Разпознаване и изчисляване с помощта на машини на Тюринг

Най-напред ще покажем как машините на Тюринг се използват както като генератори така и като разпознаватели. Ако е дадена една машина на Тюринг $M=(K,\Sigma,\delta,s,H)$ и $w\in(\Sigma\setminus\{\triangleright,\sqcup\})^*$; то началната конфигурация (стандартна) ще бъде конфигурация от вида $(s,\triangleright\underline{\sqcup}w)$. Стандартните стоп-конфигурации ще бъдат $(y,\triangleright u\underline{a}v)$ или $(n,\triangleright u\underline{a}v)$ като първата конфигурация е приемаща, а втората отхвърляща. y и n са съкращения на "yes" и "no", т.е. ако питаме дали w се приема от M, то y означава, че w се приема, а n че се отхвърля. Стоп-конфигурациите при машините на Тюринг при които изчисляване или генериране съответно функция или множество ще има вида $(h,u\underline{a}v)$, където $h\in H$.

Д. Нека $M=(K,\Sigma,\delta,s,H)$ е машина на Тюринг, където $H=\{y,n\}$. Казваме, че думата $w\in(\Sigma\setminus\{\triangleright,\sqcup\})^*$ се приема от M, ако $(s,\triangleright\underline{\sqcup}w)\vdash_M^*(y,u\underline{a}v)$ и че w се отхвърля от M, ако $(s,\triangleright\underline{\sqcup}w)\vdash_M^*(n,u\underline{a}v)$. За всяко Σ_0 такова, че $\Sigma_0\subseteq\Sigma\setminus\{\triangleright,\sqcup\}$, Σ_0 се нарича входна азбука.

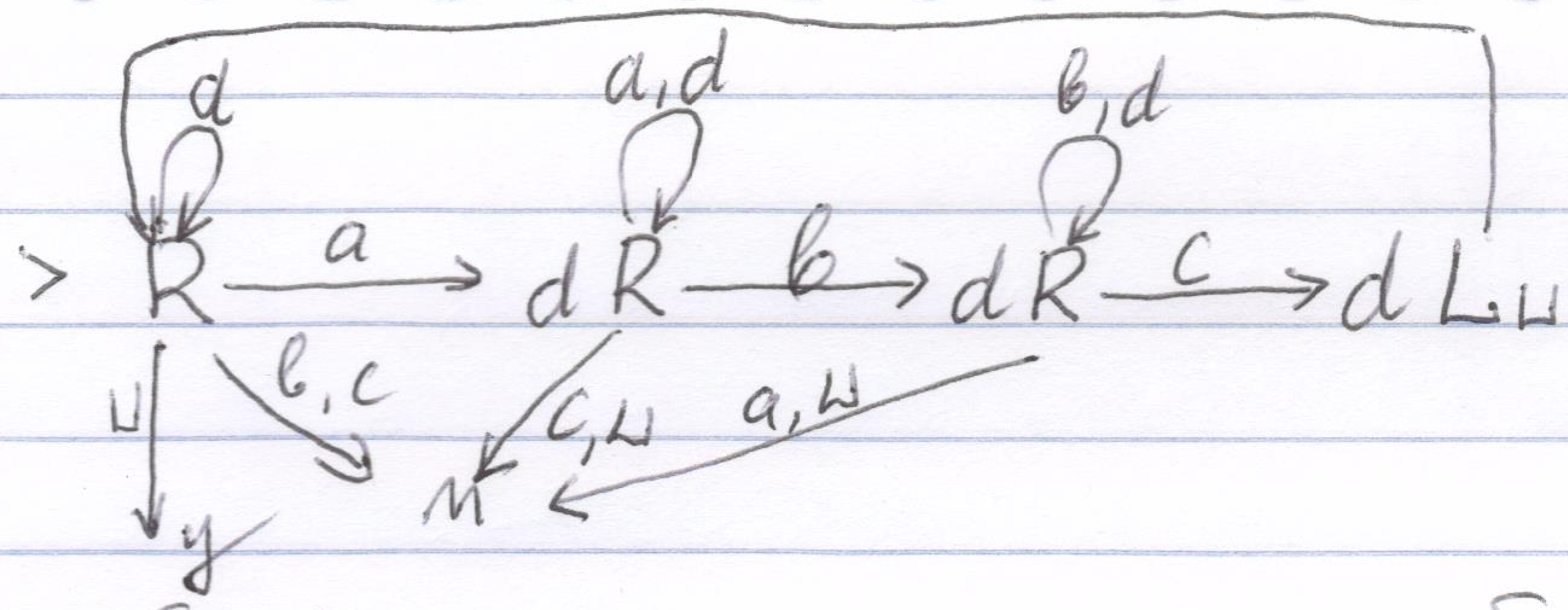
Д. Казваме, че $L\subseteq\Sigma_0^*$ се разрешава (разпознава) от дадена машина на Тюринг $M=(K,\Sigma,\delta,s,H)$, където $H=\{y,n\}$, ако за всяко $w\in\Sigma_0^*$ ($\Sigma_0\subseteq\Sigma\setminus\{\triangleright,\sqcup\}$) са изпълнени следните две условия:

а) Ако $w\in L$, то w се разпознава от M;

б) Ако $w\notin L$, то w се отхвърля от M.

Д. Един език L се нарича рекурсивен (разрешим), ако съществува машина на Тюринг M, която го разрешава.

Когато говорехме за стекови автомати ние отбелязахме, че езикът $L=\{a^nb^nc^n \mid n\in\mathbb{N}\}$ не е КСЕ. Сега ще видим, че той е рекурсивен (разрешим). За тази цел ще построим машина на Тюринг M, която го разрешава (разпознава).



Да отбележим, че тук към азбуката $\Sigma_0=\{a,b,c\}$ сме добавили една нова буква d, която се нарича помощна. Това е смисълът на $\Sigma_0 \subseteq \Sigma \setminus \{\triangleright,\sqcup\}$. В случая към Σ сме добавили d и можем да добавим още помощни букви. Алгоритъмът, описан от горната машина на Тюринг работи както следва: Когато започне да чете думата, проверява дали тя започва с a, b, c. Ако не започва с a, думата се отхвърля. Ако срещне буквата a той я замества с d и по този начин „отброява" 1 срещане на a и след това прескача всичките a докато срещне b. Ако не срещне b, думата се отхвърля. Ако срещне b, замества го с d и „отброява" 1 срещане на b, а след това прескача всички b докато стигне c. Ако не срещне c, думата се отхвърля, а в противен случай — ако срещне c, „отброява" 1 c и се връща в началото за ново „отброяване" на a-тата, b-тата и c-тата. Да разгледаме един пример: Да проверим дали $a^2b^2c^2$ се приема.

$\triangleright\underline{\sqcup}a^2b^2c^2 \xrightarrow{R} \triangleright\sqcup\underline{a}ab^2c^2 \xrightarrow{d} \triangleright\sqcup\underline{d}ab^2c^2 \xrightarrow{R} \triangleright\sqcup d\underline{a}b^2b^2$
$\xrightarrow{R} \triangleright\sqcup da\underline{b}bc^2 \xrightarrow{d} \triangleright\sqcup da\underline{d}bc^2 \xrightarrow{R} \triangleright\sqcup dad\underline{b}c^2 \xrightarrow{R} \triangleright\sqcup dadb\underline{c}c$
$\xrightarrow{d} \triangleright\sqcup dadb\underline{d}c \xrightarrow{L_\sqcup} \triangleright\underline{\sqcup}dadbdc \xrightarrow{R} \triangleright\sqcup\underline{d}adbdc \xrightarrow{R}$
$\triangleright\sqcup d\underline{a}dbdc \xrightarrow{d} \triangleright\sqcup d^2\underline{d}dbdc \xrightarrow{R} \triangleright\sqcup d^2\underline{d}bdc \xrightarrow{R} \triangleright\sqcup d^3\underline{b}dc$
$\xrightarrow{d} \triangleright\sqcup d^3\underline{d}dc \xrightarrow{R} \triangleright\sqcup d^4\underline{d}c \xrightarrow{R} \triangleright\sqcup d^5\underline{c} \xrightarrow{d} \triangleright\sqcup d^5d \xrightarrow{L_\sqcup}$
$\triangleright\underline{\sqcup}d^6 \xrightarrow{R} \ldots \xrightarrow{R} \triangleright\sqcup d^6\underline{\sqcup} \longrightarrow (y)$ думата се приема.

Вижте самостоятелно как работи върху някои от думите ab^2c^2, a^2bc^2, a^2b^2c.

D. Нека $M=(K,\Sigma,\delta,s,\{h\})$ е машина на Тюринг и $\Sigma_0 \subseteq \Sigma \setminus \{\triangleright,\sqcup\}$. Казваме, че функцията $f: \Sigma_0^* \to \Sigma_0^*$ се изчислява с помощта на M (или още M изчислява f) ако за произволно $w \in \Sigma_0^*$ са изпълнени условията:

а) Ако $f(w)=y$ за $y\in\Sigma_0^*$, то $(s, \triangleright\underline{\sqcup}w) \vdash_M^* (h, \triangleright\underline{\sqcup}y)$;
б) Ако $(s, \triangleright\underline{\sqcup}w) \vdash_M^* (h, \triangleright\underline{\sqcup}y)$ за $y\in\Sigma_0^*$, то $f(w)=y$.
Всеки път когато $(s, \triangleright\underline{\sqcup}w) \vdash_M^* (h, \triangleright\underline{\sqcup}y)$ и $w, y\in\Sigma_0^*$ w се нарича вход, а y резултат (изход) и често още ще пишем $M(w)=y$ (казваме M върху входа w дава изход (резултат) y). Обърнете, че $M(w)$ е определено само ако $(s, \triangleright\underline{\sqcup}w) \vdash_M^* (h, \triangleright\underline{\sqcup}y)$ за $y\in\Sigma_0^*$. В противен случай се казва, че M не е определено върху входа w. Тук ние дадохме дефиниция за изчислимост на функция в най-общия смисъл, когато $f: \Sigma_0^* \to \Sigma_0^*$. По-късно ще определим изчислима (рекурсивна) функция в по-тесен смисъл.
Когато за една машина на Тюринг $M=(K, \Sigma, \delta, s, H)$ и $w\in\Sigma_0^*$, $\Sigma_0 \subseteq \Sigma \setminus \{\triangleright, \sqcup\}$ е изпълнено, че $(s, \triangleright\underline{\sqcup}w) \vdash_M^*$ $(h, u\underline{a}v)$, ние казваме, че M спира работа върху входа w и пишем още $M(w)\searrow$. Ако M не спира работа върху w (т.е. никога не стига до стоп-състояние при входа w) пишем $M(w)\nearrow$.

Рекурсивни и рекурсивно номеруеми езици.

Д. Нека $M=(K, \Sigma, \delta, s, H)$ е машина на Тюринг и $\Sigma_0 \subseteq \Sigma\setminus\{\triangleright, \sqcup\}$. Казваме, че M полуразрешава езика L, $L\subseteq\Sigma_0^*$ т.т.к. за всяко $w\in\Sigma_0^*$ следната еквивалентност е изпълнена $w\in L \iff M$ спира работа върху w, т.е. $M(w)\searrow$.
Д. Един език L се нарича рекурсивно номеруем (полуразрешим) т.т.к. съществува машина на Тюринг M, такава, че M полуразрешава L.
Твърдение. Всеки рекурсивен език е полуразрешим (рекурсивно номеруем).
Д-во. Нека L е рекурсивен език и $M=(K, \Sigma, \delta, s, H)$ е машина на Т., която разпознава езика L, където $H=\{y, n\}$. Определяме нова машина на Т. $M'=(K, \Sigma, \delta', s, \{y\})$ и δ' е продължение на δ, такова че $\delta'(n, a)=(n, a)$ за всички $a\in\Sigma$. Така M ще спира работа т.т.к. входа

$w \in L$, с което твърдението е доказано.

Твърдение. Ако L е рекурсивен език, то $\bar{L}$ също е рекурсивен.

Д-во. Да напомня, че с $\bar{L}$ означаваме допълнението на L, т.е. $\bar{L} = \Sigma_0^* \setminus L$. Нека $M = (K, \Sigma, \delta, s, H)$, $H = \{y, n\}$ е машина на Тюринг, която разпознава L. Определяме нова машина на Тюринг $M' = (K, \Sigma, \delta', s, \{y, n\})$ както следва променяме δ само за тези двойки (σ, a), за които $\delta(\sigma, a) = (h, b)$, където $h \in \{y, n\}$ и за тях определяме така Ако $\delta(\sigma, a) = (y, b)$, то $\delta'(\sigma, a) = (n, b)$ и ако $\delta(\sigma, a) = (n, b)$, то $\delta'(\sigma, a) = (y, b)$, т.е. всички думи за които M дава отговор y, M' дава отговор n, и обратно, ако M дава отговор n, M' дава отговор y. Очевидно M' разпознава допълнението $\bar{L}$ на L.

Естествен е въпроса каква е връзката м/у рекурсивните и рекурсивно номеруемите езици. Вярно ли е, че всеки рекурсивно номеруем език е рекурсивен? Отговорът е НЕ!

Твърдение. Съществува рекурсивно номеруем език, който не е рекурсивен.

Д-вото ще направим по-късно! Засега не сме готови.

Примитивно рекурсивни функции.

Примитивна рекурсивност на някои функции

Ние споменахме по-рано, когато казахме причините за разглеждането на машина на Тюринг, че машините на Тюринг са тежка техническа конструкция за да се „програмира“ и да се описват алгоритми на него. Въпреки че машината на Тюринг е единствения известен модел за оценяване на сложността на алгоритмите, ние не можем да си позволим да „програмираме“ и да описваме алгоритми на „машинно“ ниво. Това е много подходящо за компютър, но не и за човек. Човек не е способен да извършва много прости операции, но за сметка на това той има възможност да описва алгоритми с по-сложни конструкции. Един такъв модел е моделът на частично рекурсивните функции, които разглеждаме тук. Като начало ще започнем с един подклас на частично рекурсивните функции, а именно примитивно рекурсивните функции (ПРФ).

<u>Основни ПРФ</u>: $O(x)=0$, $S(x)=x+1$, $I^n_i(x_1,\dots,x_n)=x_i$, $1\le i\le n$, $x, x_1,\dots,x_n\in \mathbb{N}$, т.е. Основни са O, S, I^n_i, $1\le i\le n$.

В множеството на всички функции (може и частично дефинирани) ще определим следните операции: суперпозиция и примитивна рекурсия.

Д. Нека $f: \mathbb{N}^n \to \mathbb{N}$ и $g_i: \mathbb{N}^k \to \mathbb{N}$, $i=1,\dots,n$. Казваме, че функцията h се получава от $f, g_1,\dots,g_n$ с помощта на операцията суперпозиция, ако $h: \mathbb{N}^k\to\mathbb{N}$ и за всички естествени числа $x_1,\dots,x_k$ е изпълнено равенството $h(x_1,\dots,x_k)=f(g_1(x_1,\dots,x_k),\dots,g_n(x_1,\dots,x_k))$.

Да отбележа, че равенството е условно, т.е. ако лявата страна е дефинирано, то и дясната страна е дефинирана и обратно, и, когато са дефинирани имат равни стойности.

Д. Нека $f: \mathbb{N}\to\mathbb{N}$ и $g: \mathbb{N}^3\to\mathbb{N}$. Казваме, че функцията $h: \mathbb{N}^2\to\mathbb{N}$ се получава от f и g с помощта на операцията примитивна рекурсия (ПР), ако за произволни $x, y\in\mathbb{N}$ е изпълнена системата

$$\begin{cases} h(x,0)=f(x) \\ h(x,y+1)=g(x,y,h(x,y)). \end{cases}$$

Обърнете внимание, че ако искаме да пресметнем $h(x,3)$ трябва да пресметнем първо $h(x,0)=f(x)$, след това $h(x,1)$, след това $h(x,2)$ и тогава $h(x,3)$.

Ще дадем индуктивна дефиниция на понятието примитивно рекурсивна функция.

Д. а) Всички основни ф-ции O, S, I^n_i, $1\le i\le n$, са ПРФ;

б) Ако $f, g_1,\dots,g_n$ са ПРФ, то и тяхната суперпозиция, която стандартно означаваме с $h=f(g_1,\dots,g_n)$ също е ПРФ;

в) Ако f, g са ПРФ, то и ф-цията h, която се получава от тях с помощта на ПР също е ПРФ.

Твърдение. Следните функции са ПРФ.

а) Всяка константа $C_a(x) = a$, за $x \in N$;

б) $x+y$

в) $x.y$

г) x^y

д) $$P(x) = \begin{cases} x-1, & \text{ако } x>0, \\ 0, & \text{ако } x=0; \end{cases}$$

е) $\Pi(x,y) = 2^x(2y+1)-1$.

Д-во. а) С индукция относно a. За $a=0$ $C_0(x) = O(x)$. Допускаме, че $C_a(x)$ е ПРФ ще го докажем за $a+1$. Наистина $C_{a+1}(x) = a+1 = C_a(x)+1 = S(C_a(x))$.

б) Нека $h_1(x,y) = x+y$. Определяме системата

$h_1(x,0) = x+0 = x = I_1^1(x)$

$h_1(x,y) = x+(y+1) = (x+y)+1 = S(x+y) = S(h_1(x,y)) = g_1(x,y,h_1(x,y))$, където $g_1(x,y,z) = S(z) = S(I_3^3(x,y,z))$

I_1^1 - ПРФ, g_1 е ПРФ, а оттам и $h_1(x,y)$ е ПРФ.

в) Нека $h_2(x,y) = x.y$. Определяме системата

$$\begin{cases} h_2(x,0) = x.0 = 0 = O(x) \\ h_2(x,y+1) = x.(y+1) = xy+x = h_2(x,y)+x = h_1(h_2(x,y),x) = g_2(x,y,h_2(x,y)) \end{cases}$$

$g_2(x,y,z) = h_1(z,x) = h_1(I_3^3(x,y,z), I_1^3(x,y,z))$

Следователно, h_2 е ПРФ

г) $h_3(x,y) = x^y$. Нека напомним, че по деф. $x^0 = 1$.

Тогава $h(x,0) = x^0 = 1 = C_1(x)$

$h_3(x,y+1) = x^{y+1} = x^y.x = h_3(x,y).x = h_2(h_3(x,y),x) = g_3(x,y,h_3(x,y))$. Аналогично на по-горе можем да намерим ПРФ g_3, откъдето h_3 е ПРФ.

Преди да направим д) ще дадем една дефиниция

Д. Нека $g: N^2 \to N$ и $a \in N$. Казваме, че $h: N \to N$ се получава от a, g с помощта на проста схема на ПР, ако $h(0) = a$ и $h(x+1) = g(x, h(x))$ за $x \in N$.

Всъщност чрез простата схема на ПР ние получаваме така наречените рекурентни редици.

Лема. Ако $a \in N$ и $g: N^2 \to N$ е ПРФ, то и ф-цията h, която се получава от a, g с пом. на проста схема на ПР, също е ПРФ.

Д-во. Наистина, нека $g'(x,y,z) = g(x,z)$ и да определим

$$\begin{cases} h'(y,0) = C_a(x) \\ h'(y,x+1) = g'(y,x,h'(x,y)) \end{cases}$$

Очевидно g' е ПРФ и C_a е ПРФ, откъдето h' е ПРФ. Тогава очевидно $h'(x, O(x)) = h(x) = h'(y,x)$, с което лемата е доказана.

д) $P(0) = 0$ и $P(x+1) = x = I^2_1(x, P(x))$.

Следователно P е ПРФ.

е) $\Pi(x,y) = 2^x.(2y+1)-1$ очевидно е ПРФ, като суперпозиция на ПРФ. Ние ще обърнем внимание на Π, защото тя играе роля на кодираща функция. Тя кодира всички наредени двойки естествени числа с естествени числа. Тя отговаря на всички условия за кодиране: 1) Различните двойки естествени числа имат различни кодове. Наистина, нека $(x_1,y_1) \neq (x_2,y_2)$. Да допуснем, че $\Pi(x_1,y_1) = \Pi(x_2,y_2)$. Тогава $2^{x_1}(2y_1+1)-1 = 2^{x_2}(2y_2+1)-1$, т.е. $2^{x_1}(2y_1+1) = 2^{x_2}(2y_2+1)$. Тогава очевидно $2^{x_1} = 2^{x_2}$ и $x_1 = x_2$ и $2y_1+1 = 2y_2+1$, откъдето $y_1 = y_2$. Следователно $(x_1,y_1) = (x_2,y_2)$, т.е. получихме противоречие с $(x_1,y_1) \neq (x_2,y_2)$. Следователно, $\Pi(x_1,y_1) \neq \Pi(x_2,y_2)$. Освен това всяко естествено число е код на наредена двойка. Наистина, нека $z \in N$. Търсим $x,y \in N$ такива, че $\Pi(x,y) = z$, т.е. $2^x(2y+1)-1 = z$, откъдето $z+1 = 2^x(2y+1)$. Разлагане $z+1>0$ на прости множители от вида $2^x.y'$, y' – нечетно и от y' намиране y. Видяхме, че има алгоритъм за намиране на x и y от z. Така видяхме, че това кодиране отговаря на следните условия: а) обратимост; б) ефективност; в) тоталност (не е задължителна). За по-нататъчните разглеждания ще определим и ф-ците $\Pi_1, \Pi_2, \Pi_3, \ldots$, такива, че $\Pi_k : N^k \to N$ и Π_k ще кодира наредените k-орки, по следния индуктивен начин: $\Pi_1(x) = x$ и $\Pi_{k+1}(x_1, \ldots, x_k, x_{k+1}) = \Pi(\Pi_k(x_1, \ldots x_k), x_{k+1})$. Така всяко естествено число се явява код на наредена k-орка за фиксирано k.

Отново Π_k отговаря на всичките условия за кодиране. Ще имаме нужда още и да кодираме всички крайни редици от естествени числа.

Д. Нека $\langle x_0, \dots, x_k \rangle$ е крайна редица от естествени числа. Кодът на $\langle x_0, \dots, x_k \rangle$ ще означаваме с $\tau(\langle x_0, \dots, x_k \rangle)$ и $\tau(\langle x_0, \dots, x_k \rangle) = \Pi(k, \Pi_{k+1}(x_0, \dots, x_k))$.

Зеб. Изображението τ е ефективно, инективно и тотално. Освен това съществуват ПРФ, които намират дължината на редицата и всеки член на тази редица. Това е вярно и за Π, Π_k.

Изчислимост на ПРФ.

Когато говорим за изчислимост на ПРФ ще трябва да посочим каква азбука ще разглеждаме и как ще интерпретираме естествените числа в машините на Тюринг, т.е. ние ще говорим за изчислимост на тези функции с помощта на Машина на Тюринг. В такива случай Σ_0 ще бъде $\{1\}$ и естественото число n ще интерпретираме в Σ_0 като думата 1^n. Сега ще дадем по-специална дефиниция за изчислимост на функция, дефинирана в естествените числа, с машина на Тюринг.

Д. Нека $f: N^n \to N$. Казваме, че машината на Тюринг $M = (K, \Sigma, \delta, s, H)$, $\Sigma_0 \subseteq \Sigma$ изчислява ф-цията f, ако за произволни естествени числа $k_1, \dots, k_n$ са изпълнени условията:

а) Ако $f(k_1, \dots, k_n) = k$, то $M(1^{k_1} \sqcup 1^{k_2} \sqcup \dots \sqcup 1^{k_n}) \downarrow$ и $M(1^{k_1} \sqcup \dots \sqcup 1^{k_n}) = 1^k$, с други думи $(s, \triangleright \underline{\sqcup} 1^{k_1} \sqcup \dots \sqcup 1^{k_n}) \vdash_M^* (h, \triangleright \underline{\sqcup} 1^k)$ за $h \in H$;

б) Ако $M(1^{k_1} \sqcup \dots \sqcup 1^{k_n}) \downarrow$, то $f(k_1, \dots, k_n)$ е дефинирана.

Да обърна внимание, че наредената n-орка $k_1, \dots, k_n$ в машината на Тюринг интерпретираме с $1^{k_1} \sqcup \dots \sqcup 1^{k_n}$ и за нас той е обичайния вход. За простота тук използваме разделител $\sqcup$ вместо „,". Причината е, че вече имаме функции $L_\sqcup$ и $R_\sqcup$ и вместо да определяме аналогични за „," си упростяваме работата.

Освен това ще обобщим и понятието вход и изход. Нека $M=(K,\Sigma,\delta,s,H)$ е машина на Тюринг. С цел удобство когато имаме $(s, u\underline{\sqcup}w) \vdash_M^* (h, u'\underline{\sqcup}w')$ където u, u' са произволни думи в азбуката Σ, а w, w' са от $1^{k_1}\sqcup 1^{k_2}\sqcup\ldots\sqcup 1^{k_n}$, 1^k съответно. Причината е, че често можем да пазим информация, която ни е нужна в u и в u' да има нещо непотребно за резултата. Тогава за w ще говорим за вход и за w' като за изход(резултат).

Твърдение. Всички основни ПРФ са изчислими с машина на Тюринг.

Д-во. а) $O(x)=0$

$\triangleright\underline{\sqcup}1^x \xrightarrow{R_\sqcup} \triangleright\sqcup 1^x\underline{\sqcup} \xrightarrow{D\leftarrow} \triangleright\underline{\sqcup}$,

т.е. машината на Тюринг $R_\sqcup D\leftarrow$ изчислява ф. O.

б) $S(x)=x+1$

$\triangleright\underline{\sqcup}1^x \xrightarrow{R_\sqcup} \triangleright\sqcup 1^x\underline{\sqcup} \xrightarrow{1} \triangleright\sqcup 1^x\underline{1} \xrightarrow{L_\sqcup} \triangleright\underline{\sqcup}1^{x+1}$,

т.е. машината $R_\sqcup 1 L_\sqcup$ изчислява S.

в) $I^n_i(x_1,\ldots,x_n)=x_i$, $1\le i\le n$,

$\triangleright\underline{\sqcup}1^{x_1}\sqcup\ldots\sqcup 1^{x_n} \xrightarrow{R^n_\sqcup} \triangleright\sqcup 1^{x_1}\sqcup\ldots\sqcup 1^{x_n}\underline{\sqcup} \xrightarrow{(D\leftarrow)^{n-i}}$

$\triangleright\sqcup 1^{x_1}\sqcup\ldots\sqcup 1^{x_i}\underline{\sqcup} \xrightarrow{L_\sqcup} \triangleright\sqcup 1^{x_1}\sqcup\ldots\underline{\sqcup}1^{x_i}$,

т.е. $R^n_\sqcup (D\leftarrow)^{n-i} L_\sqcup$ изчислява I^n_i.