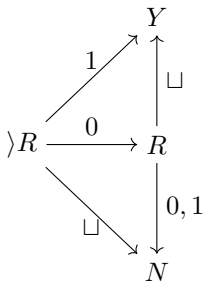


За да илюстрираме, как работи разпознавателят нека разгледаме думата 001122.

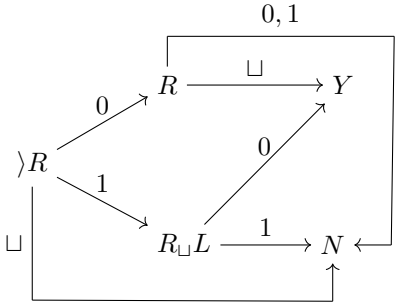
$$\begin{aligned}
\triangleright\underline{\sqcup}001122 &\vdash \triangleright\sqcup\underline{0}01122 \vdash \triangleright\sqcup\underline{d}01122 \vdash \triangleright\sqcup d\underline{0}1122 \vdash \triangleright\sqcup d0\underline{1}122 \vdash \triangleright\sqcup d0\underline{d}122\\
&\vdash \triangleright\sqcup d0d\underline{1}22 \vdash \triangleright\sqcup d0d1\underline{2}2 \vdash \triangleright\sqcup d0d1\underline{d}2 \vdash \triangleright\sqcup d0d1\underline{d}2 \vdash \triangleright\sqcup d0\underline{d}1d2\\
&\vdash \triangleright\sqcup d\underline{0}d1d2 \vdash \triangleright\sqcup\underline{d}0d1d2 \vdash \triangleright\underline{\sqcup}d0d1d2\\
&\vdash \triangleright\sqcup\underline{d}0d1d2 \vdash \triangleright\sqcup d\underline{0}d1d2 \vdash \triangleright\sqcup d\underline{d}d1d2 \vdash \triangleright\sqcup dd\underline{d}1d2 \vdash \triangleright\sqcup ddd\underline{1}d2\\
&\vdash \triangleright\sqcup dddd\underline{d}2 \vdash \triangleright\sqcup ddddd\underline{2} \vdash \triangleright\sqcup dddddd\underline{d} \vdash \triangleright\sqcup ddddd\underline{d}d\\
&\vdash \triangleright\sqcup dddd\underline{d}dd \vdash \triangleright\sqcup ddd\underline{d}ddd \vdash \triangleright\sqcup dd\underline{d}dddd \vdash \triangleright\sqcup d\underline{d}ddddd \vdash \triangleright\underline{\sqcup}dddddd\\
&\vdash \triangleright\sqcup\underline{d}dddddd \vdash \triangleright\sqcup d\underline{d}ddddd \vdash \triangleright\sqcup dd\underline{d}dddd \vdash \triangleright\sqcup ddd\underline{d}ddd \vdash \triangleright\sqcup dddd\underline{d}dd\\
&\vdash \triangleright\sqcup ddddd\underline{d} \vdash \triangleright\sqcup dddddd\underline{\sqcup} \vdash y
\end{aligned}$$

За всяко естествено число n с $B(n)$ ще означаваме двоичния запис на n. Да отбележим, че $B(n)$ е непразна дума над азбуката $\{0, 1\}^*$. Ще казваме, че множеството $A \subseteq \mathbb{N}$ е *изчислимо*, ако езикът $\{B(n) \mid n \in A\}$ е разрешим.

Пример. Множеството $\mathbb{N}$ е изчислимо, като то се разпознава от машината $\mathcal{M}_{\mathbb{N}}$



Множеството от четните числа също е разрешимо, като то се разпознава от машината



Теорема 8.1. Разрешимите езици са затворени относно операциите обединение, сечение и допълнение.

Доказателство. Нека за всяка машина $\mathcal{M} = (K, \Gamma, s, H)$, която не използва символа $\#$, с $\mathcal{M}^{\#}$ означим машината $\mathcal{M}^{\#} = (K', \Gamma \cup \{\#\}\delta', H\}$, където

$$K' = K \cup \{p'_q \mid q \in K\} \cup \{p''_q \mid q \in K\}$$

и

$$\begin{aligned}
\delta'(q,x) &= \delta(q,a) \text{ за } q \in K \text{ и } x \neq \# \\
\delta'(q,\#) &= (p'_q, \sqcup) \text{ за } q \in K \\
\delta'(p'_q, \sqcup) &= (p''_q, \rightarrow) \\
\delta'(p'_q, x) &= (p'_q, x) \text{ за } x \neq \sqcup \\
\delta'(p''_q, \sqcup) &= (p''_q, \#) \\
\delta'(p''_q, \#) &= (q, \leftarrow) \\
\delta'(p''_q, x) &= (p''_q, x) \text{ за } x \neq \sqcup, \#
\end{aligned}$$

Ролята на състоянията p'_q и p''_q е в случай, че машината достигне символа $\#$ в състояние $q \in K$, то тя да премести $\#$ една клетка надясно и да върне машината обратно към клетката, в която е бил $\#$ и да бъде отново в състояние q. По точно, за всяко $q \in K$ имаме

$$(q, \sigma\underline{\#}) \vdash_{\mathcal{M}^{\#}} (p'_q, \sigma\underline{\sqcup}) \vdash_{\mathcal{M}^{\#}} (p''_q, \sigma \sqcup \underline{\sqcup}) \vdash_{\mathcal{M}^{\#}} (p''_q, \sigma \sqcup \underline{\#}) \vdash_{\mathcal{M}^{\#}} (q, \sigma\underline{\sqcup}\#)$$

От друга страна, за всяка конфигурация (q, σ) на $\mathcal{M}$, ако $(q, \sigma) \vdash_{\mathcal{M}} (q', \sigma')$ и $\sigma' \neq \sigma\underline{\sqcup}$, то

$$(q, \sigma\#) \vdash_{\mathcal{M}} (q', \sigma'\#),$$

а ако $\sigma' = \sigma\underline{\sqcup}$, то

$$(q, \sigma\#) \vdash_{\mathcal{M}^{\#}} (q', \sigma\underline{\#}),$$

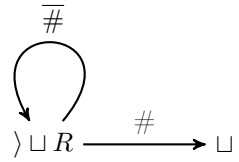
Оттук, за всяко дума w имаме

$$(q, \sigma) \vdash^*_{\mathcal{M}} (q', \sigma') \iff (q, \sigma\#) \vdash^*_{\mathcal{M}^{\#}} (q', \sigma' \sqcup^n \#) \text{ за някое } n \geq 0.$$

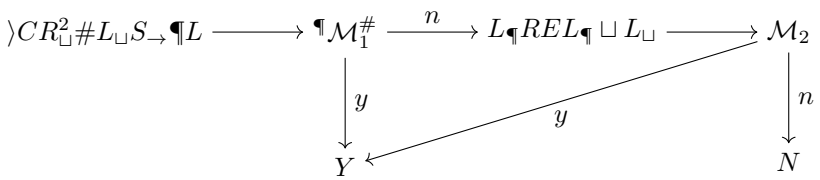
Следователно, ако $\mathcal{M}$ е разпознавател за езика L, то $\mathcal{M}^{\#}$ е разпознавател за $L\#$, като по време на изпълнението на $\mathcal{M}^{\#}$, започващо от начална конфигурация, символът $\#$ отбелязва най-дясната клетка на паметта, използвана от машината.

За всяка машина $\mathcal{M}$ с $\P\mathcal{M}$ означаваме машината, в която ролята на символа $\triangleright$ е подменена от символа $\P$.

Накрая, нека с E означим машината, която изтрива лентата надясно, започвайки от текущата позиция на четящата глава и свършвайки със символа $\#$, т.е.



Нека сега L_1 и L_2 са разрешими езици над Σ, като L_1 се разпознава от $\mathcal{M}_1$, а L_2 — от $\mathcal{M}_2$. Машина, разпознаваща $L_1 \cup L_2$ е следната



За всяка дума $w \in \Sigma^*$ частта $CR^2_{\sqcup}\#L_{\sqcup}S_{\rightarrow}\P L$ трансформира

$$\triangleright\underline{\sqcup}w$$

в

$$\triangleright \sqcup w\P\underline{\sqcup}w\#$$

След това ${}^{\P}\mathcal{M}_1^{\#}$ се изпълнява само върху частта $\P\underline{\sqcup}w\#$, като в случай че завърши с y, то $\mathcal{M}_1$ би разпознало w и тогава завършваме цялото изпълнение с y, или ${}^{\P}\mathcal{M}_1^{\#}$ завършва с n и лентата има вида

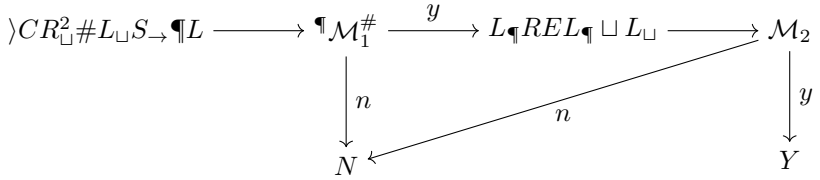
$$\triangleright \sqcup w\P\sigma\#,$$

като четящата глава се намира някъде в σ. Частта $L_{\P}REL_{\P} \sqcup L_{\sqcup}$ премества четящата глава над $\P$, след което изтрива всичко до $\#$ включително, връща се обратно над $\P$ и се позиционира отляво на w, т.е. достигаме до

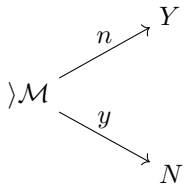
$$\triangleright\underline{\sqcup}w$$

Накрая пускаме $\mathcal{M}_2$, като ако $\mathcal{M}_2$ разпознае w, то завършваме с Y, а ако $\mathcal{M}_2$ не разпознае w, завършваме с N.

Аналогично, машина, разпознаваща $L_1 \cap L_2$ е



Накрая, ако L е език над Σ, който се разпознава от машината $\mathcal{M}$, то $\Sigma^* - L$ се разпознава от



□

Казваме, че машина на Тюринг $\mathcal{M}$ изчислява $f : \underbrace{\Sigma^* \times \cdots \times \Sigma^*}_{n} \to \Sigma^*$, $n \geq 1$ ако за всяко $w_1, \ldots, w_n \in \Sigma^*$ е изпълнено

$$(s, \triangleright\underline{\sqcup}w_1\$\ldots\$w_n) \vdash^*_{\mathcal{M}} (h, \triangleright\underline{\sqcup}f(w_1, \ldots, w_n)),$$

където s е началното състояние, h е стопсъстояние, а $\$ \notin \Sigma$. С други думи, $\mathcal{M}$ изчислява f, ако при всеки избор на думи w_1, …, w_n над Σ, ако разположим върху лентата на Машината думата $w_1\$w_2\$\ldots\$w_n$ (т.е. използваме $ като сепаратор) вдясно от левия край $\triangleright$, оставяйки една празна клетка и разположим четящата глава върху тази празна клетка, то след изпълнението на $\mathcal{M}$ главата отново ще бъде позиционирана върху клетката непосредствено до левия край $\triangleright$, като тази клетка отново ще бъде празна и непосредствено вдясно от нея ще бъде записана думата $f(w_1, w_2, \ldots, w_n)$.

Ще казваме, че една функция е *изчислима*, ако тя се изчислява от някоя машина на Тюринг.

Теорема 8.2. Изчислимите функции са затворени относно операцията суперпозиция.

Доказателство. Нека $k, n \geq 1$ и

$$g_i : (\Sigma^*)^n \to \Sigma^* \text{ за } 1 \leq i \leq k$$

и

$$h : (\Sigma^*)^k \to \Sigma^*$$

са изчислими функции. Нека $\mathcal{M}_{g_i}$ изчислява g_i за $1 \leq i \leq k$ и нека $\mathcal{M}_h$ изчислява h. Нека

$$f : (\Sigma^*)^n \to \Sigma^*$$

действа по правилото

$$f(w_1, \ldots, w_n) = h(g_1(w_1, \ldots, w_n), \ldots, g_n(w_1, \ldots, w_n)).$$

Ще опишем неформално действието на машина $\mathcal{M}$ изчисляваща f. Започваме от конфигурация

$$\triangleright\underline{\sqcup}w_1\$\ldots\$w_n.$$

1. $\mathcal{M}$ преобразува началалнта конфигурация до конфигурация

$$\triangleright\underbrace{\P\underline{\sqcup}w_1\$\dots\$w_n\P\sqcup w_1\$\dots\$w_n\P\dots\P\sqcup w_1\$\dots\$w_n\P}_{k}\#,$$

където ¶ и # не се използват от машините $\mathcal{M}_{g_i}$ за $1 \leq i \leq k$ и $\mathcal{M}_f$. Това можем да направим прилагайки многократно копиращата машина C и машината, преместваща надясно, $S_{\rightarrow}$, като слагаме символите ¶и # на подходящите места.

2. За $1 \leq i \leq k$ машината $\mathcal{M}'_{g_i}$ е модификация на машината $\mathcal{M}_{g_i}$, която работи точно както $\mathcal{M}_{g_i}$ с изключение на:

   (а) $\mathcal{M}'_{g_i}$ третира ¶ като $\triangleright$, когато го достигне при движение наляво;

   (б) когато $\mathcal{M}'_{g_i}$ достигне ¶ при движение надясно, премества всичко, което пише вдясно отнего с една клетка вдясно;

   (в) кагота четящата глава е в празна клетката непосредствено вляво от ¶ и $\mathcal{M}_{g_i}$ иска да премести главата наляво, то $\mathcal{M}'_{g_i}$ премества ¶ и всичко вдясно от ¶ с една клетка наляво, след което продължава с изпълнението на $\mathcal{M}_{g_i}$.

   Тогава $\mathcal{M}_{g_i}$ преобразува конфигурация $\triangleright\sigma$ до $\triangleright\sigma'$ тогава и само тогава, когато за всяка дума u машината $\mathcal{M}'_{g_i}$ преобразува $\P\sigma\P u\#$ до $\P\sigma'\P u\#$

   Върху конфигурацията, получена в предната точка, първо пускаме $\mathcal{M}_{g_1}$. След приключване на нейната работа получаваме конфигурацията

$$\triangleright\underbrace{\P\underline{\sqcup}g_1(w_1,\dots,w_n)\P\sqcup w_1\$\dots\$w_n\P\dots\P\sqcup w_1\$\dots\$w_n\P}_{k}\#$$

   Преместваме четящата глава на следващата празна клетка и получаваме

$$\triangleright\underbrace{\P\sqcup g_1(w_1,\dots,w_n)\P\underline{\sqcup}w_1\$\dots\$w_n\P\dots\P\sqcup w_1\$\dots\$w_n\P}_{k}\#$$

   Пускаме $\mathcal{M}_{g_2}$ и получаваме

$$\triangleright\underbrace{\P\sqcup g_1(w_1,\dots,w_n)\P\underline{\sqcup}g_2(w_1,\dots,w_n)\P\dots\P\sqcup w_1\$\dots\$w_n\P}_{k}\#$$

   Продължавайки аналогично, получаваме

$$\triangleright\P\sqcup g_1(w_1,\dots,w_n)\P\sqcup g_2(w_1,\dots,w_n)\P\dots\P\underline{\sqcup}g_k(w_1,\dots,w_n)\P\#$$

3. Изтриваме първото и последно ¶. Премахваме излишните интервали използвайки машина $S_{\leftarrow}$, аналогична на $S_{\rightarrow}$. Вътрешните ¶ заместваме с \$. Премахваме # и преместваме главата непосредствено до $\triangleright$. Така получаваме

$$\triangleright\underline{\sqcup}g_1(w_1,\dots,w_n)\$g_2(w_1,\dots,w_n)\$\dots\$g_k(w_1,\dots,w_n)$$

4. Пускаме машината $\mathcal{M}_f$ и получаваме

$$\triangleright\underline{\sqcup}h(g_1(w_1,\dots,w_n),g_2(w_1,\dots,w_n),\dots,g_k(w_1,\dots,w_n))$$

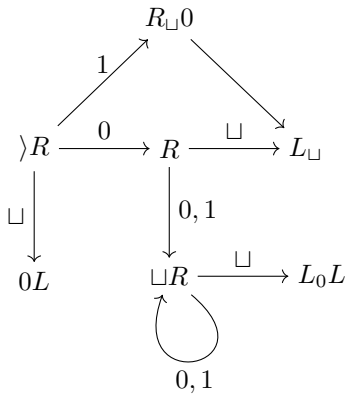
□

Ще казваме, че функцията $f:\mathbb{N}^n\to\mathbb{N}$ е изчислима, ако функцията $f_B:(\{0,1\}^*)^n\to\{0,1\}^*$, за която

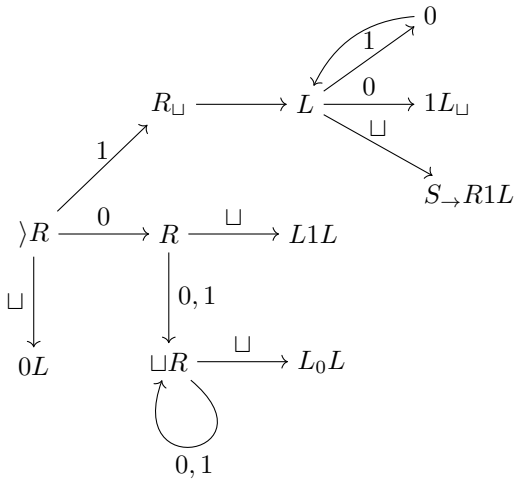
$$f_B(w_1,\dots,w_n)=\begin{cases}f(m_1,\dots,m_n), & \text{ако } w_i=B(m_i) \text{ за } 1\leq i\leq n\\ 0, & \text{иначе.}\end{cases}$$

е изчислима.

Пример. Функцията $f:\mathbb{N}\to\mathbb{N}$, действаща по правилото $f(n)=2n$ е изчислима. Машина, изчисляваща f_B, е


$R_{\sqcup}0$
1
$\rangle R$
0
R
$\sqcup$
$L_{\sqcup}$
$\sqcup$
$0, 1$
$0L$
$\sqcup R$
$\sqcup$
L_0L
$0, 1$

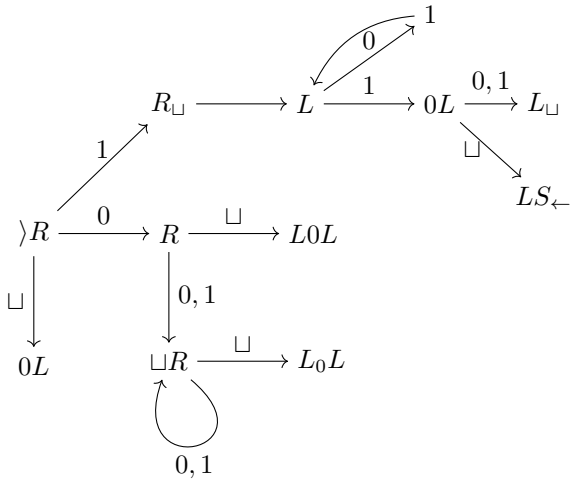
Пример. Функцията $f : \mathbb{N} \to \mathbb{N}$, действаща по правилото $f(n) = n + 1$ е изчислима. Машина, изчисляваща f_B, е $\mathcal{M}_{+1}$



Пример. Функцията $f : \mathbb{N} \to \mathbb{N}$, действаща по правилото

$$f(n) = n \dot{-} 1 = \begin{cases} n-1, & n > 0 \\ 0, & n = 0 \end{cases}$$

е изчислима. Машина, изчисляваща f_B, е $\mathcal{M}_{\dot{-}1}$



8.3 Многолентови машини на Тюринг

Основният проблем при използването на стандартни машини на Тюринг е работата с паметта. Така например алгоритъмът за разпознаване на обединението на два разрешими езика изисква непрекъснато да се следи най-дясната клетка посетена някога от четящата глава, а алгоритъмът за изчисляването на суперпозиция на изчислими функции изисква многократно преместване надясно и наляво на части от записа върху лентата. Някои от тези проблеми бихме могли да преодолеем лесно, ако машината на Тюринг имаше повече от една лента. Така например, ако можехме да използваме две ленти, за да разпознаем дали една дума принадлежи на обединението $L_1 \cup L_2$ на разршимите езици L_1 и L_2, щеше да е достатъчно да копираме думата от първата лента върху втората, след което да пуснем разпознавателя за L_1 да работи върху думата от първата лента, като в случай, че той даде отговор y, то думата е от обединението, а в случай, че даде отговор не, то пускаме разпознавателя за L_2 да работи върху втората лента. В този случай, ако той даде отговор y, то думата е от обединението, а ако отговорът е n, то думата не е от обединението. Аналогично, ако трябва да изчислим $h(g_1(w_1, \ldots, w_n), \ldots, g_k(w_1, \ldots, w_n))$, където h, g_1, $\ldots$, g_n са изичслими, достатъчно е да имаме $k+1$ ленти, като след получаване на входа $w_1\$\ldots\$w_n$ на първата лента, той се копира върху останалите k ленти, като първо пускаме машината, изчисляваща g_1 да работи върху втората лента, след което пускаме машината, изчисляваща g_2 да работи върху третата лента и т.н. След приключване на работата и последната маина, върху лентите с номера $2, \ldots k+1$ ще бъдат разположение съответно $g_1(w_1, \ldots, w_n)$, $\ldots$, $g_k(w_1, \ldots, w_n)$. Прехвърляме тази информация върху първата лента във формата $g_1(w_1, \ldots, w_n)\$\ldots\$g_k(w_1, \ldots, w_n)$, след което пускаме върху този вход да работи машината, изчисляваща h.

За да можем да работим с k ленти едновременно ще ни бъде необходима по една четяща глава за всяка лента, като всички глави трябва да могат да се движат независимо една от друга. Отново ще считаме, че всяка от лентите има ляв край, отбелязан с $\triangleright$. На всяка стъпка решението, какво да бъде предприето от четящите глави, ще зависи от това, което виждат всичките заедно. По-формално, под *k-лентова* машина на Тюринг ($k \geq 1$) ще разбираме наредена петорка $\mathcal{M} = (K, \Gamma, \delta, s, H)$, където K, Γ и H са крайни множества, представляващи съответно стоянията, символите (азбуката) и стопсъстоянията на машината ($H \subseteq K$), а δ е програмата на машината, като

$$\delta : (K - H) \times (\Gamma \cup \{\triangleright\})^k \longrightarrow K \times (\Gamma \cup \{\rightarrow, \leftarrow\})^k,$$

като ако $\delta(q, (x_1, \ldots, x_n)) = (q', (x'_1, \ldots, x'_n))$ и $x_i = \triangleright$ за някое $1 \leq i \leq k$, то $x'_i = \rightarrow$.

Конфигурация на k-лентова машина е наредена двойка $(q, (\sigma_1, \ldots, \sigma_k))$, където σ_i (за $1 \leq i \leq k$) е дума, в която точно един от символите е подчертан, като тя започва с $\triangleright$ и продължава със символи от Γ и в случай, че последния символ е $\sqcup$, то именно той е подчертаният.

Аналогично на случая на еднолентови машини, всяка k-лентова машина $\mathcal{M}$ дефинира релации $\vdash_{\mathcal{M}}$, $\vdash^n_{\mathcal{M}}$ и $\vdash^*_{\mathcal{M}}$, трансформиращи еднозначно k-лентови конфигурации съответно за една, n и неопределен брой стъпки.

Една k-лентова машина е разпознавател, ако стопсъстоянията ѝ са n и y. Казваме, че един разпознавател $\mathcal{M}$ приема (съответно отхвърля) дума w, ако $\mathcal{M}$ трансформира началтата конфигурация

$$(s, (\triangleright\underline{\sqcup}w, \triangleright\underline{\sqcup}, \ldots, \triangleright\underline{\sqcup}))$$

във финална конфигурация $(y, (\sigma_1, \ldots, \sigma_k))$ (съответно $(n, (\sigma_1, \ldots, \sigma_k))$). Един разпознавател разрешава езика $L \subseteq \Sigma^*$, ако приема всички думи от L и отхвърля всички други думи от Σ^*.

Ще казваме, че една k-лентова машина на Тюринг изчислява функцията $f : (\Sigma^*)^n \to \Sigma^*$, ако за всяко $w_1, \ldots, w_n \in \Sigma^*$ машината преобразува началната конфигурация

$$(s, (\triangleright\underline{\sqcup}w_1\$\ldots\$w_n, \triangleright\underline{\sqcup}, \ldots, \triangleright\underline{\sqcup}))$$

където \$ не символ от Σ, във финална конфигурация

$$(h, (\triangleright\underline{\sqcup}f(w_1, \ldots, w_n), \sigma_2, \ldots, \sigma_k)).$$

Теорема 8.3. Работата на всяка многолентова машина та Тюринг може да бъде симулирана от еднолентова машина на Тюринг.

Доказателство. Нека $\mathcal{M} = (K, \Gamma, \delta, s, H)$ е k-лентова машина на Тюринг, $k \geq 2$. Ще симулираме работата на $\mathcal{M}$ чрез еднолентова машина $\mathcal{M}^O$, която използва азбуката

$$\Gamma' = \Gamma \cup \{\underline{x} \mid x \in \Gamma\} \cup \{\P, \underline{\P}, \#\},$$

като символите $\underline{x}$ ще имат ролята на маркери на позицията на четящите глави на $\mathcal{M}$. Тъй като подчертаните символи са част от $\mathcal{M}^O$, за отбелязване на позицията на четящата ѝ глава ще използваме квадратче, с което ще обграждаме съответния символ.

$\mathcal{M}^O$ ще съдържа състоянията на $\mathcal{M}$. Ще казваме, че една конфигурацията на $\mathcal{M}^O$ е *точна*, ако тя има вида

$$(q, \triangleright\P\underline{\ \ x_1\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\boxed{\#}), \tag{8.1}$$

където q е състояние на $\mathcal{M}$, всички символи, които не са явно споменати не са нито ¶, нито са подчертани. Допуска се x_i да съвпада с ¶ вляво от него. Точната конфигурация (8.1) съответства на конфигурацията

$$(q, (\triangleright\underline{\ \ x_1\ \ }, \triangleright\underline{\ \ x_2\ \ }, \dots, \triangleright\underline{\ \ x_k\ \ }))$$

на $\mathcal{M}$.

В началото $\mathcal{M}^O$ трансформира входа

$$\triangleright\boxed{\sqcup}w$$

в

$$\triangleright\P\sqcup w\underbrace{\P\sqcup\P\dots\P\sqcup}_{k-1}\boxed{\#}$$

който съответства на конфигурацията

$$(s, (\triangleright\sqcup w, \triangleright\sqcup, \dots, \triangleright\sqcup))$$

на $\mathcal{M}$. $\mathcal{M}^O$ преминава към изпълнение на състоянието s на $\mathcal{M}$.

Оттук нататък работата на $\mathcal{M}^O$ ще бъде разделена на блокове, като всеки блок ще бъде посветен на изпълнението на дадено състояние на $\mathcal{M}$, като в случай, че блока започне симулация на изпълнение на състояние q на $\mathcal{M}$ от точна конфигурация

$$(q, \triangleright\P\underline{\ \ x_1\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\boxed{\#})$$

то той ще завърши в точна конфигурация

$$(q', \triangleright\P\underline{\_\_x'_1\_\_}\P\underline{\_\_x'_2\_\_}\dots\P\underline{\_\_x'_k\_\_}\boxed{\#})$$

и ще бъде в сила

$$(q, (\triangleright\underline{\ \ x_1\ \ }, \triangleright\underline{\ \ x_2\ \ }, \dots, \triangleright\underline{\ \ x_k\ \ })) \vdash_{\mathcal{M}} (q', \triangleright\underline{\_\_x'_1\_\_}, \triangleright\underline{\_\_x'_2\_\_}, \dots, \triangleright\underline{\_\_x'_k\_\_})$$

Нека $\mathcal{M}^O$ е завършила поредния блок от изпълнение на някое състояние на $\mathcal{M}$ в точна конфигурация

$$\triangleright\P\underline{\ \ x_1\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\boxed{\#}$$

и сега трябва да се изпълни състоянието $q \in K - H$ на $\mathcal{M}$.

Тъй като конфигурацията е точна, то между $\triangleright$ и $\#$ има точно k символа ¶ (някои от които евентуално подчертани). Освен това между $\triangleright$ и $\#$ има точно k подчертани символа, като между всеки два символа ¶ има точно един подчертан символ.

$\mathcal{M}^O$ премества главата вляво върху символа $\triangleright$

$$\boxed{\triangleright}\P\underline{\ \ x_1\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\#$$

като по пътя си запаметява последователно (с помощта на състоянията си) подчертаните символи $x_k, \dots, x_1$ (като ако символът е бил ¶, то вместо него запаметяваме $\triangleright$). Символите $x_1, \dots, x_k$ съответстват на символите, които виждат четящите глави на $\mathcal{M}$ и значи сега $\mathcal{M}$ трябва да изпълни състоянието q върху $(x_1, \dots, x_k)$. Нека

$$\delta(q, (x_1, \dots, x_k)) = (q', (y_1, \dots, y_k)).$$

Сега $\mathcal{M}^O$ премества четящата глава надясно, докато не достигне до първия подчертан символ. Това е символът $\underline{x_1}$.

$$\triangleright\P\underline{\ \ \boxed{x_1}\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\#$$

В този момент $\mathcal{M}^O$ изпълнява y_1 по следната схема:

1. Ако $y_1 \in \Gamma$ (т.е. y_1 не е $\rightarrow$ или $\leftarrow$), то $\mathcal{M}^O$ замества $\underline{x_1}$ с $\underline{y_1}$ и оставя четящата глава върху него.

$$\triangleright\P\underline{\ \ \boxed{y_1}\ \ }\P\underline{\ \ x_2\ \ }\dots\P\underline{\ \ x_k\ \ }\#$$

2. Ако y_1 е $\rightarrow$, то $\mathcal{M}^O$ проверява, дали следващият символ, е ¶ (или $\underline{¶}$). Ако отговорът е не, то този символ е $z_1 \in \Gamma$

$$\triangleright ¶ \_\_ \boxed{\underline{x_1}} z_1 \_\_ ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

Тогава $\mathcal{M}^O$ замества $\underline{x}_1$ с x_1, z_1 с $\underline{z}_1$ и оставя главата върху $\underline{z}_1$.

$$\triangleright ¶ \_\_ x_1 \boxed{\underline{z_1}} \_\_ ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

Ако отговорът е да, т.е. имаме

$$\triangleright ¶ \_\_ \boxed{\underline{x_1}} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

то $\mathcal{M}^O$ премества цялата информация, записана вдясно от $\underline{x}_1$ с една клетка надясно (т.е. изпълнява S_L)

$$\triangleright ¶ \_\_ \boxed{\underline{x_1}} \sqcup ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

след което замества $\underline{x}_1$ с x_1, а в празната клетка непосредствено вдясно от x_1 записва $\underline{\sqcup}$ и оставя четящата глава там

$$\triangleright ¶ \_\_ \underline{x_1} \boxed{\underline{\sqcup}} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

3. Ако y_1 е $\leftarrow$, x_1 е $\sqcup$, z_1 е символът непосрествено вляво от x_1, а символът непосредствено вдясно от x_1 е ¶

$$\triangleright ¶ \_\_ z_1 \boxed{\underline{\sqcup}} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

то $\mathcal{M}^O$ то $\mathcal{M}^O$ изтрива $\underline{x}_1$

$$\triangleright ¶ \_\_ z_1 \boxed{\sqcup} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

премества една клетка наляво цялата информация, записана вдясно от x_1

$$\triangleright ¶ \_\_ \boxed{z_1} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

след което замества символа z_1 с $\underline{z}_1$ и оставя четящата глава върху него

$$\triangleright ¶ \_\_ \boxed{\underline{z_1}} ¶ \_\_ \underline{x_2} \_\_ \ldots ¶ \_\_ \underline{x_k} \_\_ \#$$

Ако отговорът е не, то $\mathcal{M}^O$ замества $\underline{x}_1$ с x_1 и z_1 с $\underline{z}_1$ и оставя четящата глава там.

След като изпълни y_1, $\mathcal{M}^O$ премества четящата глава вдясно, докато не открие следващия подчертан символ, който е точно x_2. След като открие x_2, $\mathcal{M}^O$ изпълнява y_2 по същата схема, по която изпълнява y_1, след което мести четящата глава надясно, докато не открие следващия подчертан символ (който е x_3) и т.н. След като изпълни y_k, $\mathcal{M}^O$ премества четящата глава (надясно) върху # и преминава към изпълнение на състоянието q' на $\mathcal{M}^O$.

Ако $\mathcal{M}$ е разпознавател (т.е. $H = \{y, n\}$), когато $\mathcal{M}^O$ достигне до стопсъстояние на $\mathcal{M}$, то тя спира изпълнението в това състояние. В противен случай, когато достигне до стопсътояние $\mathcal{M}^O$ придвижва четящата глава (наляво, броейки с помощтта на състоянията) до първото ¶и проверява дали символа непосредствено вдясно от него е $\underline{\sqcup}$. Ако не, $\mathcal{M}^0$ прекратява изпълнението. Ако да, то $\mathcal{M}^0$ премества четящата глава надясно до следващото ¶, изтрива всичко от това ¶ до #, включително, след което се връща наляво до ¶, изтрива го, премества се наляво, докато не намери $\underline{x}_1$, замества го x_1, продължава наляво до първото ¶, изтрива го, премества информацията вдясно една клетка вляво и преминава във стопсъстояние.

□

Пример. Функцията $f : \mathbb{N}^2 \rightarrow \mathbb{N}$, действща по правилото $f(m, n) = m + n$ е изчислима. Наистина можем да изчислим f_B чрез трилентова машина на Тюринг, действаща по следната схема. Да напомним, че машината тррябва да работи правилно при начална конфигурация

$$\triangleright \underline{\sqcup} w_1 \$ w_2$$
$$\triangleright \underline{\sqcup}$$
$$\triangleright \underline{\sqcup}$$

където $w_1, w_2 \in \{0, 1\}^*$, като в случай, че една от тях не е двоичен запис на естествено число, машината трябва да върне 0:

1. Преместваме главата на втора лента една клетка вдясно. Преместваме четящата глава на първата лента надясно, докато не достигнем до \$, като същевременно прочетеното се изтрива от първата лента и се копира върху втората лента. Връщаме главата на втората лента до празната клетка вдясно от $\triangleright$. Така достигаме до конфигурация

$$\begin{aligned}&\triangleright \sqcup \underbrace{\underline{\sqcup} \ldots \sqcup}_{|w_1|} \underline{\$} w_2\\ &\triangleright \underline{\sqcup} w_1\\ &\triangleright \underline{\sqcup}\end{aligned}$$

2. Изтриваме \$ от първата лента. Преместваме четящата глава на първата лента надясно, докато не достигнем до $\sqcup$, като същевременно прочетеното се изтрива от първата лента и се копира върху третата лента. Връщаме главите на първата и третата лента до празната клетка вдясно от $\triangleright$. Така достигаме до конфигурация

$$\begin{aligned}&\triangleright \underline{\sqcup}\\ &\triangleright \underline{\sqcup} w_1\\ &\triangleright \underline{\sqcup} w_2\end{aligned}$$

3. Проверяваме, дали w_1 е двоичен запис на естествено число, т.е. дали w_1 е 0 или е дума, започваща с 1. Ако не е, записваме 0 във втората клетка отдясно на $\triangleright$ в първа лента, оставяме главата вляво и спираме изпълнението.

4. Проверяваме, дали w_2 е двоичен запис на естествено число. Ако не е, записваме 0 във втората клетка отдясно на $\triangleright$ в първа лента, оставяме главата вляво и спираме изпълнението.

5. Ако и двете проверки са успешни, т.е. $w_1 = B(m)$ и $w_2 = B(n)$ изпълняваме следния цикъл:

   (а) Проверяваме дали думата на трета лента е 0. Ако да, то спираме цикъла и преминаваме към следващата стъпка от алгоритъма.

   (б) Ако думата на трета лента не е 0, то върху трета лента изпълняваме $\mathcal{M}_{\dot{-}1}$, след което върху втора изпълняваме $\mathcal{M}_{+1}$

   Така на всяка итерация от цикъла, ако тя започне в конфигурация

$$\begin{aligned}&\triangleright \underline{\sqcup}\\ &\triangleright \underline{\sqcup} B(m')\\ &\triangleright \underline{\sqcup} B(n')\end{aligned}$$

   където $n' > 0$, то итерацията завършва в

$$\begin{aligned}&\triangleright \underline{\sqcup}\\ &\triangleright \underline{\sqcup} B(m'+1)\\ &\triangleright \underline{\sqcup} B(n'-1)\end{aligned}$$

   Тъй като цикълът започва от

$$\begin{aligned}&\triangleright \underline{\sqcup}\\ &\triangleright \underline{\sqcup} B(m)\\ &\triangleright \underline{\sqcup} B(n)\end{aligned}$$

   то той окончателно завършва при конфигурация

$$\begin{aligned}&\triangleright \underline{\sqcup}\\ &\triangleright \underline{\sqcup} B(m+n)\\ &\triangleright \underline{\sqcup} B(0)\end{aligned}$$

6. Преместваме главата на първа лента една клетка вдясно, преписваме думата от втора лента върху първа лента, връщаме главата на първа лента върху празната клетка до $\triangleright$

$$\begin{aligned}&\triangleright \underline{\sqcup} B(m+n)\\ &\triangleright \underline{\sqcup} B(m+n)\\ &\triangleright \underline{\sqcup} B(0)\end{aligned}$$

   и спираме изпълнението.

Пример. Функцията $f : \mathbb{N}^2 \to \mathbb{N}$, действаща по правилото $f(m, n) = mn$ е изчислима. Наистина можем да изчислим f_B чрез четирилентова машина на Тюринг, действаща по следната схема. Започваме изпълнението, аналогично на машината от предния пример докато не достигнем до конфигурацията

$$\begin{array}{l} \triangleright\underline{\sqcup} \\ \triangleright\underline{\sqcup}B(0) \\ \triangleright\underline{\sqcup}B(m) \\ \triangleright\underline{\sqcup}B(n) \end{array}$$

където $B(m) = w_1$ и $B(n) = w_2$ (в случай, че проверката, дали w_1 и w_2 са двоични записи на естествени числа, даде отрицателен резултат, то машината връща 0 и спира).

Оттук нататък изпълняваме следния цикъл:

1. Проверяваме дали думата на четвърта лента е 0. Ако да, то спираме цикъла и преминаваме към следващата стъпка от алгоритъма.
2. Ако думата на четвърта лента не е 0, то върху четвърта лента изпълняваме $\mathcal{M}_{\dot{-}1}$. Преместваме главата на втора лента до първата празна вдясно от написаната дума, записваме символа \$, след което преписваме отдясно на него думата на трета лента и връщаме главите на втора и трета лента върху празните клетки непосредствено вдясно от $\triangleright$. Накрая, върху втора изпълняваме $\mathcal{M}_+$, която е еднолентов вариант на машината от предния пример.

Така на всяка итерация от цикъла, ако тя започне в конфигурация

$$\begin{array}{l} \triangleright\underline{\sqcup} \\ \triangleright\underline{\sqcup}B(m') \\ \triangleright\underline{\sqcup}B(m) \\ \triangleright\underline{\sqcup}B(n') \end{array}$$

където $n' > 0$, то итерацията минава последователно през

$$\begin{array}{l} \triangleright\,\underline{\sqcup} \\ \triangleright\,\underline{\sqcup}B(m') \\ \triangleright\,\underline{\sqcup}B(m) \\ \triangleright\,\underline{\sqcup}B(n'-1) \end{array} \qquad \begin{array}{l} \triangleright\,\underline{\sqcup} \\ \triangleright\,\sqcup B(m')\underline{\sqcup} \\ \triangleright\,\underline{\sqcup}B(m) \\ \triangleright\,\underline{\sqcup}B(n'-1) \end{array}$$

$$\begin{array}{l} \triangleright\,\underline{\sqcup} \\ \triangleright\,\sqcup B(m')\underline{\$} \\ \triangleright\,\underline{\sqcup}B(m) \\ \triangleright\,\underline{\sqcup}B(n'-1) \end{array} \qquad \begin{array}{l} \triangleright\,\underline{\sqcup} \\ \triangleright\,\underline{\sqcup}B(m')\$B(m) \\ \triangleright\,\underline{\sqcup}B(m) \\ \triangleright\,\underline{\sqcup}B(n'-1) \end{array}$$

и завършва в

$$\begin{array}{l} \triangleright\underline{\sqcup} \\ \triangleright\underline{\sqcup}B(m'+m) \\ \triangleright\underline{\sqcup}B(m) \\ \triangleright\underline{\sqcup}B(n'-1) \end{array}$$

като е изпълнено $(m' + m) + m(n' - 1) = m' + mn$. Така окончателно той завършва в конфигурацията

$$\begin{array}{l} \triangleright\underline{\sqcup} \\ \triangleright\underline{\sqcup}B(mn) \\ \triangleright\underline{\sqcup}B(m) \\ \triangleright\underline{\sqcup}B(0) \end{array}$$

Накрая копираме съдържанието на втора лента върху първа лента

$$\begin{array}{l} \triangleright\underline{\sqcup}B(mn) \\ \triangleright\underline{\sqcup}B(mn) \\ \triangleright\underline{\sqcup}B(m) \\ \triangleright\underline{\sqcup}B(0) \end{array}$$

и спираме изпълнението.

Забележка. Машината от първия пример пресмята $m+n$ приблизително за $m+n$ стъпки, а втора — mn приблизително за mn стъпки. Важна особеност е, използваме за вход на двете машини думите $B(m)$ и $B(n)$, които имат приблизителни дължини $\log_2 m$ и $\log_2 n$. Това означава, че спрямо дължината на входа си двете машини работят експоненциално дълго време. Далеч по-ефективни са алгоритмите за събиране и умножение, изучавани в училище. При тях, алгоритъмът за събиране е линеен спрямо дължината на входната дума, а този за умножение — квадратичен спрямо входа.

Пример. Нека функцията $g : \mathbb{N}^{k+1} \to \mathbb{N}$, $k \geq 1$ е изчислима и за всяко $\overline{x} \in \mathbb{N}^k$ съществува $y \in \mathbb{N}$, такова че $g(\overline{x}, y) = 0$. Нека $f : \mathbb{N}^k \to \mathbb{N}$ е дефинирана, така че

$$\begin{aligned} g(\overline{x}, f(\overline{x})) &= 0 \\ g(\overline{x}, y) &\neq 0 \text{ за } y < f(\overline{x}) \end{aligned}$$

т.е. $f(\overline{x})$ е най-малкото y, за което $g(\overline{x}, y) = 0$. Ще означаваме

$$f(\overline{x}) = \mu y[g(\overline{x}, y) = 0]$$

и ще казваме, че f се получава от g чрез *минимизация*.

Функцията f е изчислима с помощта на следната четирилентова машина на Тюринг. В началото машината трансформира началната конфигурация

$$\begin{aligned} &\triangleright\underline{\sqcup} w_1\$ \ldots \$w_k \\ &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} \end{aligned}$$

в

$$\begin{aligned} &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} w_1\$ \ldots \$w_k \\ &\triangleright\underline{\sqcup} 0 \end{aligned}$$

след което проверява, че w_1, w_2, …, w_k са двоични записи на естествени числа. Ако някоя от думите не е двоичен запис на естествено число, то връщаме 0 и прекратяваме изчислението. В противен случай, конфигурацията има вида

$$\begin{aligned} &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} \\ &\triangleright\underline{\sqcup} B(n_1)\$ \ldots \$B(n_k) \\ &\triangleright\underline{\sqcup} 0 \end{aligned}$$

където n_1, …, n_k са естествени числа.

Оттук нататък изпълняваме следния цикъл:

1. Преписваме третата лента върху втората лента, като оставяме главата на втората лента вдясно от думата, а главата на третата лента вдясно от думата.
2. Върху втората лента записваме \$, след което преписваме съдържанието на четвъртата лента непосредствено вдясно от \$ и преместваме четящата глава непосредствено вдясно от $\triangleright$.
3. Пускаме върху втората лента еднолентов вариант на машина изчисляваща g.
4. Ако върху втората лента пише 0, преписваме четвъртата лента върху първата лента и прекратяваме изпълнението на машината. Ако не, то изтриваме съдържанието на втора лента и пускаме върху четвърта лента машината $\mathcal{M}_{+1}$

5. Започваме цикъла отначало.

Така всяка итерация от цикъла започва в конфигурация

$$\begin{array}{l}\triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\\ \triangleright\underline{\sqcup}B(n)\end{array}$$

минава последователно през

$$\begin{array}{ll}\triangleright\underline{\sqcup} & \triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup} & \triangleright\sqcup B(n_1)\$\ldots\$B(n_k)\underline{\sqcup}\\ \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k) & \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\\ \triangleright\underline{\sqcup}B(n) & \triangleright\underline{\sqcup}B(n)\end{array}$$

$$\begin{array}{ll}\triangleright\underline{\sqcup} & \triangleright\underline{\sqcup}\\ \triangleright\sqcup B(n_1)\$\ldots\$B(n_k)\underline{\$} & \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\$B(n)\\ \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k) & \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\\ \triangleright\underline{\sqcup}B(n) & \triangleright\underline{\sqcup}B(n)\end{array}$$

$$\begin{array}{l}\triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup}B(g(n_1,\ldots,n_k,n)\\ \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\\ \triangleright\underline{\sqcup}B(n)\end{array}$$

В случай, че на втора линия пише 0, то $g(n_1,\ldots,n_k,n)=0$ и това е първото n, за което това се случва и за това машината извежда $B(n)$ на първата лента. Ако на втора лента пише нещо различно от 0, то $g(n_1,\ldots,n_k,n)\neq 0$, при което машината трансформира конфигурацията в

$$\begin{array}{l}\triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup}\\ \triangleright\underline{\sqcup}B(n_1)\$\ldots\$B(n_k)\\ \triangleright\underline{\sqcup}B(n+1)\end{array}$$

и започва цикъла отначало.

Пример. Лесно може да се съобрази, че всяка функция $I_n^k:\mathbb{N}^n\to\mathbb{N}$, където $n\geq 1$ и $1\leq k\leq n$, действаща по правилото

$$I_n^k(x_1,\ldots,x_n)=x_k$$

е изчислима.

Изчислимите функции над естествените числа имат следното алгебрично описание

Теорема 8.4. Класът на изчислимите функции над $\mathbb{N}$ е най-малкият клас, съдържащ функциите събиране, умножение, както и функциите I_n^k за всяко $n\geq 1$ и всяко $1\leq k\leq n$, и затворен относно суперпозиция и минимизация (както е дефинирана в предпоследния пример).

8.4 Полуразрешими и изчислимономеруеми множества. Частични изчислими функции

Да напомним, че една машина на Тюринг е разпознавател, ако има точно две стопсъстояния, а именно y и n, като при спиране в състояние y машината приема входа, а при спиране в състояние n отхвърля входа. Един разпознавател разрешава език L, ако приема думите от L и отхвърля думите, които не са от L.

Така в случай, че един разпознавател разрешава един език, то той трябва да спира върху всеки вход. От друга страна, машините на Тюринг, дори и когато са разпознаватели, не са длъжни да спират работата си върху всеки вход. Да разгледаме например разпознавателя със състояние $\{s, y, n\}$, символи $\{0, 1\}$ и програма δ, действаща по правилото

$$\delta(s, x) = (s, \rightarrow), \text{ за всеки символ } x.$$

Действието на тази машина е неограничено движение надясно, като входа на машината няма никакво отношение към нейното поведение. Тази машина няма да завърши работа върху нито един вход (в частност не разрешава нито един език).

За да можем да дадем по-точно описание на отношението между езици и разпознаватели въвеждаме понятието полуразрешимост. Казваме, че един език L е полуразрешим, ако съществува разпознавател, който приема думите от L, а думите, които не са от L, или се отхвърлят, или разпознавателят никога не спира върху тях. Отново за да можем да работим не само с езици (т.е. множества от думи), а и с релации от думи, като на релацията

$$R \subseteq (\Sigma^*)^n$$

съпоставяме езика

$$L_R = \{w_1\$ \ldots \$w_n \mid (w_1, \ldots, w_n) \in R)\}$$

(където \$ е специален разделител, който не се съдържа в Σ), като релацията R е полуразрешима тогава и само тогава, когато L_R е полуразрешим. Както и в случай на разрешимост на множества от естествени числа, казваме че едно множество

$$A \subseteq \mathbb{N}^k$$

е полуразрешимо, ако езикът

$$\mathbb{B}(A) = \{B(n_1)\$ \ldots \$B(n_k) \mid (n_1, \ldots, n_k) \in A\}$$

е полуразрешим.

Пример. Да разгледаме множеството $A \subseteq \mathbb{N}^2$, състоящо се от онези двойки (a, d), за които аритметичната прогресия с първи член a и разлика d съдържа поне два последователни члена, които са прости числа. Не е ясно, дали множеството A е разрешимо. За сметка на това, можем да покажем, че A е полуразрешимо, реализирайки следната проста схема: изчисляваме последователно a, $a + d$, $a + 2d$, $\ldots$, като същевременно проверяваме дали всяко едно от тези числа е просто. В случай, че за някое k се окаже, че $a + kd$ и $a + (k + 1)d$ са прости то спираме и разпознаваме двойката (a, d). Тази схема спира работа тогава и само тогава, когато двойката $(a, d) \in A$ и значи A е полуразрешимо.

Теорема 8.5. Езикът $L \subseteq \Sigma^*$ е разрешим тогава и само тогава, кагото L и $\Sigma^* - L$ са полуразрешими.

Доказателство. Ясно е, че ако един език е разрешим, то тогава той е полуразрешим (при това с един и същи разпознавател). Следователно, ако L е разрешим, то имаме, че $\Sigma^* - L$ също е разрешим, откъдето L и $\Sigma^* - L$ са полуразрешими.

Нека сега L и $\Sigma^* - L$ са полуразрешими и нека $\mathcal{M}_+$ и $\mathcal{M}_-$ са два еднолентови разпознавателя, които полуразрешават съответно L и $\Sigma^* - L$. Да разгледаме двулентов разпознавател $\mathcal{M}$, който работи по следната схема:

1. В началото преписва входа (разположен върху първата лента) върху втората лента.

2. Докато една от машините $\mathcal{M}_+$ и $\mathcal{M}_-$ не разпознае входа, пуска $\mathcal{M}_+$ да работи една стъпка върху първата лента, след което $\mathcal{M}_-$ една стъпка върху втората лента, след което отново $\mathcal{M}_+$ една стъпка върху първа лента, след което отново $\mathcal{M}_-$ една стъпка върху втората лента и т.н. В случай, че $\mathcal{M}_+$ приеме входа, то $\mathcal{M}$ спира работа и приема входа, а ако $\mathcal{M}_-$ приеме входа, то $\mathcal{M}$ спира работа и отхвърля входа.

□

Казваме, че изображението f е *номерация* на множеството A, ако е сюрективно изображение от $\mathbb{N}$ върху A. Ще казваме, че езикът L е *изчислимономеруем*, ако съществува изчислима номерация на L. Ще считаме, че празното множество е изчислимономеруемо.

С други думи, един непразен език L е изчисимономеруем, ако съществува изчислима редица

$$a_0, a_1, \ldots, a_n, \ldots,$$

такава, че

$$L = \{a_n \mid n \in \mathbb{N}\}.$$

Да отбележим, че нямаме никакви ограничения върху реда, в който се изреждат елементите на L. Освен това, един и същи елемент на L може да участва повече от един път в редицата. Нещо повече, ако L е краен, то поне един от неговите елементи трябва да участва безброй много пъти в редицата.

Пример. Нека Σ е $n+1$ буквена азбука, $n \geq 0$. Фиксираме едно изреждане

$$a_0, \ldots, a_n$$

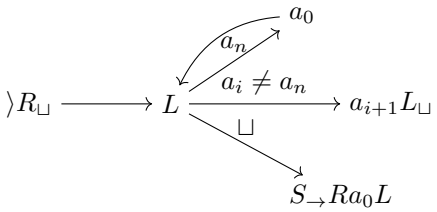
на буквите на Σ. Лексикографската наредбда $<_L$ върху думите на Σ се дефинира по следния начин

$$w <_L w' \quad \stackrel{\text{def}}{\iff} \quad \begin{array}{l} |w| < |w'| \text{ или} \\ |w| = |w'|,\ w = ua_iv,\ w' = ua_jv',\ \text{като } i < j \end{array}$$

Да отбележим, че лексикографската наредба е добра наредба, т.е. има най-малък елемент и няма безкрайно намаляваща редица. В частност, тъй като Σ^* е изброимо, то съществува единствена биекция $\lambda : \mathbb{N} \to \Sigma^*$, такава че

$$m < n \iff \lambda(m) <_L \lambda(n).$$

Ще докажем, че λ е изчислима номерация. За целта първо разглеждаме машината $\mathcal{M}_S$



която при вход $\lambda(n)$ дава изход $\lambda(n+1)$. Сега можем да разгледаме трилентовата машина $\mathcal{M}_\lambda$, изчисляваща λ по следната схема:

1. Проверява дали входа е двоичен запис на естествено число. Ако не е, то изписва 0 върху първата лента и спира работа. В противен случай:

2. Записва 0 в третата лента, като по този начин получаваме конфигурацията

$$\begin{array}{l} \triangleright \underline{\sqcup} B(n) \\ \triangleright \underline{\sqcup} \\ \triangleright \underline{\sqcup} 0 \end{array}$$

и започва изпълнение на следния цикъл:

(а) Проверява дали числата върху първа и трета лента съвпадат. Ако това е така, то изтрива съдържанието на първа лента, копира думата от втора лента върху първа лента и спира работата на машината. Ако не, то:

(б) Изпълнява машината $\mathcal{M}_S$ върху втора лента.

(в) Изпълнява машината $\mathcal{M}_{+1}$ върху трета лента.

(г) Започва цикъла отначало.

По този начин всяка итерация на цикъла започва при конфигурация

$$\triangleright\underline{\sqcup}B(n)$$
$$\triangleright\underline{\sqcup}\lambda(m)$$
$$\triangleright\underline{\sqcup}B(m)$$

където $m \leq n$, като в случай че $n = m$ машината преминава в конфигурация

$$\triangleright\underline{\sqcup}\lambda(n)$$
$$\triangleright\underline{\sqcup}\lambda(n)$$
$$\triangleright\underline{\sqcup}B(n)$$

и спира работа, а в случай че $n < m$ изпълнява пълната итерация и завършва в конфигурация

$$\triangleright\underline{\sqcup}B(n)$$
$$\triangleright\underline{\sqcup}\lambda(m+1)$$
$$\triangleright\underline{\sqcup}B(m+1)$$

Теорема 8.6. Един език е полуразрешим тогава и само тогава, когато той е изчислимономеруем.

Доказателство. Нека $L \subseteq \Sigma^*$ е полуразрешим. Ако $L = \emptyset$, то L е изчислимономеруем. Нека сега L е непразен и нека $\mathcal{M}_L$ е машина, полуразрешаваща L. Ясно е, че множеството

$$S_L = \{(w, n) \mid \mathcal{M}_L \text{ разпознава } w \text{ за по-малко от } n \text{ стъпки}\}$$

е разрешимо. Номерираме L по следната схема: Нека

$$a_0, a_1, \ldots, a_m, \ldots$$

е лексикографското изреждане на думите от Σ^*. Ще използваме брояч C, който в началото е равен на 0. Изпълняваме следния цикъл:

Проверяваме последователно, дали двойките

$$(a_0, C), (a_1, C), \ldots, (a_C, C)$$

принадлежат на S_L. В случай, че (a_i, C) принадлежи, то номерираме a_i. Наркрая увеличаваме брояча C с 1 и започваме цикъла отначало. Така например, ако a_{10} се приема от $\mathcal{M}_L$ за 5 стъпки, a_3 за 7 стъпки, a_{23} за 8 стъпки, и няма друга дума, която се приема за по-малко от 10 стъпки, то след 10 итерации на цикъла бихме имали редицата

$$\underbrace{a_{10}}_{\text{6-та}}, \underbrace{a_{10}}_{\text{7-ма}}, \underbrace{a_3, a_{10}}_{\text{8-а}}, \underbrace{a_3, a_{10}, a_{23}}_{\text{9-а}}, \underbrace{a_3, a_{10}, a_{23}}_{\text{10-а}}$$

За всяка дума a_k на Σ^*, ако $a_k \notin L$, то $(a_k, m) \notin S_L$ за всяко m и значи a_k никога няма да бъде номерирано. От друга страна, ако $a_k \in L$, то $\mathcal{M}_L$ разпознава a_k за някакъв брой стъпки n и следователно a_k ще бъде номерирана на всяка итерация на цикъла след n-тата.

Нека сега L е непразен и изчислимономеруем и нека f е изчислима номерация на L. Можем да полуразрешим L по следната схема: При даден вход w генерираме последователно

$$f(0), f(1), f(2), \ldots$$

В случай, че $f(i) = w$ за някое i, то прекратяваме изчислението и приемаме w. Така (тъй като f е номерация) ще приемем всяка дума от L, а за всяка дума, която не е от L изчислението никога няма да спре.

□

Казваме, че f е *частична функция* от A в B, ако f е функция с $\mathrm{dom} f \subseteq A$ и $\mathrm{range} f \subseteq B$. Казваме, че f е *частична изчислима функция* от $(\Sigma^*)^n$ в Σ'^*, ако f е частична функция от $(\Sigma^*)^n$ в Σ'^* и съществува машина на Тюринг $\mathcal{M}$, така че за всяка n-орка $(w_1, \ldots, w_n)$, ако $(w_1, \ldots, w_n) \in \mathrm{dom} f$, то

$$(s, \triangleright\underline{\sqcup}w_1\$\ldots\$w_n) \vdash^*_{\mathcal{M}} (h, \triangleright\underline{\sqcup}f(w_1, \ldots, w_n) \text{ за някое стопсъстояние } h$$

а ако $(w_1, \ldots, w_n) \notin \mathrm{dom} f$, то

$$(s, \triangleright \underline{\sqcup} w_1 \$ \ldots \$ w_n) \not\vdash^*_{\mathcal{M}} (h, \sigma) \text{ за всяко стопсъстояние } h,$$

т.е. $\mathcal{M}$ никога не спира.

Теорема 8.7. Една функция е частична изчислима тогава и само тогава, когато графиката ѝ е полуразрешимо множество.

Доказателство. Нека първо f е частична изчислима функция от $(\Sigma^*)^n$ в Σ'^* и нека $\mathcal{M}_f$ е машина на Тюринг, изчисляваща f. Можем да полуразрешим

$$G_f = \{(w_1, \ldots, w_n, f(w_1, \ldots, w_n)) \mid (w_1, \ldots, w_n) \in \mathrm{dom} f\}$$

чрез разпознавателя $\mathcal{M}$, действащ по следната схема: При вход

$$\triangleright \underline{\sqcup} w_1 \$ \ldots \$ w_n \$ w$$

преписваме

$$\underline{\sqcup} w_1, \ldots, w_n$$

върху втора лента и пускаме върху нея да работи $\mathcal{M}_f$. В случай, че $\mathcal{M}_f$ спре работа, то втората лента има вида

$$\triangleright \underline{\sqcup} f(w_1, \ldots, w_n)$$

Проверяваме дали $f(w_1, \ldots, w_n) = w$. Ако да, спираме работа и приемаме думата, а ако не, спираме работа и отхвърляме думата.

Нека сега G_f е полуразрешимо множество. Нека g е изчислима номерация на G_f. Можем да изчислим f чрез трилентова машина на Тюринг $\mathcal{M}$, действаща по следната схема: Нека е даден вход

$$\triangleright \underline{\sqcup} w$$

С помощта на третата лента, която служи за брояч, генерираме последвователно

$$g(0), g(1), \ldots g(n), \ldots$$

върху втората лента. Всяко $g(i)$ има вида

$$u^i_1 \$ \ldots \$ u^i_n \$ f(u^i_1, \ldots, u^i_n)$$

След като то бъде изписано върху втората лента проверяваме, дали

$$u^i_1 \$ \ldots \$ u^i_n = w$$

Ако да, то изтриваме първата лента, изписваме върху нея $f(u^i_1, \ldots, u^i_n)$ и спираме работа. Ако не, то генерираме $g(i+1)$ и повтаряме проверката.

□

Теорема 8.8. Един език е полуразрешим тогава и само тогава, котаго той е дефиниционна област на частична изчислима функция.

Доказателство. Нека $L \subseteq \Sigma^*$ е полуразрешим език и нека $\mathcal{M}_L$ полуразрешава L. Тогава L е дефиниционната област на частичната функция f, действща по правилото

$$f(w) = \begin{cases} \varepsilon, & w \in L \\ \neg!, & w \notin L \end{cases}$$

където $\neg!$ означава, че функцията не е дефинирана. Частичната функция може да се изчисли по следния начин: При вход

$$\triangleright \underline{\sqcup} w$$

пускаме $\mathcal{M}_L$ да работи върху този вход. В случай, че $\mathcal{M}_L$ спре и приеме думата, то връщаме ε и спираме, а в случай, че $\mathcal{M}_L$ спре и не приеме думата, то започваме процедура, която никога не завършва (например, на всяка стъпка местим четящата глава надясно).

Обратно, нека L е дефиниционната област на частичната изчислима функция g и нека $\mathcal{M}_g$ изчислява g. Можем да полуразрешим L по следния начин: При вход

$$\triangleright \underline{\sqcup} w$$

пускаме $\mathcal{M}_g$ да работи върху входа. В случай, че тя спре и даде резултат, спираме изпълнението и приемаме думата.

□

8.5 Код на машина на Тюринг. Универсална машина

Нека Γ е фиксирана азбука, съдържаща $\sqcup$, 0 и 1, но не и $\triangleright$, $\leftarrow$, $\rightarrow$. Нека

$$a_1, a_2, \ldots, a_m$$

е фиксирано изреждане на символите от

$$\Gamma \cup \{\triangleright, \leftarrow, \rightarrow\},$$

като

$$a_1 = \triangleright,\ a_2 = \rightarrow .$$

Нека $\mathcal{M} = (K, \Gamma, \delta, s, H)$ е машина на Тюринг. Ще казваме, че четворката

$$(q, x, q', y)$$

е преход на $\mathcal{M}$, ако

$$\delta(q, x) = (q', y)$$

Тъй като функцията δ е крайна (има крайна дефиниционна област $(K - H) \times (\Gamma \cup \{\triangleright\})$), то тя може да бъде описана чрез крайния списък от всички преходите на $\mathcal{M}$. Да отбележим, че от този списък можем да възстановим, множеството $K - H$, азбуката Γ, както и онези стопсъстояния h, за които съществува състояние q, буква x и команда y, такива че

$$\delta(q, x) = (h, y).$$

Без ограничения можем да считаме, че H се състои само от такива състояния. Освен това можем да считаме, че състоянията на $\mathcal{M}$ са

$$q_1, q_2, \ldots, q_m,$$

като $s = q_1$, а състояния от H са последните няколко в списъка, като в случай, че $\mathcal{M}$ е разпознавател, то $q_{m-1} = y$ и $q_m = n$. С тези уговорки, машината $\mathcal{M}$ се описва изцяло от списъка с преходите ѝ.

Дефинираме кода $\ulcorner\mathcal{M}\urcorner$ на $\mathcal{M}$ по следната схема. Ще кодираме състоянието q_i чрез думата 10^i (за $1 \leq i \leq n$), а символа a_j чрез думата 10^j (за $1 \leq j \leq m$). Използвайки кодовете на състоянията и символите, кодираме прехода (q_i, a_j, q_k, a_l) чрез думата

$$10^i 10^j 10^k 10^l$$

Нека накрая

$$p_1, p_2, \ldots, p_t$$

е фиксирано изреждане на преходите на $\mathcal{M}$ и нека κ_i е кодът на прехода p_i. Тогава код $\ulcorner\mathcal{M}\urcorner$ на $\mathcal{M}$ е естественото число, което има двоичен запис

$$\kappa_1 \kappa_2 \ldots \kappa_t,$$

ако $\mathcal{M}$ не е разпознавател и

$$1\kappa_1 \kappa_2 \ldots \kappa_t,$$

ако $\mathcal{M}$ е разпознавател.

Пример. Нека $\mathcal{M}$ е машина на Тюринг с $K = \{s, h\}$, $\Gamma = \{0, 1, \sqcup\}$, $H = \{h\}$ и δ се състои от преходите

$$(s, \triangleright, s, \rightarrow), (s, \sqcup, h, 0), (s, 0, h, 1), (s, 1, s, \leftarrow).$$

Нека фиксираното изреждане на символите от $\Gamma \cup \{\triangleright, \leftarrow, \rightarrow\}$ е

$$\triangleright, \rightarrow, \leftarrow, \sqcup, 0, 1.$$

Тогава код на $\mathcal{M}$ е числото с двоичен запис

$$\underbrace{10101010010100001001000001010000010010000001010000001010000}_{(s,\triangleright,s,\rightarrow)\quad(s,\sqcup,h,0)\quad(s,0,h,1)\quad(s,1,s,\leftarrow)}$$

Нека отбележим, че $\ulcorner\mathcal{M}\urcorner$ зависи от изреждането на преходите на $\mathcal{M}$. Така всяка машина има $t!$ кода, където t е броят на преходите на $\mathcal{M}$.

Нека още отбележим, че по даден двоичен запис на естествено число, можем да определим, дали то е код на машина на Тюринг. За целта, ако двоичния запис започва с две единици, изтриваме първата и извършваме следните проверки:

1. Разбиваме думата на думи всяка, от които започва с 1, съдържа 4 единици и всяка 1 е последвана от поне една 0. Ако не можем да го направим, то тогава двоичният запис не е код на машина на Тюринг.

2. Намираме най-големия брой нули n, които следват след първата единица на всяка една от получените думи, както и най-големия брой на нули m, които следват след трета единица на всяка една от получените думи.

   Забележка. Числото n е евентуалният брой на състоянията, които не са стопсъстояния, а m е броят на всички състояния.

3. За всяко $1 \leq i \leq n$ и всяко j, такова че $a_j \neq \leftarrow, \rightarrow$, проверяваме, дали точно една от думите започва с 10^i10^j. Ако не, то числото не е код на машина на Тюринг.

   Забележка. Това е условието δ да бъде функция, дефинирана в $(K - H) \times (\Gamma \cup \{\triangleright\})$.

4. За всяко $1 \leq i \leq n$ проверяваме, дали думата, започваща с 10^i101 е от вида 10^i1010^k100 за някое k. Ако не, то числото не е код на машина на Тюринг.

   Забележка. Това е условието за всяко състояние $q \in K - H$ да бъде изпълнено $\delta(q, \triangleright) = (q', \rightarrow)$.

5. За всяка дума $10^i10^j10^k10^l$ проверяваме дали $a_l \neq \triangleright$. Ако не, то числото не е код на машина на Тюринг.

   Забележка. Това е условието δ да бъде функция в $K \times (\Gamma \cup \{\rightarrow, \leftarrow\})$.

6. За всяко $n < k < m$ проверяваме дали има дума от вида $10^i1^0j10^k10^l$. Ако не, то числото не е код на Машина на Тюринг.

   Забележка. Това е допълнителното условие всяко стопсъстояние да бъде трета координата на някой преход.

7. Ако числото е започвало с 11, т.е. то е евентуално код на разпознавател, проверяваме дали $m = n + 2$. Ако не, то числото не е код на Машина на Тюринг.

   Забележка. Ако машината е разпознавател, то тя трябва да има точно две стопсъстояния.

Ако всички проверки са успешни то числото е код на машина на Тюринг. Такива числа ще наричаме истински кодове. Ще считаме, че останалите числа са кодове на машина на Тюринг, която мести безкрайно дълго четящата глава надясно. По този начин всяко естествено число е код на машина на Тюринг.

Теорема 8.9. Съществува машина на Тюринг $\mathcal{M}_U^\Gamma$, която при вход

$$B(\ulcorner\mathcal{M}\urcorner)\$w$$

където $\mathcal{M}$ е машина с азбука Γ, симулира действието на $\mathcal{M}$ върху вход w.

Доказателство. Отново ще опишем неформално поведението на $\mathcal{M}_U^\Gamma$. При вход $u\$w$ първо местим думата u върху втора лента, като на първа лента остава $\triangleright\sqcup w$. След това проверяваме дали u е естествено число в двоичен запис. В случай, че не е, то спираме действието на $\mathcal{M}_U^\Gamma$. Ако $u = B(n)$ за някое n проверяваме, дали $B(n)$ е истински код. Ако не е, пускаме четящата глава на първа лента да се движи безкрайно дълго надясно. В противен случай $n = \ulcorner\mathcal{M}\urcorner$ за някоя машина $\mathcal{M}$. Преобразуваме $B(n)$, замествайки всяка част

$$10^i10^j10^k10^l$$

съответстваща на даден преход, с

$$10^i1a_j10^ka_l$$

Накрая записваме

$$101\sqcup$$

на трета лента и започваме симулацията на $\mathcal{M}$ чрез следния цикъл:

1. Нека върху трета лента е изписана думата 10^i1a_j, като главата на първа лента се намира точно върху символ a_j

2. Търсим върху втора лента преход започващ с 10^i1a_j. Ако такъв няма, то това означава, че сме достигнали до стопсъстояние на $\mathcal{M}$. В този случай, ако $\mathcal{M}$ е разпознавател ($B(n)$ започва с 11), то изчисляваме дали това стопсъстояние е първото или второто по големина на кода, като в случай, че то е първото, прекратяваме изчислението на $\mathcal{M}_U^\Gamma$ и приемаме входа, а във вслучай, че е второто, прекратяваме изчислението на $\mathcal{M}_U^\Gamma$ и отхвърляме входа. В случай, че $\mathcal{M}$ не е разпознавател, просто прекратяваме изчислението на $\mathcal{M}_U^\Gamma$.

3. Нека сме намерили преход $10^i1a_j10^k1a_l$ (тъй като $B(n)$ е истински код, то този преход е единствен).

   (а) Ако $a_l \neq \leftarrow, \rightarrow$, то заместваме символа a_i, който се вижда от главата на първа лента, със символа a_l (без да местим главата), и заместваме думата 10^i1a_j с 10^k1a_l.

   (б) Ако $a_l = \rightarrow$, то преместваме четящата глава на първа лента с една клетка надясно. Нека там е записан символа a_s. Заместваме думата 10^i1a_j с 10^k1a_s.

   (в) Ако $a_l = \leftarrow$, то преместваме четящата глава на първа лента с една клетка наляво. Нека там е записан символа a_s. Заместваме думата 10^i1a_j с 10^k1a_s.

4. Повтаряме цикъла отначало.

□

8.6 Стоппроблем

Неформално стоппроблемът може да се изкаже по следния начин: *Вярно ли е, че машината на Тюринг $\mathcal{M}$ спира върху вход w?* Можем да формализираме стоппроблема, използвайки кодовете на машина на Тюринг. По-точно, нека при фиксирана азбука Γ (съдържаща 0, 1 и $\sqcup$ и несъдържаща $\triangleright$, $\rightarrow$ и $\leftarrow$) и непразна $\Sigma \subseteq \Gamma - \{\sqcup\}$ с $\mathcal{H}$ означим множеството

$$\mathcal{H} = \{(n, w) \mid \text{машината } \mathcal{M} \text{ с код } n \text{ спира при вход } w \in \Sigma^*\},$$

т.е. $(n, w) \in \mathcal{H}$ тогава и само тогава, когато за машината $\mathcal{M}$ с код n е в сила

$$(s, \triangleright\underline{\sqcup}) \vdash^*_{\mathcal{M}} (h, \sigma)$$

където s е началното състояние, а h е някое стопсъстояние на $\mathcal{M}$.

Така неформалния проблем се свежда до разрешимостта на множеството $\mathcal{H}$. Ясно, че $\mathcal{H}$ е полуразрешимо чрез универсалната машина $\mathcal{M}^\Gamma_U$. Наистина при вход (n, w) машината $\mathcal{M}^\Gamma_U$ симулира работата на машината $\mathcal{M}$ с код n върху входа w и значи $\mathcal{M}^\Gamma_U$ спира върху (n, w) тогава и само тогава, когато $\mathcal{M}$ спира върху входа w.

Теорема 8.10. Множестово $\mathcal{H}$ не е разрешимо.

Доказателство. Нека фиксираме $a \in \Sigma$. Да допуснем, че $\mathcal{H}$ е разрешимо и нека $\mathcal{M}_\mathcal{H}$ е машина, която го разрешава. Тъй като Γ съдържа 0 и 1, без ограничение можем да считаме, че азбуката на $\mathcal{M}_\mathcal{H}$ е Γ. [1] Да разгледаме множеството

$$\mathcal{H}_1 = \{a^n \mid (n, a^n) \in \mathcal{H}\}.$$

Тъй като $\mathcal{H}$ е разрешимо, то $\mathcal{H}_1$ също е разрешимо чрез машина, използваща Γ (от a^n можем да възстановим двоичен запис на n, четейки думата отляво надясно и прибавяйки 1 при всяко движение надясно) и следователно множеството

$$\mathcal{H}_1^\partial = \{a^n \mid a^n \notin \mathcal{H}_1\}.$$

също е разрешимо посредством машина, използваща азбуката Γ. Оттук $\mathcal{H}_1$ е полуразрешимо чрез машина с код n_0, която спира точно върху думите от множеството. Тогава

$$a^{n_0} \in \mathcal{H}_1^\partial \iff (n_0, a^{n_0}) \in \mathcal{H} \iff a^{n_0} \in \mathcal{H}_1 \iff a^{n_0} \notin \mathcal{H}_1^\partial,$$

което е невъзможно. Следователно $\mathcal{H}$ не е разрешимо.

Следствие 8.11. Не всяко полуразрешимо множество е разрешимо.

[1] Ако една машина $\mathcal{M}$ работи с азбука $\Gamma' = \{a_1, a_2, \ldots, a_n\}$, то тя може да бъде симулирана от машина, която работи с азбука, съдържаща 0 и 1, като всяка буква $a_i \in \Gamma'$ се замества с думата 10^i.

8.7 Машини с неограничени регистри

Машините с неограничени регистри спадат към клас машини с непоследователен достъп до паметта (RAM-памет). Всяка машина с неограничени регистри се състои от памет и програма. Паметта на машината е разпределена в краен брой регистри, всеки от които има адрес, представляващ естествено число, и в който във всеки един момент (стъпка от изпълнението) се съхранява естествено число. В началото на изпълнението всеки регистър съдържа числото 0. Програмата на машината е крайна редица от редове, индексирани с последователни естествени числа (номера на редовете), съдържащи една от следните команди:

$Z(n)$ //Анулирай регистъра с адрес n. Изпълни следващия ред на програмата.

$S(n)$ //Прибави единица към регистъра с адрес n. Изпълни следващия ред на програмата.

$T(n,m)$ //Копирай съдържанието на регистъра с адрес n в регистъра с адрес m. Изпълни следващия ред на програмата.

$J(n,m,l)$ //Ако съдържанието на регистъра с адрес n съвпада със съдържанието на регистъра с адрес m, изпълни ред с номер l от програмата. В противен случай изпълни следващия ред на програмата.

В случай, че машината стигне до ред от програмата без съдържание, то тя спира изпълнението си. В началото на изпълнението машината получава вход наредена k-орка от естествени числа $(n_1, \ldots, n_k)$, като числата n_1, n_2, $\ldots$, n_k се поставят съответно в регистрите с адреси 1, 2, $\ldots$, k, след което машината започва да изпълнява първия ред на програмата. В случай, че в даден момент машината стигне до празен ред (ред без команда), машината спира и изходът ѝ е съдържанието на регистъра с адрес 0.

Така всяка машина с неограничени регистри изчислява частична функция от $\mathbb{N}^k$ в $\mathbb{N}$.

Пример. Машината с неограничени регистри с програма

1. $Z(3)$
2. $Z(4)$
3. $Z(5)$
4. $T(1,0)$
5. $J(2,3,9)$
6. $S(0)$
7. $S(3)$
8. $J(4,5,5)$

изчислява функцията $f_+ : \mathbb{N}^2 \to \mathbb{N}$, действаща по правилото

$$f_+(m,n) = m + n.$$

Всяка машина с неограничени регистри може да се използва като подпрограма на друга машина с неограничени регистри. За целта е достатъчно да преномерираме редовете на програмата, така че да няма повторение в номерата на редовете, и да транслираме с подходяща константа адресите на използваните регистри, така че подпрограмата да не използва регистри, които се използват от основната програма. Така например горната програма може да се използва като подпрограма в друга машина, модифицирайки я чрез две контстанти l и r, като l е транслацията на редовете, а r е транслацията на регистрите, а именно:

$l+1.$ $Z(r+3)$
$l+2.$ $Z(r+4)$
$l+3.$ $Z(r+5)$
$l+4.$ $T(r+1, r+0)$
$l+5.$ $J(r+2, r+3, l+9)$
$l+6.$ $S(r+0)$
$l+7.$ $S(r+3)$
$l+8.$ $J(r+4, r+5, l+5)$

Тази подпрограма при вход n и m, разположен в регистри $r+1$ и $r+2$, връща изход $n+m$ в регистър $r+0$ и предава управлението, пращайки към изпълнението на ред $l+9$. Нека означим тази подпрограма с $N_+(l,r)$.

Пример. Машината с програма

1. $Z(3)$
2. $Z(4)$
3. $Z(5)$
4. $Z(6)$
5. $T(2,7)$
6. $T(6,8)$
7. $J(2,3,11)$
8. $N_+(7,6)$
9. $S(3)$
10. $J(4,5,6)$
11. $T(4,0)$

изчислява функцията $f_\times : \mathbb{N}^2 \to \mathbb{N}$, действаща по правилото

$$f_\times(m,n) = mn.$$

Считаме, че една машина с неограничени регистри изчислява множеството $A \subseteq \mathbb{N}^k$, ако тя изчислява характеристичната функция χ_A на A, където $\chi_A : \mathbb{N}^k \to \{0,1\}$ действа по правилото

$$\chi_A(n_1,\ldots,n_k) = \begin{cases} 1, & (n_1,\ldots,n_k) \in A; \\ 0, & (n_1,\ldots,n_k) \notin A. \end{cases}$$

По този начин, освен функции, машините с неограничени регистри изчисляват и релации между естествени числа.

Пример. Релацията $\leq$ се изчислява от машината

1. $Z(3)$
2. $Z(4)$
3. $Z(5)$
4. $Z(6)$
5. $J(1,3,9)$
6. $J(2,4,10)$
7. $S(3)$
8. $S(4)$
9. $S(0)$

Пример. Функцията $f_{\dot{-}} : \mathbb{N}^2 \to \mathbb{N}$, действаща по правилото $f_{\dot{-}}(m,n) = m \dot{-} n$, където

$$m \dot{-} n = \begin{cases} m-n, & m \geq n; \\ 0, & \text{иначе} \end{cases}$$

се изчислява от машина с неограничени регистри със следното неформално описание:

1. Машината получава входа си в регистри 1 и 2.
2. Проверяваме, дали съдържанието на регистър 2 е по-голяма от съдържанието на регистъра 1. Ако да прекратяваме изпълнението.
3. Използваме регистър 3 за брояч. В началото той е нула.
4. Използваме регистър 4 за съхранение на сумата на регистри 2 и 3. Преписваме съдържанието на регистър 2 в регистър 4.

5. Започваме следния цикъл:
   (а) Проверяваме, дали съдържанието на регистър 4 е равно на съдържанието на регистър 1. Ако да прекратяваме изпълнението на цикъла.
   (б) Увеличаваме съдържанието на регистър 3 с 1.
   (в) Увеличаваме съдържанието на регистър 4 с 1.
   (г) Започваме цикъла отначало.
6. Прехвърляме съдържанието на регистър 3 в регистър 0 и спираме работа.

Пример. Функцията $Div : \mathbb{N}^2 \to \mathbb{N}$, която по дадени (m, n) изчислява цялата част при деленето на m на n (считаме, че цялата част при делене на 0 е 0) се изчислява от машина с неограничени регистри със следното неформално описание:

1. Машината получава входа си в регистри 1 и 2.
2. Проверяваме, дали съдържанието на регистър 2 е 0. Ако да прекратяваме изпълнението.
3. Използваме регистър 3 за брояч. В началото той е нула.
4. Използваме регистър 4 за съхранение на произведението на регистри 2 и 3. В началото той е нула.
5. Започваме следния цикъл:
   (а) Проверяваме, дали съдържанието на регистър 4 е по-голямо от съдържанието на регистър 1. Ако да прекратяваме изпълнението на цикъла.
   (б) Увеличаваме съдържанието на регистър 3 с 1.
   (в) В регистър 4 записваме сумата на съдържанията на регистър 2 и регистър 4.
   (г) Започваме цикъла отначало.
6. Вадим 1 от съдържанието на регистър 3.
7. Прехвърляме съдържанието на регистър 3 в регистър 0 и спираме работа.

Пример. Функцията $Rem : \mathbb{N}^2 \to \mathbb{N}$, която по дадени (m, n) изчислява остатъка при деленето на m на n (считаме, че цялата част при делене на 0 е делимото), защото

$$Rem(m, n) = m \dot{-} nDiv(m, n)$$

Теорема 8.12. Всяка машина с неограничени регистри може да бъде симулирана от машина на Тюринг

Доказателство. Нека е дадена машина с неограничени регистри $\mathcal{N}$, която използва r регистъра. Ще симулираме работата на $\mathcal{N}$ чрез машина на Тюринг $\mathcal{M}$ с r ленти, работеща над азбука $\Gamma = \{1, \sqcup\}$. Всяка лента на $\mathcal{M}$ съответства на един от регистрите на $\mathcal{N}$. При това, съдържание на k-ти регистър равно на n съответства на съдържание $\triangleright \sqcup 1^n$ на k-та лента. Машината $\mathcal{M}$ разполага с главни състояния $Q_s, \ldots, Q_{s+l}$ съответстващи на редовете на програмата на $\mathcal{N}$ (включително първия празен ред след края ѝ), като всяко главно състояние има свои собствени спомагателни състояния. Началното състояние на машината е Q_s, а стопсъстояния са всички главни състояния, съответстващи на празни редове.

Машината $\mathcal{M}$ се държи така, че при изпълнение на всяко главно състояние, четящата глава на всяка лента да се намира върху празната клетка непосредствено до левия ѝ край. Главното състоянието Q_i действа по следния начин (чрез помощните си състояния):

1. Ако i-тият ред на $\mathcal{N}$ е $Z(n)$, то Q_i пуска машина, която изтрива n-тата лента. Предава управлението на състояние Q_{i+1}.
2. Ако i-тият ред на $\mathcal{N}$ е $S(n)$, то Q_i пуска машина, която добавя буквата 1 в десния край на думата, изписана на n-та лента. Предава управлението на състояние Q_{i+1}.

3. Ако i-тият ред на $\mathcal{N}$ е $T(m,n)$, то Q_i пуска машина, която изтрива n-тата лента, след което пуска втора машина, която копира съдържанието на m-та лента върху n-та лента. Предава управлението на състояние Q_{i+1}.

4. Ако i-тият ред на $\mathcal{N}$ е $J(m,n,s)$, то Q_i пуска машина, която проверява дали съдържанието на m-тата лента съвпада с това на n-тата лента. Ако проверката е успешна, то предава управлението на състояние Q_s, а в противен случай — на състояние Q_{i+1}.

□

Теорема 8.13. Работата на всяка машина на Тюринг може да бъде симулирана от машина с неограничени регистри.

Доказателство. Нека $\mathcal{M} = (K, \Gamma, \delta, s, H)$ е еднолентова машина на Тюринг. Нека фиксираме изреждане

$$a_0, a_1, \ldots, a_{p-1}$$

на елементите на $\Gamma \cup \{\triangleright\}$, като $a_0 = \sqcup$. Тъй като машините с неограничени регистри работят с естествени числа, а не с произволна азбука, трябва първо да кажем, как ще кодираме думите над азбуката Γ. Възползваме се от това, че за всяко $p \geq 2$ и всяко естествено n съществува единствено $s \geq 0$ и числа $0 \leq k_0, k_1, \ldots, k_s < p$ такива, че

$$n = k_0 p^s + k_1 p^{s-1} + \ldots + k_{s-1} p + k_s \text{ (p-ичен запис на n)}$$

На буквата a_i съпоставяме числото $\overline{a_i} = i$. Тогава функцията, която на думата $x_1 x_2 \ldots x_t \in (\Gamma \cup \{\triangleright\})^*$ съпоставя числото

$$\overline{x_1 x_2 \ldots x_t} = \overline{x_0} p^t + \overline{x_1} p^{t-1} + \ldots + \overline{x_{t-1}} p + \overline{x_t}$$

е биективна.

Машината с неограничени регистъра $\mathcal{N}$, симулираща $\mathcal{M}$ ще използва регистри 1 и 2 за да съхранява, записаното върху лентата на $\mathcal{M}$, както и позицията на четящата глава, по следната схема: нека върху лентата е записана думата

$$x_1 x_2 \ldots x_{m-1} \underline{x_m} x_{m+1} \ldots x_n$$

където x_i са букви, като $x_1 = \triangleright$, а в случай, че $x_n = \sqcup$, то $n = m+1$. Тогава в регистър 1 ще съхраняваме числото

$$\overline{x_1 x_2 \ldots x_{m-1} x_m},$$

а в регистър 2 числото

$$\overline{x_n x_{n-1} \ldots x_{m+1}}.$$

Да забележим, че

$$\begin{aligned} Div\,(\overline{x_1 x_2 \ldots x_{m-1} x_m}, p) &= \overline{x_1 x_2 \ldots x_{m-1}} \\ Rem\,(\overline{x_1 x_2 \ldots x_{m-1} x_m}, p) &= \overline{x_m} \\ \overline{x_1 x_2 \ldots x_{m-1} x_m}.p &= \overline{x_1 x_2 \ldots x_{m-1} x_m a_0} \end{aligned}$$

Нека състоянията на $\mathcal{M}$ са $q_0, q_1, \ldots, q_l$, където $q_0 = s$ и за някое k е в сила $K - H = \{q_0, q_1, \ldots, q_k\}$. Програмата на $\mathcal{N}$ е разделена на $k+1$ блока, като i-тия блок съответства на състоянието q_i. Всеки блок е разделен на p подблока, които съответстват на p-те символа на Γ. Описваме действието на i-тия блок по следния неформален начин:

1. Записваме остатъкът от деленето на съдържанието на регистър 1 на p (т.е. $\overline{x_m}$ — това което се вижда от четящата глава на $\mathcal{M}$) в регистър 3.

2. Анулираме регистър 4. Ако съдържанието му съвпада с това на регистър 3, предаваме изпълнението на подблок 0. Иначе добавяме 1 към регистър 4. Ако съдържанието на регистри 3 и 4 съвпада, то предаваме управлението на подблок 1. Иначе добавяме 1 към регистър 4 и така нататък (общо p пъти). По този начин, коректно разпознаваме поредността на буквата x_m в списъка

$$a_0, a_1, \ldots, a_{p-1}.$$

3. Подблокът j, отговаря за това да бъде изпълнен прехода $\delta(q_i, a_j)$ на $\mathcal{M}$. Нека $\delta(q_i, a_j) = (q_{i'}, y)$. Възможни са три случая:

- $y = a_{j'}$ за някое $0 \leq j' \leq p-1$.
  (а) Записваме в регистър 1 частното при делене на p на съдържанието на регистър 1 (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_{m-1}}$).
  (б) Записваме в регистър 1 произведението на съдържанието на регистър 1 и p (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_{m-1} a_0}$).
  (в) Записваме в регистър 1 сумата на съдържанието на регистър 1 и $\overline{a_{j'}}$ (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_{m-1} a_{j'}}$).
  (г) Предаваме изпълнението на блок i'.
- $y = \rightarrow$.
  (а) Записваме в регистър 4 остатъкът при деленето на съдържанието на регистър 2 на p (т.е. $\overline{x_{m+1}}$).
  (б) Записваме в регистър 2 частното при делене на p на съдържанието на регистър 2 (така регистър 2 съдържа $\overline{x_n x_{n-1} \dots x_{m+2}}$).[2]
  (в) Записваме в регистър 1 произведението на съдържанието на регистър 1 и p (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_{m-1} x_m a_0}$).
  (г) Записваме в регистър 1 сумата на съдържанието на регистър 1 и регистър 4 (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_m x_{m+1}}$).
  (д) Предаваме изпълнението на блок i'.
- $y = \leftarrow$.
  (а) Записваме в регистър 1 частното при делене на p на съдържанието на регистър 1 (така регистър 1 съдържа $\overline{x_1 x_2 \dots x_{m-1}}$).
  (б) Записваме в регистър 2 произведението на съдържанието на регистър 2 и p (така регистър 2 съдържа $\overline{x_n x_{n-1} \dots x_{m+1} a_0}$).
  (в) Записваме в регистър 2 сумата на съдържанието на регистър 2 и регистър 3 (така регистър 2 съдържа $\overline{x_n x_{n-1} \dots x_{m+1} x_m}$).
  (г) Предаваме изпълнението на блок i'.

□

Тезис на Чърч. Машините на Тюринг са най-общото ефективно алгоритмично средство.

[2] В случай, че $n = m+1$, то при делене на p се получава частно $0 = \overline{a_0}$