COMPUTABILITÀ

Appunti di Andrea Domenico Giuliano
Anno accademico 2015/2016

01 Introduzione

Algoritmo: descrizione di passi elementari per trasformare dei dati input in dei dati output

Passi Elementari: operazione che può essere eseguita meccanicamente, che non richiede intelligenza.

L'algoritmo definisce una funzione che mappa l'input nell'output.

1ª definizione di funzione calcolabile:
Una funzione e' calcolabile se esiste un algoritmo che la calcola.

A noi interessa sapere che questo algoritmo esista, non quale esso sia.

Modello di calcolo (ed algoritmi) effettivi:

un algoritmo per poter essere eseguibile deve avere le seguenti caratteristiche:

- limite finito al numero ed alla complessità delle istruzioni.
- essere eseguito su un agente di calcolo con le seguenti caratteristiche:
  - memoria:
    - usata per memorizzare input, output e risultati intermedi
    - non ha un limite superiore
  - calcola per passi discreti e deterministici.
  - la computazione può finire dopo un numero finito di passi e produrre un output, oppure può non terminare, non rilasciando alcun risultato.

Una macchina con tali caratteristiche non calcolerà tutte le funzioni, questo indica che esistono delle funzioni non calcolabili.

$F = \{f \mid f : N \to N\}$ insieme delle funzioni unarie

Per calcolare un algoritmo dobbiamo fissare un modello di calcolo, che determina la classe di algoritmi disponibili per il calcolo con tale modello.
classe di algoritmi: Å

dato un algoritmo $A \in Å$ si indica con f_A la funzione calcolata

$$F_A = \{f_A \mid A \in Å\} \subset F$$

Sia I il set d'istruzioni (che è finito):

$$Å \subseteq \bigcup_n I^n$$

$|Å|$ cardinalità di $Å$

$Å$ ed $\bigcup_n I^n$ sono enumerabili, poiché hanno una cardinalità finita, ossia $|Å| \leq |N|$

$$|Å| \leq |\bigcup_n I^n| \leq |N|$$

$$|F_Å| \leq |Å| \leq |N|$$

Le funzioni calcolabili sono infinite, ma se messe in confronto con la totalità delle funzioni esistenti sono una piccola parte.

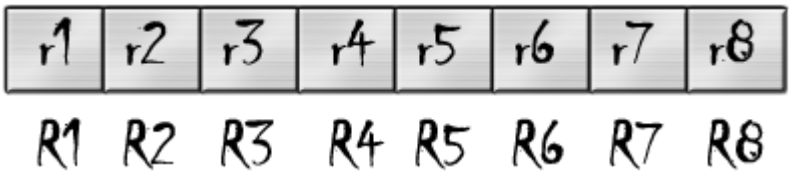
$F_Å$: funzioni calcolabili $F_Å \subset F$
funzioni non calcolabili: $(F \setminus F_Å)$

$$F = F_Å \cup (F \setminus F_Å)$$

F = funzioni calcolabili unite alle funzioni non calcolabili

02 URM

La URM (acronimo di Unlimited Register Machine) è un modello matematico di un computer.
La URM è una macchina ideale costituita da un numero infinito di registri chiamati $R_1, R_2, R_3, ...$ ognuno dei quali contiene un numero naturale.



$r_i \in N$

La macchina URM ha memoria infinita, ma noi useremo sempre una parte finita di essa.
La macchina è inoltre dotata di un agente in grado di eseguire i programmi, ossia di eseguire la sequenza di istruzioni $I_1, I_2, ...$

Istruzioni:

1. $Z(n)$ Funzione zero: imposta il valore del registro R_n a zero.
2. $S(n)$ Funzione successore: aumenta il valore di R_n di un unità.
3. $T(m,n)$ Funzione trasferimento: trasferisce il contenuto di R_m in R_n.
4. $J(m,n,t)$ Funzione salto: vengono confrontati i valori contenuti in R_m ed R_n:
   - Nel caso tale confronto dia esito positivo viene effettuato un salto all'istruzione I_t.
   - Nel caso tale confronto dia esito negativo viene eseguita l'istuzione successiva.

esemio: somma		R_1	R_2	R_3	R_4	
I_1	$J(2,3,end)$	1	2	0	0	...
I_2	$S(1)$	2	2	0	0	...
I_3	$S(3)$	2	2	1	0	...
I_4	$J(1,1,1)$	3	2	2	0	…

Per convenzione l'output del programma viene sempre inserito nel registro R_1
Lo stato iniziale del programma P viene indicato con $P(a_1, a_2, a_3, ...)\downarrow$
Per indicare che il programma P con configurazione $(a_1, a_2, ..., a_n)$ temina in R_1 con output x viene usata la notazione $P(a_1, a_2, ..., a_n)\downarrow x$
Definizione:
Una funzione $f: N \to N$ è URM-calcolabile se esiste un programma URM P tale che per ogni $(a_1, a_2, ..., a_n) \in N$ e con $x \in N$

$$P(a_1, a_2, ..., a_n)\downarrow x \text{ se e solo se}$$
$$(a_1, a_2, ..., a_n) \in dom(f) \text{ e}$$
$$f(a_1, a_2, ..., a_n) = x$$

(in parole povere si tratta del "è calcolabile se esiste un programma che lo calcola" scritto diversamente)
ζ classe delle funzioni URM-calcolabili.

$\zeta^{(k)}$ classe delle funzioni URM-calcolabili k-arie

Una funzione quando è calcolabile lo è da un numero infinito di programmi.

Sia URM' una macchina senza $T(m,n)$ e sia ζ' la classe di funzioni URM'-calcolabili
$\zeta' \subseteq \zeta$ sia $f : N^k \to N$ URM'-calcolabile, esiste un programma P' tale che:

$f = f^{(k)}_{P'}$ ma P' è anche un programma URM $\to f \in \zeta$

$(\zeta \subseteq \zeta')?$ In pratica ci chiediamo se quello che abbiamo tolto non possa essere implementato con quello che ci rimane.

I_1	$T(m,n)$	$Z(n)$
I_2		$J(n,m,end)$
I_3		$S(n)$
I_4		$J(1,1,2)$

in questo caso $\zeta \subseteq \zeta$
Quindi esiste un programma P' in URM' tale che $f^{(k)}_{P'} = f^{(k)}_{P} = f$

dimostrazione per induzione: h = # istruzioni
$(h = 0)$ P è gia in URM' $\to P' = P$
$(h \to h+1)$ Sia P con $h+1$ istruzione $T(m,n)$

P		P'	
	I_1		I_1
	.		.
	.		.
	.		.
	$I_s = T(m,n)$		$I_s = J(1,1,l+2)$
	.		I_{s+1}
	.		.
	I_l (end)		I_l
			$I_{l+1} = J(1,1,l+6)$
			$I_{l+2} = Z(n)$
			$I_{l+3} = J(n,m,s+1)$
			$I_{l+4} = S(n)$
			$I_{l+5} = J(1,1,l+3)$
			I_{l+6} (end)

Attenzione: le dimostrazioni per induzione possono essere ingannevoli, a volte è possibile arrivare ad un risultato errato perché non si ha dimostrato la cosa giusta.

03 Predicati decidibili ed altri domini

Predicati Decidibili
un predicato $Q(x_1, ..., x_k)$ può essere visto come una relazione binaria $Q : N^k \rightarrow \{vero, falso\}$ che indica se tale affermazione sia vera o meno.

def:
Un predicato $Q : N^k$ è decidibile nel caso la funzione che lo compone sia calcolabile.
(Quindi se esiste un programma che lo calcola)

La funzione caratteristica di un predicato è una funzione $\chi_Q : N^k \rightarrow N$.
La funzione $\chi_Q(x^{\rightarrow})$:

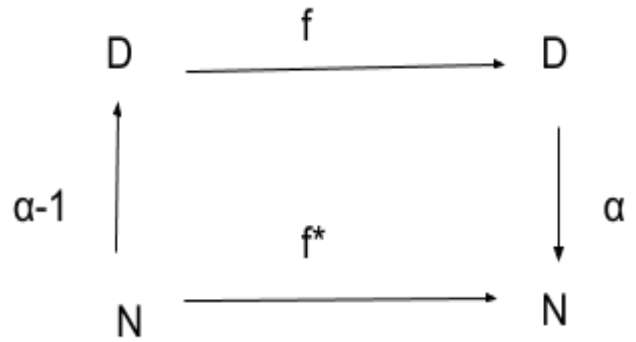
- è uguale ad 1 nel caso la funzione $Q(x^{\rightarrow})$ sia vero.
- è uguale a 0 altrimenti.

(La funzione caratteristica è una funzione totale)

Computabilità su altri domini
Visto che l'URM gestisce solo numeri naturali N, la nostra attuale nozione di computabilità e decidibilità di funzioni e predicati si applica solamente
Tale nozione è facilmente estendibile a qualsiasi altro dominio, visto che possiamo codificare qualsiasi dominio per renderlo equivalente ad N.

D : dominio
α : funzione biunivoca $D \rightarrow N$
(tale funzione è ovviamente invertibile)

Avendo una funzione $f : D \rightarrow D$ essa è calcolabile se la sua funzione codificata $f^* = \alpha(f)$ è calcolabile.



Praticamente se agiamo su domini estranei a quello dei numeri naturali e vogliamo controllare se una tale funzione sia calcolabile, basterà che la sua rispettiva funzione codificata lo sia.

04 composizione e ricorsione

Avendo ζ classe delle funzioni URM-calcolabili.
ζ è chiusa rispetto alle seguenti operazioni:

- composizione (funzionale)
- ricorsione primitiva
- minimizzazione

Composizione di funzioni calcolabili
Avendo delle funzioni di base che sappiamo essere calcolabili:

- $zero : N^k \to N \quad Z(x_1, ..., x_k) = 0 \qquad Z(1)$
- $successore : N \to N \quad succ(x) = x + 1 \qquad S(1)$
- $unione \ \bigcup_i^k : N^k \to N \quad \bigcup_i^k (x_1, ..., x_k) = x_i \qquad T(i, 1)$

possiamo dire che una funzione è calcolabile se posso scriverla come composizione di funzioni che sappiamo essere calcolabili.

Notazione:
Dato un programma P URM indichiamo con $\varrho(P)$ (ro di P) il numero massimo di registri usati dal programma.

$$\varrho(P) = max\{i | R_i \text{ è } usato\ in\ P\}$$

Si dice che P è in forma standard se usa una lunghezza di registri ($l(P)$) limitata.

$$J(m, n, t) \qquad t \le l(P) + 1$$

Dato un programma P si indicherà con $P[i_1, ..., i_k \to l]$ il programma con input nei registri $R_{i_1} ... R_{i_k}$ ed output in R_l (non si assume che i restanti registri siano impostati a zero).

Composizione generalizzata
dato una funzione $f : N^n \to N$ e date le funzioni $g_1, ..., g_n : N^k \to N$ definisco la funzione $h : N^k \to N \qquad h(x^{\rightarrow}) = f(g_1(x^{\rightarrow}), ..., g_n(x^{\rightarrow}))$
$h(x^{\rightarrow})$ è definito su tutte le g_i, ovviamente anche f è definita.

$$h(x^{\rightarrow})\downarrow \text{ se e solo se } g_1(x^{\rightarrow})\downarrow, ..., g_n(x^{\rightarrow})\downarrow \text{ e } f(g_1(x^{\rightarrow}), ..., g_n(x^{\rightarrow}))\downarrow$$

Proposizione: se $f : N^n \to N \ g_1, ..., g_n : N^k \to N \in \zeta$ allora anche $h : N^k \to N$ $h(x^{\rightarrow}) = f(g_1(x^{\rightarrow}), ..., g_n(x^{\rightarrow})) \in \zeta$
(se le funzioni base sono calcolabili allora anche la funzione creata dalla loro composizione lo sarà)

Ricorsione primitiva

dato una funzione $f : N^k \to N$ ed una funzione $g : N^{k+2} \to N$ definiamo $h : N^{k+1} \to N \quad x^{\rightarrow} \in N^k$

$$h(x^{\rightarrow}, 0) = f(x^{\rightarrow})$$
$$h(x^{\rightarrow}, y+1) = g(x^{\rightarrow}, y, h(x^{\rightarrow}, y))$$

$$\Phi : (N^k \to N) \to (N^k \to N)$$
$$\Phi(h)(x, 0) = f(x^{\rightarrow})$$
$$\Phi(h)(x^{\rightarrow}, y+1) = g(x^{\rightarrow}, y, h(x^{\rightarrow}, y))$$

(h è un punto fisso della funzione Φ (fi))

L'obbiettivo è trovare un h tale che $\Phi(h) = h$ ed avente le seguenti caratteristiche:

- h esiste
- h unico

esempi:

- somma

  $h(x, y) = x + y$

  $x + 0 = x$ $\qquad$ $f(x) = x$

  $x + (y+1) = (x+y) + 1$ $\qquad$ $g(x, y, z) = succ(z)$

- prodotto

  $h(x, y) = x * y$

  $x * 0 = 0$ $\qquad$ $f(x) = 0$

  $x * (y+1) = xy + 1$ $\qquad$ $g(x, y, z) = z + x$

Proposizione:

se $f : N^k \to N$ $g : N^{k+2} \to N \in \zeta$ allora h definita per ricorsione primitiva è $\in \zeta$.

esempi:

data una funzione $f : N^k \to N$ totale calcolabile:

- $\sum_{z<y} f(x^{\rightarrow}, z)$

  dimostrazione: (funzione definita per ricorsione primitiva su y)

  $$\sum_{z<0} f(x^{\rightarrow}, z) = 0$$

  $$\sum_{z<y+1} f(x^{\rightarrow}, z) = (\sum_{z<y} f(x^{\rightarrow}, z)) + f(x^{\rightarrow}, y)$$

  Calcolabile Totale.

- $\prod_{z<y} f(\vec{x}, z)$

  dimostrazione: (funzione definita per ricorsione primitiva su y)

  $$\prod_{z<0} f(\vec{x}, z) = 1$$

  $$\prod_{z<y+1} f(\vec{x}, z) = (\prod_{z<y} f(\vec{x}, z)) + f(\vec{x}, y)$$

  Calcolabile Totale.

05 chiusura e minimizzazione

Chiusura

ζ è chiuso rispetto a:

- Composizione
- Ricorsione primitiva

questo ci permette che dalle funzioni fondamentali:

- Zero
- Successore
- Proiezione

di derivare le seguenti funzioni:

- $x+y$
- $x-y$
- $x*y$
- x^y
- $x!$
- min
- max
- $|x-y|$
- qt
- rs
- sg

Osservazione:

date delle funzioni $f_1,...,f_n : N^k \to N$ calcolabili totali ed avendo dei predicati $Q_1,...,Q_n \subseteq N^k$ decidibili, allora $\forall x^{\to} \in N \; \exists! i \; Q_i(x^{\to})$

$\exists!$: esiste un solo

la funzione $f(x^{\to})$:

$f_1(x^{\to})$ se $Q_1(x^{\to})$
$f_2(x^{\to})$ se $Q_2(x^{\to})$
.
.
.
$f_n(x^{\to})$ se $Q_n(x^{\to})$

poichè esso sia vero la funzione $f(x^{\to})$ dev'essere totale

$f(x^{\to}) = f_1(x^{\to})\chi_{Q_1}(x^{\to}) + ... + f_n(x^{\to})\chi_{Q_n}(x^{\to})$

χ_{Q_i} : funzione caratteristica dell' i-esimo predicato

(praticamente vale solo la parte dove il predicato è vero, ossia dove la funzione caratteristica da come risultato 1)

Algebra della decidibilità

avendo due predicati $Q(x^{\rightarrow}), Q'(x^{\rightarrow}) \subseteq N^k$ decidibili, allora ogni loro composizione funzionale è decidibile.

Composizioni funzionali:

- $\neg Q(x^{\rightarrow})$

  dimostrazione:

  $\chi_{\neg Q}(x^{\rightarrow}) = \overline{sg}(\chi_Q(x^{\rightarrow}))$

  ($\overline{sg}$ è la funzione segno negato)

  Sia $\overline{sg}$ che $\chi_Q(x^{\rightarrow})$ sono funzioni calcolabili, quindi $\chi_{\neg Q}(x^{\rightarrow})$ è una composizione di funzioni calcolabili.

- $Q(x^{\rightarrow}) \wedge Q'(x^{\rightarrow})$

  dimostrazione:

  $\chi_{Q \wedge Q'}(x^{\rightarrow}) = \chi_Q(x^{\rightarrow}) * \chi_{Q'}(x^{\rightarrow})$

  Sia $\chi_Q(x^{\rightarrow})$ che $\chi_{Q'}(x^{\rightarrow})$ sono funzioni calcolabili,quindi $\chi_{Q \wedge Q'}(x^{\rightarrow})$ è una composizione di funzioni calcolabili.

- $Q(x^{\rightarrow}) \vee Q'(x^{\rightarrow})$

  dimostrazione:

  $\chi_{Q \vee Q'}(x^{\rightarrow}) = sg(\chi_Q(x^{\rightarrow}) + \chi_{Q'}(x^{\rightarrow}))$

  La funzione segno sg è usata per fare in modo che la funzione caratteristica $\chi_{Q \wedge Q'}(x^{\rightarrow})$ possa dare 0 od 1 come risultato.

  $\chi_Q(x^{\rightarrow}), \chi_{Q'}(x^{\rightarrow})$ e sg sono funzioni calcolabili,quindi $\chi_{Q \vee Q'}(x^{\rightarrow})$ è una composizione di funzioni calcolabili.

Un predicato Q è decidibile se la sua funzione caratteristica χ_Q è calcolabile.

Quantificazione Limitata (Predicati)

(piccola generalizzazione di N)

Avendo un predicato $Q(x^{\rightarrow}, y)$ decidibile, allora:

$Q_1(x^{\rightarrow}, y) = \forall z < y \; Q(x^{\rightarrow}, y)$ decidibile

$Q_2(x^{\rightarrow}, y) = \exists z < y \; Q(x^{\rightarrow}, y)$ decidibile

dimostrazione:

$$\chi_{Q_1}(x^{\rightarrow}, y) = \prod_{z<y} \chi_Q(x^{\rightarrow}, y)$$

$\chi_{Q_2}(x^{\rightarrow}, y) = sg(\sum_{z<y} \chi_Q(x^{\rightarrow}, y))$ (anche qui uso sg per limitare il risultato a 2 possibili valori, 0 ed 1)

Minimizzazione Limitata

Idea: cercare un punto in cui la funzione si annulla (ossia è uguale a 0)

Avendo una funzione $f(x^{\rightarrow}, z) : N^{k+1} \rightarrow N$ calcolabile totale definisco $h : N^{k+1} \rightarrow N$

$h(x^{\rightarrow}, y) = \mu z < y . f(x^{\rightarrow}, z) = 0$

minimo $z < y$ tale che $f(x^{\rightarrow}, z) = 0$ (quindi si annulla)

μ = $\mu - operatore$

Quindi $h(x^{\rightarrow}, y)$ rilascerà come risultato:

- il minimo $z < y$ tale che $f(x^{\rightarrow}, z) = 0$ se esso esiste
- y altrimenti

dimostrazione: (tramite ricorsione su y)

$h(x^{\rightarrow}, 0) = 0$

$h(x^{\rightarrow}, y + 1) =$ 2 casi

a. z se $h(x^{\rightarrow}, y) < y \rightarrow h(x^{\rightarrow}, y)$
   (vuol dire che f si annulla su un valore minore di y)
b. $h(x^{\rightarrow}, y) = y$
   (vuol dire che $\forall z < y \ f(x^{\rightarrow}, z) \neq 0$)
   Si divide in ulteriori 2 sottocasi
   - y se $f(x^{\rightarrow}, y) = 0$
   - $y + 1$ altrimenti

quindi:

$h(x^{\rightarrow}, y + 1) = (caso\ a) + (caso\ b) = sg(y - h(x^{\rightarrow}, y)) * (h(x^{\rightarrow}, y)) + \overline{sg}(y - h(x^{\rightarrow}, y)) * (y + sg(f(x^{\rightarrow}, y)))$

(due modi per scrivere la stessa cosa)

Osservazione: le seguenti funzioni sono calcolabili

- $D(x)$ = funzione che rilascia il $\#$ di divisori di x.
  dimostrazione per composizione di funzioni calcolabili:

  $D(x) = \sum_{y<x} \overline{sg}(rm(y, z))$

  (rm = funzione resto)
  (tale funzione da 1 ogni volta che $rm(y, z)$ =0 e somma tutte queste occorrenze)

- $P_2(x)$ = funzione che rilasca 1 se x è un numero primo, 0 altrimenti
  dimostrazione per composizione di funzioni calcolabile
  $P_2(x) = sg(|D(x) - 2|)$

- P_x = funzione che rilascia l'x-mo numero primo
  dimostrazione per minimizzazione limitata:

  $P_0 = 0$ (banale)

  $P_{x+1} = \mu\, z \leq P_{x!} + 1 . |P_2(z) * \overline{sg}(P_{x+1} - z) - 1|$

  (da 1 quando entrambe sono vere)

- $(x)_y$ = esponente di P_y nella decomposizione di x.
  dimostrazione per minimizzazione limitata:
  $(x)_y = \mu\, z \leq x . \overline{sg}(rm(P_y^{z+1}, x))$

Minimizzazione Illimitata

(come prima, ma non è limitata (banana) e non è più totale)

dato $f : N^{k+1} \to N$ definisco $h : N^k \to N$

$h(x^{\rightarrow}) = \eta z.f(x^{\rightarrow}, z)$

il minimo z tale che $f(x^{\rightarrow}, z) = 0$

questo porta a 2 problemi:

- se $f(x^{\rightarrow}, z)$ sempre $\neq 0$ allora $h(x^{\rightarrow}) = \uparrow$ (indefinito)
- se $f(x^{\rightarrow}, z) \uparrow$ (indefinito) prima di trovare un $= 0$ allora $h(x^{\rightarrow}) = \uparrow$ (indefinito)

$h(x^{\rightarrow}) = \eta z.f(x^{\rightarrow}, z) =$

- minimo z tale che $f(x^{\rightarrow}, z) = 0$ e per ogni $z' < z$ $f(x^{\rightarrow}, z) \downarrow \neq 0$ (se esiste)
- $\uparrow$ altrimenti

$z_f = \{z \mid f(x^{\rightarrow}, z) \ \wedge \ \forall z' < z \, f(x^{\rightarrow}, z) \downarrow \neq 0\} =$

- $min \, z_{f,x^{\rightarrow}}$ se $z_{f,x^{\rightarrow}} \neq 0$
- $\uparrow$ altrimenti

Proposizione: se $f : N^{k+1} \to N \in \zeta$ allora anche $h : N^k \to N \in \zeta$

(È come dire che se la funzione f è URM-calcolabile allora lo è anche la sua minimalizzazione)

Dimostrazione: sia $f : N^{k+1} \to N \in \zeta$ e sia P il programma in forma normale per f

$\mu y.f(x^{\rightarrow}, y) \in ? \ \zeta$

con $m = max\{\varrho(P), k\}$

$$
\begin{array}{ll}
 & T([1,k],[m+1,...,m+k]) \\
loop & P[m+1,...,m+k \to 1] \\
 & J(1,m+k+2,END) \\
 & S(m+k+1) \\
 & J(1,1,loop) \\
END & T(m+k+1,1)
\end{array}
$$

Questo è il programma per $\mu y.f(x^{\rightarrow}, y)$, quindi $\mu y.f(x^{\rightarrow}, y) \in \zeta$

Esempi:

_ dato una funzione $f : N \to N$, $f(x) =$

- $\sqrt{x}$ se x è quadrato
- $\uparrow$ altrimenti

  $$f(x) = \mu y.|y^2 - x|$$

  essendo f definita per minimizzazione, $f \in \zeta$

_data una funzione $f : N^2 \to N$, $f(x,y) =$

- x/y se $y \neq 0$ ed x divisibile per y
- $\uparrow$ altrimenti

  $$f(x,y) = \mu k.(|x - yk| + \overline{sg}(y))$$ (il $\overline{sg}(y)$ serve se $y = 0$, da come risultato 1 e fa in modo che la funzione diverga in quel caso)

  essendo f definita per minimizzazione, $f \in \zeta$

Definizione:

Una funzione finita $\vartheta : N^k \to N$ è una funzione con una relazione finita, ossia ha un finito numero di coppie, quindi è definita su un numero finito di punti ed indefinita su tutti gli altri.

(ϑ = peta)

$\vartheta(x^{\rightarrow}) =$

- y_1 se $x^{\rightarrow} = x_1^{\rightarrow}$
- .
- .
- .
- y_n se $x^{\rightarrow} = x_n^{\rightarrow}$
- $\uparrow$ altrimenti

Proposizione: tutte le funzioni finite sono calcolabili.

Dimostrazione: per semplicità assumiamo $\vartheta : N \to N$ (unario)

$\vartheta(x) =$

- y_1 se $x = x_1$
- .
- .
- .
- y_n se $x = x_n$
- $\uparrow$ altrimenti

$$\vartheta(x) = y_1\overline{sg}|x - x_1| + ... + y_n\overline{sg}|x - x_n| + \prod_{i=1}^{n} |x - x_i|$$

essendo ϑ definita per composizione e minimizzazione, $\vartheta \in \zeta$

06 Altri modelli di calcolo

Tesi di Church: (modello che abbiamo usato fin'ora)
Ogni funzione è calcolabile tramite procedimento effettivo se e soltanto se è URM-calcolabile.

Modello Alternativo:
Funzioni parziali ricorsive (Gödel e Kleene), Indicherà un insieme di funzioni calcolabili indicabili con $\Re$

Definizione:
la classe $\Re$ delle funzioni parziali ricorsive è la minima classe di funzioni che contiene le funzioni di base:

(a) Zero $z(x^{\rightarrow}) = 0$
(b) Successore $s(x) = x + 1$
(c) Proiezioni $U_i^k(x_1 ... x_k) = x_i$

Inoltre la classe di funzioni $\Re$ è chiusa rispetto a:

(1) Composizione Generalizzata
(2) Ricorsione Primitiva
(3) Minimizzazione Illimitata

Definizione:
Una classe di funzioni X è definita come "ricca" se contiene le funzioni base (a),(b) e (c) ed è chiusa rispetto a (1),(2) e (3).
$\Re \subseteq X$ per ogni X ricca e di conseguenza anche $\Re$ è una classe ricca.
Se le classi X_1 ed X_2 sono classi ricche, allora anche $X_1 \cap X_2$ è una classe ricca.

Definizione:

$\Re = \bigcap_{X\, ricca} X_i$ il che vuol dire che $\Re$ è uguale alle funzioni base (a),(b) e (c) assieme a tutte le funzioni che ottengo da questo usando le chiusure (1),(2) e (3).

Definizione:
La classe delle funzioni primitive ricorsive $P\Re$ è la minima classe di funzioni che contengono le funzioni base (a),(b) e (c) ed è chiusa rispetto ad (1) e (2).

Teorema: $\Re = \zeta$
Dimostrazione:
(Basta dimostrare che vale $\Re \subseteq \zeta$ e $\zeta \subseteq \Re$ contemporaneamente)

- $\Re \subseteq \zeta$

  Lo abbiamo già dimostrato, poiché $\Re = \bigcap_{X\, ricca} X_i$ e ζ è una classe ricca.

- $\zeta \subseteq \Re$

  Meno banale della dimostrazione precedente, bisogna lavorarci un pochino…

Per la funzione $f \in \zeta$ URM-calcolabile esiste un programma P URM in forma normale

		r_1	...	r_k	r_{k+1}	...
P	I_1	x_1	...	x_k	0	...
	.					
	.					
	.					
	I_s					

Avremo le funzioni totali $C_P^1, J_P : N^{k+1} \to N$

$C_P^1(x^{\rightarrow}, t) =$

- Contenuto del registro r_1 dopo t passi, se $P(x^{\rightarrow})\downarrow \leq t$ passi
- Contenuto finale del registro r_1 altrimenti

$J_P(x^{\rightarrow}, t) =$

- la prossima istruzione da eseguire dopo t passi di $P(x^{\rightarrow})$ se non termina prima
- 0 altrimenti

Se $f(x^{\rightarrow})\downarrow$ (definito)

Il programma P termina su $x^{\rightarrow}$ in $t_0 = \mu t.\, J_P(x^{\rightarrow}, t)$

$f(x^{\rightarrow}) = C_P^1(x^{\rightarrow}, t_0) = C_P^1(\mu t.\, J_P(x^{\rightarrow}, t))$

Se $f(x^{\rightarrow})\uparrow$ (non definito)

$\mu t.\, J_P(x^{\rightarrow}, t)\uparrow$

$f(x^{\rightarrow}) = C_P^1(x^{\rightarrow}, t_0) = C_P^1(\mu t.\, J_P(x^{\rightarrow}, t))$

Se si dimostra che $C_P^1, J_P \in \Re \Rightarrow f \in \Re$

Supposizione: $C_P^1, J_P \in \Re$

Possiamo codificare la memoria come un numero su cui operare.

Configurazione memoria $< r_1, r_2, ... > \ \to \ \prod_{i \geq 1} p^{r_i} = x$

Essendo x la codifica della memoria, allora $r_i = (x)_i$

Dimostrazione che $C_P : N^{k+1} \to N \ \in \Re$

($C_P(x^{\rightarrow}, t) =$ contenuto della memoria dopo t passi del programma $P(x^{\rightarrow})$)

e che $J_P : N^{k+1} \to N \ \in \Re$

($J_P(x^{\rightarrow}, t) =$ istruzione da eseguire dopo t passi di $P(x^{\rightarrow})$ se non termina, 0 altrimenti)

$\xi_P(x^{\rightarrow}, t) = \Pi(C_P(x^{\rightarrow}, t), J_P(x^{\rightarrow}, t))$

Dimostrazione per ricorsione primitiva:

su $t = 0$

$C_P(x^{\rightarrow}, 0) = \prod_{i=1}^{k} p_1^{x_i}$

$J_P(x^{\rightarrow}, 0) = 1$

Utilizzando $C = C_P(x^{\rightarrow}, t)$ e $J = J_P(x^{\rightarrow}, t)$

su $t+1$ I_s = ultima istruzione

$C_P(x^{\rightarrow}, t+1) =$

- $q^t(P_n^{(c)_n}, c)$ se $1 \le j \le s$ e $I_j = Z(n)$ (l'elevazione a $(c)_n$ di P serve ad azzerare)
- $p_n c$ se $1 \le j \le s$ e $I_j = S(n)$
- $P_n^{(c)_n} q^t(P_n^{(c)_n}, c)$ se $1 \le j \le s$ e $I_j = T(m, n)$
- c altrimenti

$J_P(x^{\rightarrow}, t+1) =$

- $j+1$ se $1 \le j < s$ e ($I_j = Z(n), S(n)$ o $T(m,n)$) oppure($I_j = J(m,n,q)$ e $(C)_m \neq (C)_n$)
- q se $1 \le j < s$ e $I_j = J(m,n,q)$, $(C)_m = (C)_n$ e $1 \le q \le s$ (controllo che non si salti fuori dal programma)
- 0 altrimenti

Essendo C_p e J_P definita per ricorsione primitiva e composizione, allora $C_p, J_p \in P\Re \subseteq \Re$

$f(x) = (C_P(x^{\rightarrow}, \mu t.J_P(x^{\rightarrow}, t)))_1 \in \Re$ $()_1$ = prendo il contenuto del registro r_1

Osservazione:

$for \approx ricorsione\ primitiva$ $while \approx minimizzazione$

La ricorsione primitiva (come il for) terminerà prima o poi, quindi ha un numero conosciuto di iterazioni, la minimizzazione invece terminerà solamente quando la sua condizione d'uscita è verificata, venendo ripetuta un numero sconosciuto di volte (come il while),quindi potrebbe non terminare mai.

Da notare che solo con la minimizzazione possiamo definire funzioni non totali.

$URM_{for,while}$

$\zeta_{for,while} = \zeta = \Re$

$\zeta_{for} = P\Re \subset \Re = \zeta$

Non avendo il while (minimizzazione), la classe $P\Re$ non può definire funzioni non totali.

La classe $P\Re$ può definire tutte le funzioni totali? NO!

Un esempio di funzione totale non definibile da $P\Re$ è la funzione di Ackermann $\psi : N^2 \rightarrow N$

$\psi(x,y) =$ (ψ =psi)

- $\psi(0, y) = y + 1$
- $\psi(x+1, 0) = \psi(x, 1)$
- $\psi(x+1, y+1) = \psi(x, \psi(x, \psi(x+1, y)))$

Tale funzione ha una crescita mostruosa, gia $\psi(5,5)$ risulta praticamente incalcolabile, dando come risultato $2^{10^{32000}}$.
Ordine $(N^2, \leq_{lex})$ $(x,y) \leq_{lex} (x',y')$ se $x \leq_{lex} x'$ oppure se $x = x'$ e $y \leq_{lex} y'$

- $\psi(x+1,0) = \psi(x,1)$
  $(x,1) \leq_{lex} (x+1,0)$
- $\psi(x+1,y+1) = \psi(x,\psi(x,\psi(x+1,y)))$
  $(x+1,y) \leq_{lex} (x+1,y+1)$

Il problema è che quest'ultima è una catena di configurazione infinita.
(Questo poiché il nostro ordine attuale non è ben fondato)

Ordine ben fondato $(D, \leq)$
Un ordine è ben fondato se non ha catene discendenti infinite.
$d_0 > d_1 > d_2 > d_3 > \ldots$ $conf_i \leq conf_j$
In un ordine ben fondato ogni sottoinsieme non vuoto $(N^2, \leq_{lex})$ ha un minimo dato $x \subseteq N^2$
$x_o = min\{x | (x,y) \in X\}$
$y_o = min\{y | (x,y) \in X\}$
$(x_o, y_o) = min\, X$
Ackermann sembra avere una ricorsione sensata:
$P(P(0) \wedge P(n) \rightarrow P(n+1)) \Rightarrow \forall n\, P(n)$
$\forall n(\forall n' < n\, P(n') \rightarrow P(n)) \Rightarrow \forall n\, P(n)$
Traduzione degli orrori sopra: se una proprietà vale per tutti i numeri prima, allora essa varrà per tutti quelli dopo.

Induzione ben fondata: se $(D, \leq)$ ben fondato, P proprietà:
se $\forall d \in D\ (\forall d' < d\ \ P(d')) \Rightarrow P(d) \Rightarrow \forall d \in D$

Proprietà di Ackermann:

1) ψ è totale

   $\forall (x,y) \in N^2 \ \psi(x,y)\downarrow$

   Dimostrazione:

   sia $(x,y) \in N^2$ tale che $\forall (x',y') <_{lex} (x,y) \ \ \psi(x',y')\downarrow$

   vari casi:

   - $x = 0$ $\quad \psi(x,y) = \psi(0,y) = y+1 = \ \downarrow$
   - $y = 0, x > 0$ $\quad \psi(x,y) = \psi(x-1,1) <_{lex} (x,0) = \ \downarrow$
   - $y > 0, x > 0$ $\quad \psi(x,y) = \psi(x-1, \psi(x,y-1)) = \psi(x-1,u) \ = \ \downarrow$

     $u = \psi(x,y-1)$

   La funzione è sempre definita, quindi è totale.

2) ψ è calcolabile

   Intuizione: per calcolare $\psi(x,y)$ uso $\psi(x',y')$ per finiti (x',y')

   Insieme valido $S \subseteq N^3$

   $(x,y,z) \in S \Rightarrow z = \psi(x,y)$

   $(x,y,\psi(x,y)) \in S \Rightarrow S$ contiene anche tutto $(x',y',\psi(x',y'))$ necessario per $\psi(x,y)$

   Definizione: $S \subseteq N^3$ valido se

   $(0,y,z) \in S \Rightarrow z = y+1$

   $(x+1,0,z) \in S \Rightarrow (x,1,z) \in S$

   $(x+1,y+1,z) \in S \Rightarrow$ esiste $u \in N$ tale che $(x+1,y,u)$

   Proprietà: $\psi(x,y) = z$ se e solo se esiste $S \subseteq N^3$ valida tale che $(x,y,z) \in S$

   $\psi(x,y) = \mu(S,z).\ S\ valido\ (x,y,z) \in S$

3) $\psi \notin P\Re$

   Considerando per assurdo $\psi \in P\Re$

   Sia $n = \#$ numero di annidamenti di ricorsione primitiva nel programma per ψ

   $\psi(n+2,n+2) < \psi_{n+2}(max\{n+2,n+2\}) = \psi_{n+2}(n+2) = \psi(n+2,n+2)$

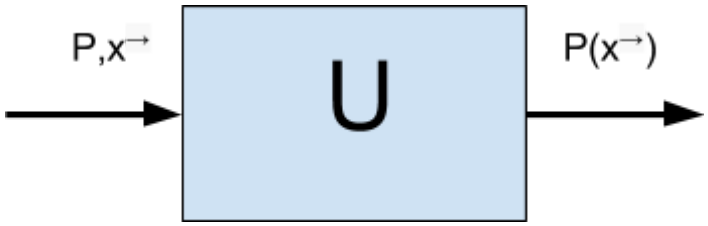
   Assurdo, la funzione di Ackermann dovrebbe essere inferiore a se stessa.

   Quindi $\psi \notin P\Re$

Quindi per poter definire tutte le funzioni totali serve anche la minimizzazione.

07 Codifica dei programmi

Programma Universale (alias: compilatore)



L'idea è di un programma che possa emulare qualsiasi altro programma.
Serve un enumerazione delle funzioni calcolabili.

Ogni programma, che altro non è che un insieme d'istruzioni, può essere codificato come un numero.
X numerabile se esiste $f : N \to X$ biunivoca.

Osservazione: ci sono enumerazioni biunivoche effettive:

1) $\Pi : N^2 \to N$

$$\Pi(x, y) = 2^x(2y + 1) - 1$$
$$\Pi^{-1}(n) = (\Pi, (n), \Pi_2(n))$$

2) $\upsilon : N^3 \to N$

$$\upsilon(x_1, x_2, x_3) = \Pi(\Pi(x_1, x_2), x_3)$$
$$\upsilon^{-1}(n) = (\upsilon_1(n), \upsilon_2(n), \upsilon_3(n))$$
$$\upsilon_1(n) = \Pi_1(\Pi_1(n))$$

3) $\tau : \bigcup_{k+1} N^k \to N$

$$(x_1, ..., x_k) \to \prod_{i=1}^{k} P_i^{x_k} - 1$$

esempi:

$$(2, 1) = 2^2 * 3^1 - 1$$
$$(2, 1, 0) = 2^2 * 3^1 * 5^0 - 1$$

$$\tau(x_1, ..., x_k) = \prod_{i=1}^{k} P_i^{x_i} * P_k^{x_{k+1}} - 2$$

n decodifica $l(n) = max\ k, P_k | (n + 2)$

$a(n, 1) =$

- $(n + 2)_i \qquad i < l(n)$
- .
- .
- .
- $(n + 2)_i \qquad i = l(n)$

$$\tau^{-1}(n) = (a(n, 1), a(n, 2), ..., a(n, l(n)))$$

Proposizione: esistono codifiche effettive biunivoche

$\beta : Istr_{URM} \to N$ (ad ogni istruzione assegna un numero)

$\gamma : P \to N$ (ad ogni programma assegna un numero)

Essendo i programmi una sequenza d'istruzioni, γ altro non è che il prodotto dell'elevamento a potenza della codifica in base alle istruzioni presenti.

Essendo i programmi calcolabili enumerabili ed essendo le funzioni calcolabili quelle che hanno un programma che la calcolano, allora anche tali funzioni sono enumerabili.

Definizione: fissato $\gamma : P \to N$

$\varphi_n^{(k)} = f_{P_n}^{(k)}$

$\varphi_n^{(k)}$ p la funzione di k argomenti calcolata dal programma $P_n = \gamma^{-1}(n)$

$W_n^{(k)} = dom(\varphi_n^{(k)}) \subseteq N^k = \{x^{\rightarrow} | x^{\rightarrow} \in N^k \ tale \ che \ \varphi_n^{(k)}(x^{\rightarrow}) \downarrow\}$

$E_n^{(k)} = cod(\varphi_n^{(k)}) = \{y | \exists x^{\rightarrow} \ \varphi_n^{(k)}(x^{\rightarrow}) = y\}$

Essendo $f : N \to X$ surgettiva per una funzione o non ci sono programmi che la calcolano o ce ne sono infiniti.

08 Diagonale di Cantor e SMN

Diagonale di Cantor

Avendo degli oggetti $x_0, x_1, x_2, \ldots$ la tecnica della diagonale di Cantor ci permette di creare un oggetto diverso da x_i nella componente i, ossia un oggetto diverso da tutti gli altri.
Tale tecnica fu creata da Cantor per dimostrare che ci sono diversi livelli d'infinito.

$2^N = \{x | x \subseteq N\}$

$A^B \qquad A \to B$

$2^N \qquad N \to \{0,1\}$

Proposizione: $|2^N| > |N|$ 2^N non enumerabile
Dimostrazione: Per assurdo supponiamo che 2^N sia numerabile
Avendo 2^N oggetti $x_0, x_1, x_2, \ldots$

$N/2^N$	x_0	x_1	x_2	...	$X = \{i \mid i \notin x_i\}$	prendo i numeri che non appartengono
0	SI	SI	NO	...		all'insieme di quella posizione (quelli con
1	NO	NO	SI	...		la diagonale uguale a NO)
2	SI	SI	NO	...	$X = \{i \mid i \notin x_i\} \subseteq N$	
.	.	.	.	...	$X = \{i \mid i \notin x_i\} \in 2^N$	
.	.	.	.	...	$\exists i_o \in N \qquad X = X_{i_o}$	

$i_o \in X_{i_o} \Rightarrow i_o \notin X = X_{i_o}$ Assurdo
$i_o \notin X_{i_o} \Rightarrow i_o \in X = X_{i_o}$ Assurdo
In entrambi i casi si arriva ad una contraddizione, quindi $|2^N| > |N|$

Altro esempio:
L'insieme delle funzioni $F = \{f \mid f : N \to N\}$ non è enumerabile $|F| > |N|$
Dimostrazione:

$F \supseteq \{f \mid f : N \to \{0,1\}\}$

$|F| \geq |\{f \mid f : N \to \{0,1\}\}| = |2^N| > |N|$ (l'ultima parte l'abbiamo già dimostrata)

Altro metodo di dimostrazione: $|F| > |N|$

Considerando un enumerazione degli elementi di F, dimostriamo che c'è un $f \in F$ che non compare.
Avendo gli oggetti $f_0, f_1, f_2, ...$

$N/$	f_0	f_1	f_2	...
0	$f_0(0)$	$f_1(0)$	$f_2(0)$	...
1	$f_0(1)$	$f_1(1)$	$f_2(1)$	...
2	$f_0(2)$	$f_1(2)$	$f_2(2)$	...
.	.	.	.	...
.	.	.	.	...

Definisco $f(n)$:
- $\uparrow$ se $f_n(n)\downarrow$
- 0 se $f_n(n)\uparrow$

Per definizione tale funzione differisce da qualsiasi funzione sulla diagonale.
$\forall n\, f_n \neq f$ poiché $f_n(n)\downarrow$ se e solo se $f(n)\uparrow$

Di conseguenza l'enumerazione non contiene tutte le funzioni di F, quindi F non enumerabile.
(morale della favola: se hai infinite funzioni, non vuol dire che le hai tutte)

Osservazione: esiste una funzione f totale non calcolabile.
$f(n) =$
- $\varphi_n(n) + 1$ se $n \in W_n$
- 0 se $n \notin W_n$

$N/\zeta^{(k)}$	φ_0	φ_1	φ_2	...
0	$\varphi_0(0)$	$\varphi_1(0)$	$\varphi_2(0)$	...
1	$\varphi_0(1)$	$\varphi_1(1)$	$\varphi_2(1)$	...
2	$\varphi_0(2)$	$\varphi_1(2)$	$\varphi_2(2)$	...
.	.	.	.	...
.	.	.	.	...

(abbiamo un enumerazione delle funzioni calcolabili)
f è totale poiché essa è sempre definita.
Col fatto che la funzione $f(n) = \varphi_n(n) + 1$ se $n \in W_n$, essa sarà per definizione diversa da tutte le funzioni calcolabili, per cui essa sarà non calcolabile.

Teorema SMN (o teorema del parametro)
Prendendo una funzione $f : N^2 \to N$ calcolabile (non necessariamente totale) con 2 parametri.
Per ogni valore fissato a di x, $f(x,y)$ darà luogo ad una funzione unaria computabile g_a tale che:

$$g_a(y) \simeq f(a,y)$$

Finché la funzione g_a risulta computabile essa avrà un indice e tale che:

$$f(a,y) \simeq \varphi_e(y)$$

Il teorema SMN mostra che tale indice e può essere ottenuto direttamente dal valore fissato a.

Teorema SMN (forma semplice)
Supponendo che la funzione $f(x,y)$ sia calcolabile, ci sarà una funzione $k(x)$ totale calcolabile tale che:

$$f(x,y) \simeq \varphi_{k(x)}(y)$$

Dimostrazione:

Per ogni valore fissato a, $k(a)$ sarà la codifica del programma Q_a che dalla configurazione iniziale:

R_1	R_2			
y	0	0	0	...

calcolerà $f(a,y)$

Essendo F il programma che calcolerà f, per Q_a noi scriveremo F preceduto dalle istruzioni che trasformano la configurazione precedente in

R_1	R_2			
a	y	0	0	...

Quindi, definendo Q_a dal seguente programma:

$T(1,2)$
$Z(1)$
$S(1)$
.
. (a volte)
.
$S(1)$
F

Ora definiamo

$k(a) =$ la codifica del programma Q_a

Finché F è fissato e per il fatto che la nostra numerazione γ dei programmi è effettiva, , possiamo vedere che k è una funzione calcolabile.

Quindi, per la tesi di Church, k è calcolabile.

$\varphi_{k(a)}(y) \simeq f(a,y) \quad \forall a$

Teorema SMN:

Per ogni $m, n \geq 1$ esiste una funzione $(m+1)-aria$ calcolabile $s_n^m(e,x)$ tale che:

$$\varphi_e^{(m+n)}(x,y) \simeq \varphi_{s_n^m(e,x)}^{(n)}(y)$$

Dimostrazione: (Generalizziamo la dimostrazione dell'SMN in forma semplice)

Per ogni $i \geq 1$, $Q(i,x)$ sarà una subroutine

$Z(i)$
$S(i)$
.
. (x volte)
.
$S(i)$

Tale subroutine sostituirà il contenuto corrente di R_i di x.

Poi per i fissati m, n definiamo $s_n^m(e,x)$ per essere la codifica del prossimo programma:

$T(n, m+n)$
.
.
.
$T(2, m+2)$
$T(1, m+1)$
$Q(1, x_1)$
$Q(2, x_2)$
.
.
.
$Q(m, x_m)$
P_e

La parte del programma prima di P_e trasforma ogni configurazione:

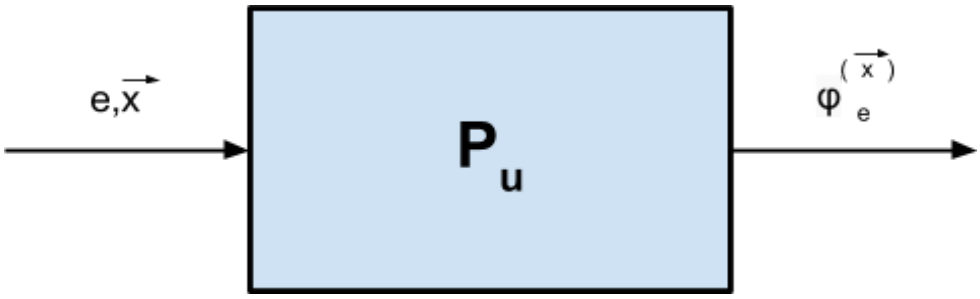
R_1		R_n				
y_1	...	y_n	0	0	0	...

in

R_1		R_m	R_{m+1}		R_{m+n}	
x_1	...	x_m	y_1	...	y_n	...

Per tale definizione e per γ ed γ^{-1} possiamo dire che s_n^m è calcolabile.

09_Programma_universale



Il programma P_u che calcola la funzione e su input $x^{\rightarrow}$, quindi si tratta di un programma che contiene tutti gli altri programmi.

Programma Universale: ψ_U

$\psi_U : N^2 \rightarrow N$ $\quad x =$ indice del programma.

$\psi_U(x,y) = \varphi_x(y)$ $\quad y =$ input.

Definizione:

$\psi_U^{(k)} : N^{k+1} \rightarrow N$

$\psi_U(e, x^{\rightarrow}) = \varphi_e^{(k)}(x^{\rightarrow})$

Teorema: $\psi_U^{(k)} : N^{k+1} \rightarrow N$ è calcolabile.

Dimostrazione:

- $\psi_U(e, x^{\rightarrow}) = \varphi_e(x^{\rightarrow})$
- $\gamma^{-1}(e) = P_e$
- eseguendo $P_e(x^{\rightarrow})$:
  - output di $P_e(x^{\rightarrow})$ se $\psi_U(e, x^{\rightarrow})\downarrow$
  - non termina se $\psi_U(e, x^{\rightarrow})\uparrow$

Memoria:

r_1	r_2	r_3	...	0	...

$C = \prod_{i=1}^{l} P_i^{r_i}$ $\quad r_i = (C)_i$

$\sigma = \prod(C, J)$

$C_k, J_k : N^{k+2} \rightarrow N$

$C_k(e, x^{\rightarrow}, t) =$ contenuto dei registri dopo t passi di $P_e(x^{\rightarrow})$.

$J_k(e, x^{\rightarrow}, t) =$

- istruzioni da eseguire dopo t passi di $P_e(x^{\rightarrow})$ se non termina.
- 0 altrimenti.

$C_k, J_k \in P\Re \subseteq \Re = \zeta$

$\psi_U^k(e, x^{\rightarrow}) = \varphi_e^{(k)}(x^{\rightarrow}) = (C_k(e, x^{\rightarrow}, \mu t.\, J_k(e, x^{\rightarrow}, t)))_1 \in \Re = \zeta$

Effetto dell'esecuzione delle istruzioni:

$$zero(C,n) = qt(p_n^{(C)_n}, C)$$
$$succ(C,n) = p_n.C$$
$$T(C,m,n) = p_n^{(C)_m}.zero(C,n)$$

$nextconf(e,c,t) =$ Configurazione dei registri al prossimo passo.
| Partendo da C esegue la $t-ma$ istruzione di P_e
$= change(C, a(e,t))$

$is(C,i,t) =$ Istruzione successiva se eseguo l'istruzione $i = \beta$ ed essa è la
| $t-ma$ istruzione del programma.
=

- $t+1$ se $rm(4,i) \neq 3$ (4 poichè è il numero d'istruzioni possibili), oppure $(C)_{J_{arg_1}(i)} \neq (C)_{J_{arg_2}(i)}$
- $J_{arg_1}(i)$ altrimenti

$nextistr(e,c,t) =$ prossima istruzione partendo da C ed eseguendo la $t-ma$
| istruzione di P_e.
=

- $is(C, a(e,t), t)$ se $is(C, a(e,t), t) \neq l(e)+1$
- 0 altrimenti

Quindi:

$C_k : N^{k+2} \to N$ $\quad$ $J_k : N^{k+2} \to N$

Essendo sia C_k che J_k basate su funzioni $\in P\Re$

$$C_k, J_k \in P\Re \subseteq \Re = \zeta$$

$$C_k(e, x^{\rightarrow}, 0) = \prod_{i=1}^{k} P_i^{x_i}$$
$$J_k(e, x^{\rightarrow}, 0) = 1$$

$$C_k(e, x^{\rightarrow}, t+1) = nextconf(e, C_k(e, x^{\rightarrow}, t), J_k(e, x^{\rightarrow}, t))$$
$$C_k(e, x^{\rightarrow}, t+1) = nextistr(e, C_k(e, x^{\rightarrow}, t), J_k(e, x^{\rightarrow}, t))$$

$$\psi_U^{(k)}(e, x^{\rightarrow}) = (C_k(e, \mu t.\, J_k(e, x^{\rightarrow}, t)))_1 \in \Re$$

Corollario: sono decidibili i seguenti predicati

1. $H_k(e, x^{\rightarrow}, t) = P_e(x^{\rightarrow}) \downarrow$ in t o meno passi.
2. $S_k(e, x^{\rightarrow}, y, t) = P_e(x^{\rightarrow}) \downarrow y$ in t o meno passi.

(traduzione: sapere se un programma finisce in un numero finito di passi è decidibile)

1. $\chi_{H_k}(e, x^{\rightarrow}, t) = \overline{sg}(J_k(e, x^{\rightarrow}, t))$
2. $\chi_{S_k}(e, x^{\rightarrow}, y, t) = \overline{sg}(J_k(e, x^{\rightarrow}, t) + |y - C(e, x^{\rightarrow}, t)_1|)$

   (la 2. da 1 se entrambi le parti danno 0)

Ricapitolando:
$\psi_U^{(K)}(e, x^{\rightarrow}) = \varphi_e^{(K)}(x^{\rightarrow})$ calcolabile
$H^{(K)}(e, x^{\rightarrow}, t)$ e $S^{(K)}(e, x^{\rightarrow}, y, t)$ decidibili

_ inversa
$f : N \rightarrow N$ iniettiva calcolabile
$f^{-1}(y) =$

- x tale che $f(x) = y$ se $y \in cod(f)$
- $\uparrow$ altrimenti

$f = \varphi_e$ per qualche $e \in N$
$f^{-1}(y) = \mu(x, t)\ S(e, x, y, t)$

$f^{-1}(y) = \mu w.\ S(e, (w)_1, y, (w)_2)$
$\downarrow$
$(w)_1, (w)_2$ (interpreto il numero w come coppia)

$f^{-1}(y) = (\mu w.\ |\chi_{S(e,(w)_1,y,(w)_2)-1}|)_1$ che è calcolabile poiché basata sulla minimizzazione e sulla proiezione.

Corollario: il predicato $Tot(x) =$ "$\varphi(x)$ *totale*" non è decidibile
Supponendo che esso sia decidibile:

$f(x) =$

- $\varphi_x(x) + 1$ totale
- 1 altrimenti

(tale funzione è ovviamente totale essendo sempre definita)

f totale, $f \neq \varphi_x\ \forall x$ tale che φ_x totale, poiché se φ_x totale, allora
$f(x) = \varphi_x(x) + 1 \neq \varphi(x)$
f calcolabile

$$(a) \qquad\qquad (b)$$
$$f(x) = (\varphi_x(x) + 1) * \chi_{Tot(x)} + 1 * (1 - \chi_{Tot(x)}) = (\psi_U(x, x) + 1) * \chi_{Tot(x)} + 1 * (1 - \chi_{Tot(x)})$$

Dove sta il problema? se qualcosa nella funzione è indefinito, tutto lo è, il che è una contraddizione poiché $f(x)$ è definita come una funzione totale.
(Potrei scegliere $\varphi_e = 0\ (0(x) \uparrow\ \forall x)$, funzione sempre indefinita)
Inoltre, sebbene ad una prima occhiata le due versioni possano sembrare identiche, b è diverso da a, poiché $\varphi_x(x)$ può essere indefinita, mentre $\psi_U(x, x)$ no.
Scegliendo $\varphi_e = 0\ (0(x) \uparrow\ \forall x)$

$f(e) = 1$

$(\varphi_e(e) + 1) * \chi_{Tot(e)} + 1 * (1 - \chi_{Tot(e)})\uparrow$ poiché $\varphi_e(e)\uparrow$

$f(x) = \mu w.(S(x, x, (w)_1, (w)_2) \ \wedge \ Tot(x) \vee ((w)_1 = 0 \wedge \neg Tot(x)))_1 + 1$

traduzione: cerco un w tale che:

- $\varphi_x(x)$ quando x è totale.
- 0 altrimenti.

In ogni caso dopo aggiungo 1, in modo che dia $\varphi_x(x) + 1$ o 1.

Essa è decidibile poiché vengono usati solo predicati decidibili.

Con S, se la funzione è totale troverò il numero di passi con cui essa finisce.

Operazioni effettive sui programmi

$p_x \qquad p_y$

$\varphi_x \qquad \varphi_y$

$(\varphi_x + \varphi_y)(z) = \varphi_x(z) + \varphi_y(z) = \varphi_e(z)$

|

$\varphi^{(z)}_{K(x,y)}$ K funzione totale calcolabile

Esiste una funzione $K : N^2 \to N$ totale calcolabile tale che $\varphi^{(z)}_{K(x,y)} = \varphi_x(z) + \varphi_y(z)$

Dimostrazione:

$g(x, y, z) = \varphi_x(z) + \varphi_y(z)$

|

$= \psi_U(x, z) + \psi_U(y, z)$ calcolabile

Per SMN esiste una funzione $K : N^2 \to N$ totale calcolabile tale che $g(x, y, z) = \varphi^{(z)}_{K(x,y)}$

$\varphi^{(z)}_{K(x,y)} = g(x, y, z) = \varphi_x(z) + \varphi_y(z)$

Se abbiamo 2 programmi P_x e P_y che rispettivamente calcolano le funzioni f_x e f_y allora è possibile calcolare la somma dei due programmi.

_ Esiste una funzione $K : N \to N$ totale calcolabile tale che:

$\varphi_{K(x)}(y) = (\varphi_x)^{-1}(y)$

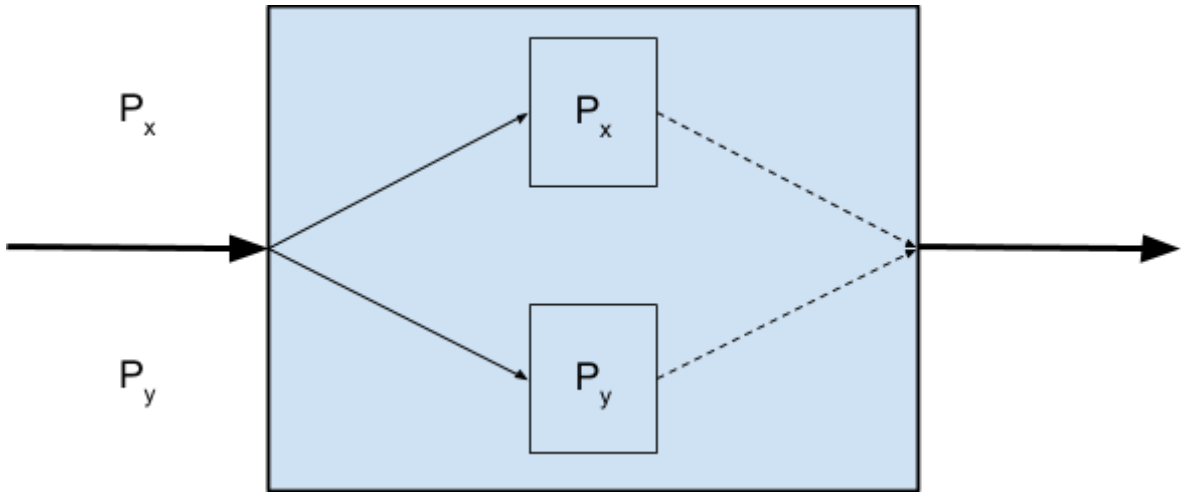
$g(x,y) = (\varphi_x)^{-1}(y)$

|

$= (\mu w.S(x,(w)_1,y,(w)_2)_1$

Calcolabile, quindi per SMN esiste una funzione $K : N \to N$ calcolabile totale tale che $\varphi_{K(x)}(y) = g(x,y)$

_Esiste una funzione $S : N^2 \to N$ tale che $W_{S(x,y)} = W_x \cup W_y$



$1(...) \to$ quando termina vale 1

$g(x,y,z) = 1(\mu w.\ (H(x,z,w) \vee H(y,z,w)))$ = (Attenzione: $\vee$ è un or, quindi basta ne termini uno)

- $\downarrow$ se $z \in W_x \vee z \in W_y$
- $\uparrow$ altrimenti

$g(x,y,z) =$

- 1 se $z \in W_x \vee z \in W_y$
- $\uparrow$ altrimenti

$= 1(\mu w.\ H(x,z,w) \vee H(y,z,w))$ è calcolabile.

Per SMN esiste una funzione $K : N^2 \to N$ calcolabile totale tale che $\varphi_{K(x,y)}(z) = g(x,y,z)$

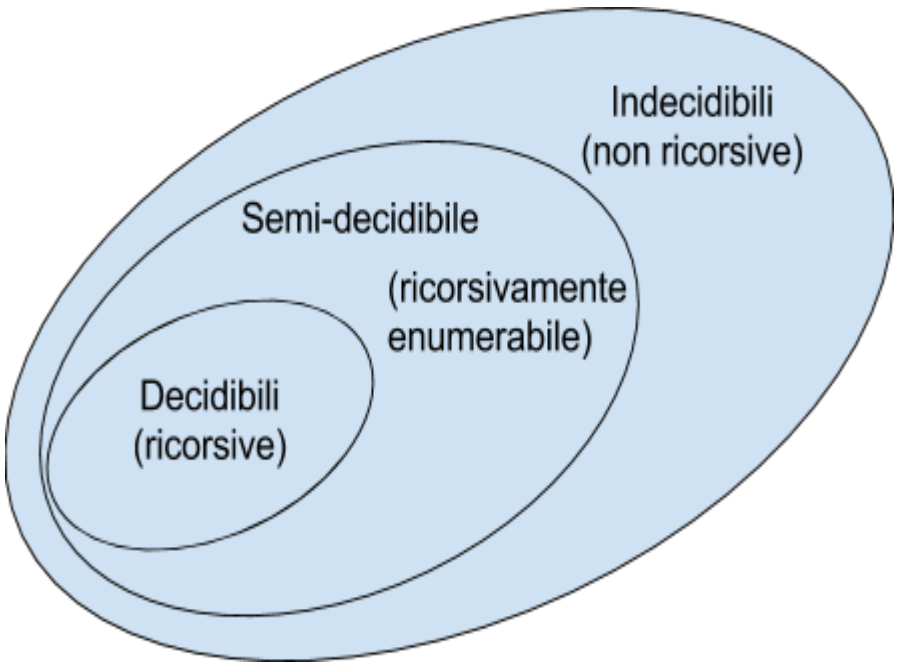
$z \in W_{K(x,y)}$ se e solo se $\varphi_{K(x,y)}\downarrow$ e solo se $z \in W_x \cup W_y \Rightarrow W_{K(x,y)} = W_x \cup W_y$

10 Insiemi Ricorsivi e Riduzione

Ogni proprietà “interessante” del comportamento dei programmi non è decidibile.
Le uniche proprietà decidibili sono quelle facili, mentre quelle complesse ed interessanti non sono decidibili o lo sono solo in parte (semi-decidibili).

Ricorsive = facili
Ricorsivamente Enumerabili = meno facili
Non Ricorsive = Difficili

Dal punto di vista della computabilità vale la seguente regola:



Quando la proprietà è decidibile l’insieme è ricorsivo.
Quando la proprietà è semi-decidibile l’insieme è ricorsivamente enumerabile.
Quando la proprietà è indecidibile l’insieme è non ricorsivo.

Insieme Ricorsivamente Enumerabile (RE):
Definizione:

$A \subseteq N$ RE se $SC_A(x) =$

- $1 \qquad x \in A$
- $\uparrow \qquad x \notin A$

$SC_A \rightarrow$ funzione semi-caratteristica di A

Tale funzione sa dirmi se tale proprietà è vera, ma non se essa sia falsa.

"$x \in A$" è semidecidibile.

Proposizione:

sia $A \subseteq N$, se A ricorsivo allora $A, \overline{A}$ RE.

Dimostrazione:

basta dimostrare che se $\chi_A(x) =$

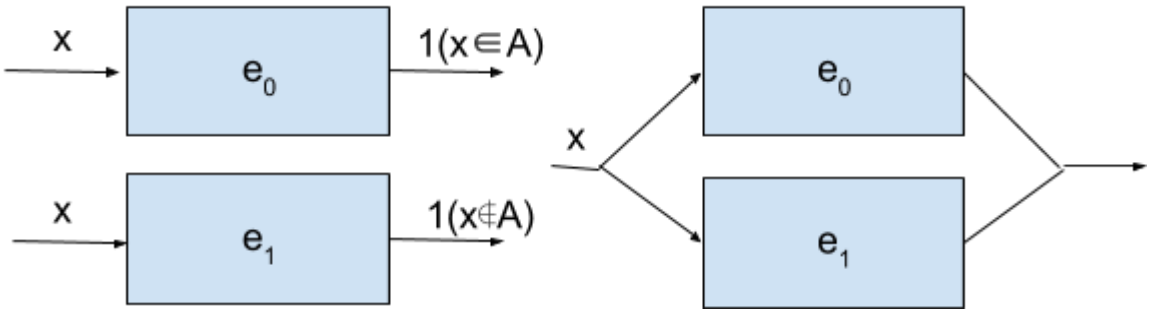
- $1 \qquad x \in A$
- $0 \qquad x \notin A$

calcolabile allora $SC_A(x) = 1(\mu w.\ \overline{sg}(\chi_A(x)))$ calcolabile $\Rightarrow$ A RE.

Essendo A RE $\Rightarrow$ $\overline{A}$ RE. (Dato A ricorsivo)

SC_A e $SC_{\overline{A}}$ calcolabili.

$\exists e_0, e_1$ tale che $\varphi_{e_0} = SC_A$ e $\varphi_{e_1} = SC_{\overline{A}}$



$\chi_A(x) = (\mu w.\ S(e_0, x, (w)_1, (w)_2)\ \mathrm{V}\ S(e_1, x, (w)_1, (w)_2))_1$ calcolabile, quindi A ricorsivo.

_ $A \subseteq N$ è ricorsivo:

$\chi_A : N \to N$

$\chi_A(x) =$

- $1 \qquad x \in A$
- $0 \qquad$ altrimenti

calcolabile $\to$ $x \in A$ è decidibile.

_ $P_r = \{x \mid x\ numero\ primo\}$ è ricorsivo:

$\chi_{P_r} : N \to N$

$\chi_{P_r}(x) =$

- $1 \qquad x$ numero primo
- $0 \qquad$ altrimenti

calcolabile $\to$ x numero primo è una proprietà decidibile.

_ $A \subseteq N$ finito è ricorsivo:

$A = \{a_1, ..., a_n\}$

$\chi_A(x) = \overline{sg}(\prod_{i=1}^{n} |x - a_i|)$

calcolabile $\to$ composizione di funzioni calcolabili.

_ $K = \{x \mid \varphi_x(x)\downarrow\}$ non ricorsivo

se per assurdo consideriamo K ricorsivo:

$f(x) =$

- $\varphi_x(x) + 1 \qquad x \in K$
- $0 \qquad x \notin K$

f calcolabile

$f(x) = \mu w.((x \in K \land (w)_1 = \varphi_x(x) \land (w)_2 = \#\, passi\ terminali) \lor (x \notin K \land (w)_1 = 0))_1 + 1$

scrivendola meglio:

$f(x) = (\mu w.((x \in K) \land S(x, x, (w)_1, (w)_2)) \lor ((x \notin K) \land (w)_1 = 0))_1 + 1$ calcolabile.

Ma il fatto che esso sia calcolabile è un problema, poiché:

$\forall x\ \varphi_x \neq f$:

- $x \in K \qquad f(x) = \varphi_x(x) + 1 \neq \varphi_x(x)$
- $x \notin K \qquad f(x) = 1 \neq \varphi_x(x)\uparrow$

Quindi dopo tutto questo ragionamento possiamo dire che K non sia ricorsivo.

Osservazione: $A, B \subseteq N$ ricorsivi

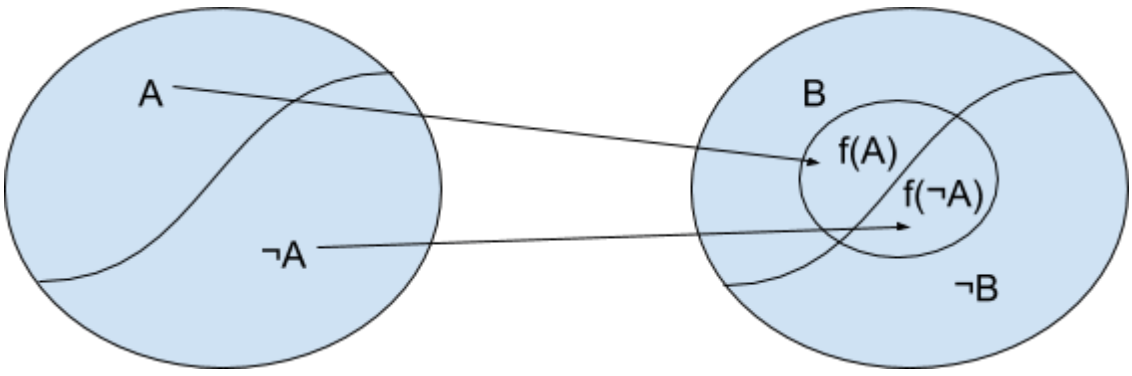
1. $\overline{A} = N \setminus A$ ricorsivo
2. $A \cap B$ ricorsivo
3. $A \cup B$ ricorsivo

Riduzione: ridurre un problema A ad un problema B più "semplice" (m-riducibilità).

$A, B \subseteq N$

Il problema $x \in A$ si riduce al problema $x \in B$ se esiste una funzione $f : N \to N$ calcolabile totale tale che $\forall x\ x \in A$ se e solo se $f(x) \in B$.

notazione: $A \leq_m B$



Osservazioni: se $A, B \subseteq N$ $A \leq_m B$

1. se B ricorsivo $\rightarrow$ A ricorsivo
2. se A non ricorsivo $\rightarrow$ B non ricorsivo

dim:

B ricorsivo

$\chi_B(x) =$

- 1 $x \in B$
- 0 altrimenti

calcolabile

$\chi_A(x) =$

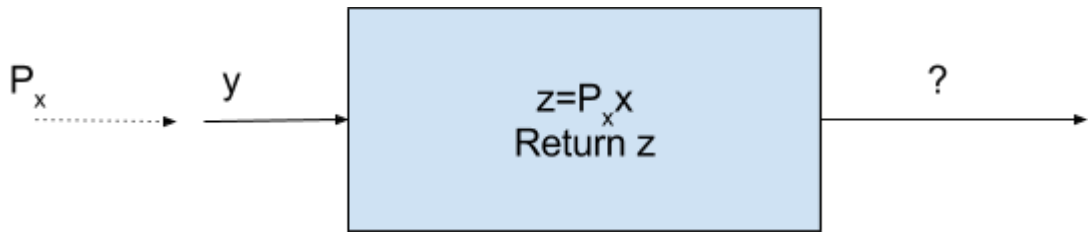
- 1 $x \in A$
- 0 altrimenti

$\chi_A(x) = \chi_B(f(x))$ calcolabile (per composizione) $\Rightarrow$ A ricorsivo.

Esempio: $T = \{x \mid \varphi_x \ totale\}$ non ricorsivo

$K \leq_m T$ (è invertito poiché voglio dimostrare la non ricorsività)

$f : N \rightarrow N$ calcolabile totale $x \in K$ se e solo se $f(x) \in T$



- sempre output se $x \in K$ $(P_x(x)\downarrow)$
- mai output se $x \notin K$ $(P_x(x)\uparrow)$

$y \rightarrow$ input (non viene usato)

$g(x,y) =$

- $\varphi_x(x)$ $x \in K$
- $\uparrow$ altrimenti

$g(x,y) = \varphi_x(x) = \psi_U(x,x)$ calcolabile

Quindi per SMN esiste una funzione $f : N \rightarrow N$ calcolabile totale tale che $\varphi_{f(x)}(y)$, quindi per cui f p una funzione di riduzione di K a T.

dimostrazione:

- $x \in K \Rightarrow f(x) \in T$
  $x \in K \Rightarrow \varphi_{f(x)}(y) = g(x,y) = \varphi_x(x)\downarrow \ \forall y$
  $x \in K \Rightarrow \varphi_{f(x)}$ totale, ossia $f(x) \in T$
- $x \notin K \Rightarrow f(x) \notin T$
  $x \notin K \Rightarrow \varphi_{f(x)}(y) = g(x,y) = \varphi_x(x)\uparrow \ \forall y$
  $x \notin K \Rightarrow \varphi_{f(x)}$ non totale, ossia $f(x) \notin T$

11 Teorema di Rice

Qualsiasi proprietà di un programma non è mai decidibile.
Per definire questo teorema dobbiamo introdurre il concetto di Insieme Saturato

Insieme Saturato:
$A \subseteq N$, se $x \in A$ ed $y \in N$ e $\varphi_x = \varphi_y \Rightarrow y \in A$. ($\varphi_x = \varphi_y$ calcolano la medesima funzione)
La proprietà non dipende dal programma ma dalla funzione che essa calcola.
Un insieme saturato contiene tutti gli indici (che sono infiniti) dei programmi che calcolano delle funzioni con una particolare caratteristica comune.

Un altro modo per dirlo:
A è saturato se e solo se esiste $Å \subseteq \zeta$ tale che $A = \{x \mid \varphi_x \in Å\}$

Esempio di insieme saturato: $T = \{x \mid \varphi_x \ totale\} = \{x \mid \varphi_x \in \tau\}$
$\tau = \{f \mid f \ totale\}$

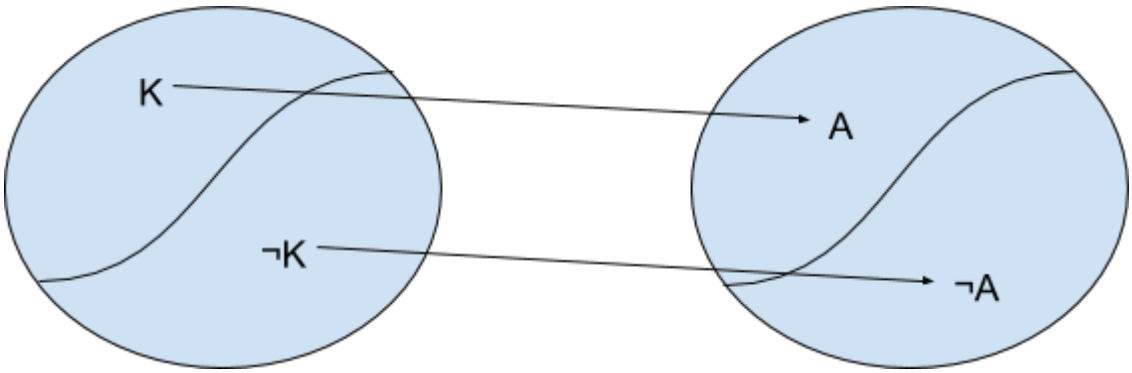
$K = \{x \mid x \in W_x\}$ non è saturato poiché non dipende dalla funzione.
Dimostrazione:
Mostro che esiste una certa funzione calcolabile $e \in N$ tale che
$\varphi_e(x) =$
- e se $x = e$
- $\uparrow$ altrimenti

$= \{(e, e)\}$ $e \in K$ poiché $e \in W_e$
Esiste una funzione $e' \in N$ tale che $\varphi_{e'} = \varphi_e$ ed $e' \neq e$
$\varphi_{e'}(e') = \varphi_e(e') \uparrow$ (essendo tale funzione definita soltanto su e)

Quindi ricapitolando:
$e \in K$
e' tale che $\varphi_{e'} = \varphi_e$ (calcolano la stessa funzione)
$e' \notin K$

Teorema di Rice (1953)
Sia $A \subseteq N$ saturato, $A \neq 0$ ed $A \neq N$ allora A non ricorsivo.
Non possiamo sapere se un programma abbia una proprietà specifica con esattezza.

Dimostrazione: $K \leq_m A$



Sia $e_0 \in N$, $\varphi_{e_0} \neq 0 \qquad 0(x)\uparrow \ \forall x$

$x \in K$ se e solo se $f(x) \in A$

2 casi

1) $e_0 \in \overline{A}$

Sia $e_1 \in A$ qualunque (esiste $A \neq 0$)

$g(x,y) =$

- $\varphi_{e_1}(y)$ $\qquad x \in K$
- $\varphi_{e_0}(y)$ $\qquad x \notin K$

$\Downarrow$

$g(x,y) =$

- $\varphi_{e_1}(y)$ $\qquad x \in K$
- $\uparrow$ $\qquad$ altrimenti

$\Downarrow$

$g(x,y) = \varphi_{e_1}(y) * 1(\varphi_x(x)) = \varphi_{e_1}(y) * 1(\psi_U(x,x))$ $\qquad$ calcolabile

Per il teorema SMN esiste una funzione $f : N \to N$ calcolabile totale tale che

$g(x,y) = \varphi_{f(x)}(y) =$

- $\varphi_{e_1}(y)$ $\qquad x \in K$
- $\varphi_{e_0}(y)$ $\qquad$ altrimenti

f è una funzione di riduzione

Dim:

_ $x \in K \Rightarrow \varphi_{f(x)}(y) = \varphi_{e_1}(y) \ \forall y$

$\Rightarrow \varphi_{f(x)} = \varphi_{e_1}$

Sapendo che $e_1 \in A$ ed A è saturo allora $f(x) \in A$

_ $x \notin K \Rightarrow \varphi_{f(x)}(y) = \varphi_{e_0}(y) \ \forall y$

$\Rightarrow \varphi_{f(x)} = \varphi_{e_0}$

Sapendo che $e_0 \notin A$ ed A è saturo allora $f(x) \notin A$

Dato che K non è ricorsivo e che $K \leq_m A \ \Rightarrow \ A$ non ricorsivo.

(Attenzione: tutta la dimostrazione si basa sulla supposizione che $\varphi_{e_0} \neq 0,$ cosa che potrebbe essere non vera)

2) $e_0 \in A$

$B = \overline{A}$ $0 \neq B$

$N \neq B$

B saturo

$e_0 \notin B$

Per il punto 1 $\Rightarrow$ B non ricorsivo

Un insieme è non ricorsivo se e solo se il suo complementare non lo è a sua volta.

$B = \overline{A}$ non ricorsivo $\Rightarrow$ A non ricorsivo

12 Struttura e Proiezione

Recap: Insiemi r.e. $SC_A : N \to N$

- $A \subseteq N$ r.e. se $SC_A(x) =$
  - $1 \quad x \in A$
  - $\uparrow \quad$ altrimenti

  Calcolabile
- A ricorsivo se e solo se $A, \overline{A}$ r.e.

Esistono anche insieme r.e. non ricorsivi

$K = \{x \mid x \in W_x\} = \{x \mid \varphi_x(x)\downarrow\}$ r.e.

$SC_K =$

- $1 \quad x \in K$
- $\uparrow \quad$ altrimenti

$\Downarrow$

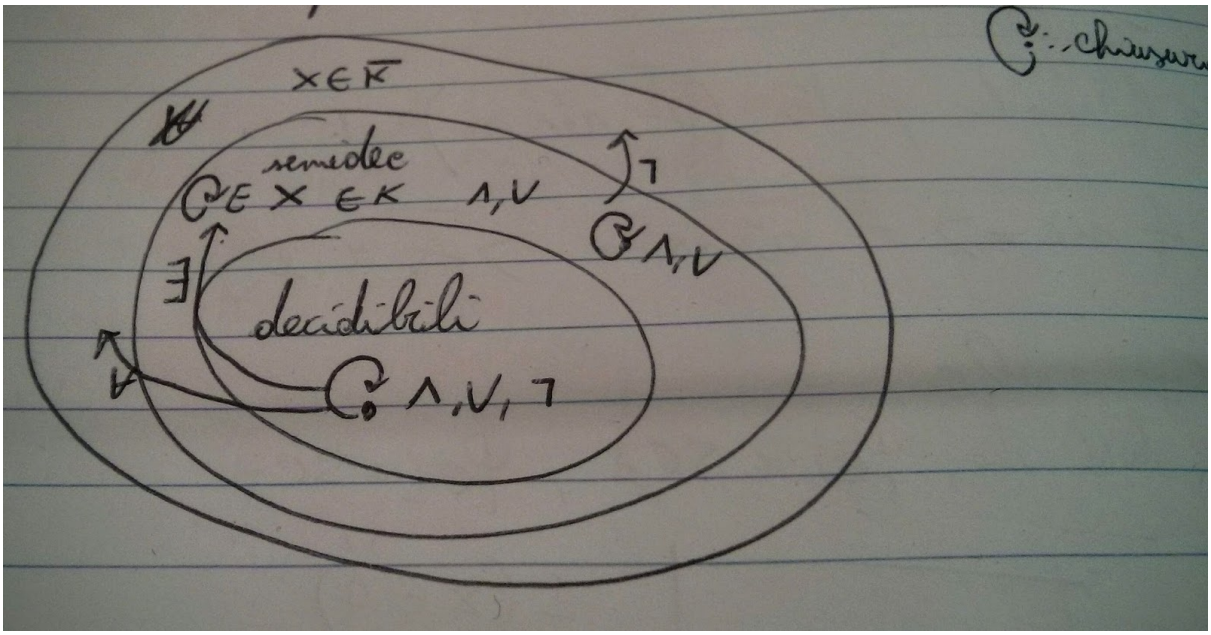
$SC_K =$

- $1 \quad \varphi_x(x)\downarrow$
- $\uparrow \quad$ altrimenti

$\Downarrow$

$SC_K = 1(\varphi_x(x)) = 1(\psi_U(x,x))$ calcolabile

In questo caso $\overline{K}$ non r.e. poiché K r.e. ma non ricorsivo.



(con onesta vi confido che non avevo voglia di rifare lo schema quindi l'ho preso dai miei appunti cartacei)

Teorema di Struttura:
Sia $P(x^{\rightarrow})$ predicato ($x \in N^K$)
$P(x^{\rightarrow})$ semi-decidibile se e solo se esiste $Q(t, x^{\rightarrow})$ decidibile tale che $P(x^{\rightarrow}) = \exists t.\ Q(t, x^{\rightarrow})$
(si vuole sapere in quale punto vale il predicato, bisogna considerare il fatto che tale punto potrebbe anche non esistere)

Dimostrazione:
Sia $P(x^{\rightarrow}) = \exists t.\ Q(t, x^{\rightarrow})$ con Q decidibile

$SC_P =$

- 1 se $P(x^{\rightarrow})$
- $\uparrow$ altrimenti

$\Downarrow$

$SC_P = 1(\mu t.\ |\chi_{Q(t,x)} - 1|)$ calcolabile (perché si tratta di una composizione di funzioni calcolabili e poiché Q è decidibile)

Quindi $P(x^{\rightarrow})$ semidedibile.
Sia $P(x^{\rightarrow})$ semidecidibile allora $SC_P =$

- 1 se $P(x^{\rightarrow})$
- $\uparrow$ altrimenti

calcolabile.
Ovvero $SC_P = \varphi_e$ per qualche $e \in N$ (esiste un programma che la calcola)

Il predicato $P(x^{\rightarrow})$ vale: se e soltanto se $SC_P = \varphi_e = 1$
se e soltanto se $\varphi_e(x^{\rightarrow})\downarrow$
se e soltanto se $\exists t.\ H(e, x^{\rightarrow}, t)$ (che è decidibile)
Indicato con $Q(t, x^{\rightarrow}) = H^{(K)}(e, x^{\rightarrow}, t)$ decidibile e $P(x^{\rightarrow}) = \exists t.\ Q(t, x^{\rightarrow})$

$P(x^{\rightarrow}) = \exists t.\ H(e, x^{\rightarrow}, t)$ semi-decidibile

Teorema di Proiezione: (chiusura per $\exists$ dei predicati semi-decidibili)
Sia $P(x, y^{\rightarrow}) \subseteq N^{K+1}$ semi-decidibile $\Rightarrow$ $\exists x.\ P(x, y^{\rightarrow})$ semi-decidibile

Dimostrazione:
Per ipotesi $P(x, y^{\rightarrow})$ semi-decidibile.
Per il teorema di struttura $P(x, y^{\rightarrow}) = \exists t.\ Q(t, x, y^{\rightarrow})$ con Q decidibile
$P'(y^{\rightarrow}) = \exists x.\ \exists t.\ Q(t, x, y^{\rightarrow}) = \exists w.\ Q((w)_1, (w)_2, y^{\rightarrow})$
$\Downarrow$
$Q'(w, y^{\rightarrow})$ decidibile.
Quindi per il teorema di struttura $P'(y^{\rightarrow})$ semi-decidibile.
$Q'(w, y^{\rightarrow}) = Q((w)_1, (w)_2, y^{\rightarrow})$
$\chi_{Q'}(w, y^{\rightarrow}) = \chi_Q((w)_1, (w)_2, y^{\rightarrow})$ calcolabile

Chiusura per $\wedge,\vee$

Siano $P_1(x^{\rightarrow}), P_2(x^{\rightarrow})$ semi-decidibili, allora:

1) $P_1(x^{\rightarrow}) \vee P_2(x^{\rightarrow})$ semi-decidibile
2) $P_1(x^{\rightarrow}) \wedge P_2(x^{\rightarrow})$ semi-decidibile

Dimostrazione:

Per il teorema di struttura $P_1(x^{\rightarrow}) = \exists t.\ Q_1(t, x^{\rightarrow}),\ P_2(x^{\rightarrow}) = \exists t.\ Q_2(t, x^{\rightarrow})$ con Q_1 e Q_2 decidibili.

1) $P_1(x^{\rightarrow}) \vee P_2(x^{\rightarrow}) = \exists t.\ Q_1(t, x^{\rightarrow}) \vee \exists t.\ Q_2(t, x^{\rightarrow}) = \exists t.\ (Q_1(t, x^{\rightarrow}) \vee Q_2(t, x^{\rightarrow}))$
   Essendo Q_1 e Q_2 decidibili $\Rightarrow$ $P_1(x^{\rightarrow}) \vee P_2(x^{\rightarrow})$ semi-decidibile per il teorema di struttura.
2) $P_1(x^{\rightarrow}) \wedge P_2(x^{\rightarrow}) = \exists t.\ Q_1(t, x^{\rightarrow}) \wedge \exists t.\ Q_2(t, x^{\rightarrow}) \neq \exists t.\ (Q_1(t, x^{\rightarrow}) \wedge Q_2(t, x^{\rightarrow}))$
   $\Downarrow$
   $= \exists w.\ (Q_1((w)_1, x^{\rightarrow}) \wedge Q_2((w)_2, x^{\rightarrow}))$
   Essendo Q_1 e Q_2 decidibili $\Rightarrow$ $P_1(x^{\rightarrow}) \wedge P_2(x^{\rightarrow})$ semi-decidibile per il teorema di struttura.

Non-Chiusura per $\forall$

$Q(t, x) = \neg H(x, x, t)$ decidibile

$Q'(x) = \forall t.\ Q(t, x) = x \in \overline{K}$ $\quad (K = \{x \mid \varphi_x(x)\downarrow\} \rightarrow \overline{K} = \{x \mid \varphi_x(x)\uparrow\})$

(Quindi c'è si potrebbe saltare nei non ricorsivi)

$x \in k$ non decidibile

$\neg(x \in k) = x \in \overline{k}$ non semi-decidibile

Riducibilità ed r.e.

$A, B \subseteq N$ e $A \leq_m B$

1) se B r.e. $\Rightarrow$ A r.e.
2) se A non r.e. $\Rightarrow$ B non r.e.

Dimostrazione:

1) se B r.e. $\Rightarrow$ A r.e.
   Sia $f : N \rightarrow N$ calcolabile totale e $x \in A$ se e solo se $f(x) \in B$
   $SC_A(x) = SC_B(f(x))$ calcolabile (composizione di funzioni calcolabili)
2) se A non r.e. $\Rightarrow$ B non r.e.
   Si dimostra in modo analogo ad 1)

Enumerabilità:

A enumerabile se $f : N \rightarrow A$ suriettiva $f(0), f(1), f(2), \ldots$

$\Downarrow$

$f : N \rightarrow N$ ed $img(f) = A$

$\downarrow$

cod

Teorema di etimologia:

Sia $A \subseteq N$ con:

$A \neq 0$ oppure A r.e. se e solo se esiste $f : N \to N$ calcolabile totale tale che $A = cod(f)$

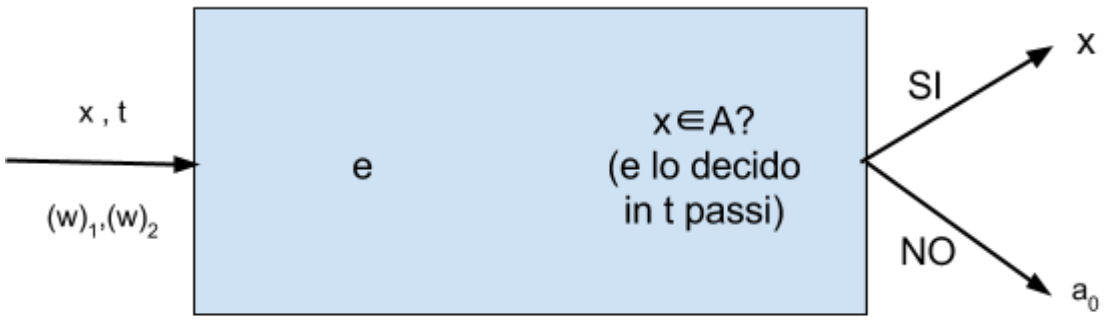
Dimostrazione:

Sia A r.e. :

- se $A = 0$ ok
- sia $A \neq 0$

  sia $a_0 \in A$ fissato

  sia e tale che $\varphi_e = SC_A$



$f(w) =$

- $(w)_1$ se $H(e,(w)_1,(w)_2)$
- a_0 altrimenti

⇓

$= (w)_1 * \chi_H(e,(w)_1,(w)_2) + a_0 * |1 - \chi_H(e,(w)_1,(w)_2)|$ calcolabile totale

$A = img(f)$

⊆) se $x \in A \Rightarrow \varphi_e(x) = SC_A(x) = 1$

$\Rightarrow \exists t.\ H(e,x,t)$

$\Rightarrow w \in N$ tale che $(w)_1 = x$, $(w)_1 = t$

$f(w) = x \Rightarrow x \in cod(f)$

⊇) sia $z \in cod(f)$, ovvero $z = f(w)$ per $w \in N$

2 possibilità:

- $f(w) = a_0 \in A$
- $f(w) = (w)_1 \to H(e,(w)_1,(w)_2) = \varphi_e((w)_1)\downarrow = 1$

$\downarrow$

SC_A

$\Rightarrow (w)_1 \in A$

Se $A = 0 \Rightarrow A$ ricorsivo $\Rightarrow$ A r.e.

Se $A = cod(f)$ f totale calcolabile

$SC_A(x) = 1(\mu w.\ |f(w) - x|)$ calcolabile $\Rightarrow$ A r.e.

13 Teorema di Rice-Shapiro

Le proprietà del comportamento (funzione calcolata) dei programmi possono essere semi-decidibili se tale proprietà è Finitaria.
Per definire in modo opportuno tale concetto bisogna introdurre le funzioni finite.

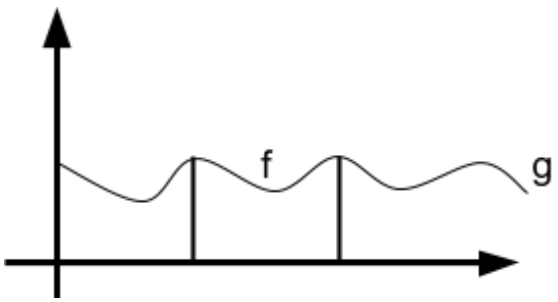
Funzione Finita:
Funzione $\vartheta : N \to N$ con $dom(\vartheta)$ finito.
$\vartheta(x) =$

- y_1 se $x = x_1$
- .
- .
- .
- y_n se $x = x_n$
- $\uparrow$ altrimenti

$\Downarrow$

$= \{(x_1, y_1), ..., (x_n, y_n)\}$

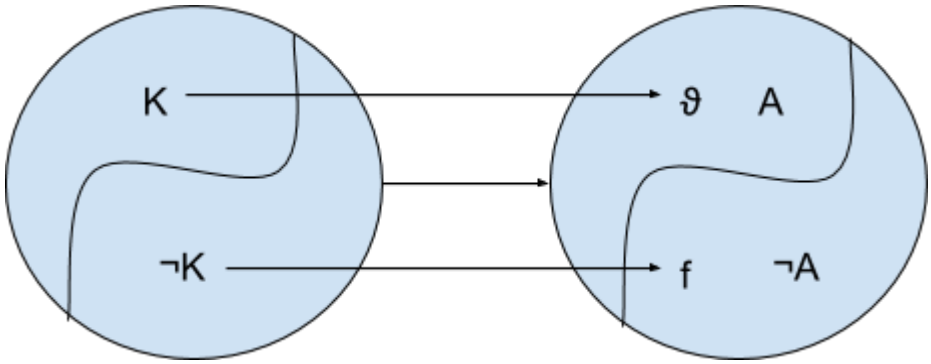
$f, g : N \to N$
$f \subseteq g$ se $\forall x \quad f(x)\downarrow \Rightarrow g(x)\downarrow$ e $g(x) = f(x)$



Teorema di Rice-Shapiro:
Sia $Å \subseteq \zeta$
Sia $A = \{x \mid \varphi_x \in Å\}$
Se A r.e. allora $\forall f \in \zeta \ (f \in Å \Leftrightarrow \exists \vartheta \subseteq f \ finito \ \vartheta \in Å)$
Attenzione: se una proprietà è r.e., allora essa è finitoria, ma non vale il contrario
(potrebbe essere finitoria ma non r.e.)

Dimostrazione:

1. $\exists f \ f \notin A$ e $\exists \vartheta \subseteq f \ \vartheta \in A \Rightarrow A$ non r.e.
   $f \notin Å$ ed esiste $\vartheta \subseteq f$ finito $\vartheta \in A \Rightarrow A$ non r.e.
   Sia $f \notin Å$ e sia $\vartheta \subseteq f$ finito $\vartheta \in A$
   K funzione di riduzione tale che: $K \leq_m A$



$g(x,y) =$

- $\vartheta(y)$ $x \in \overline{K}$
- $f(y)$ $x \in K$

$\Downarrow$ (è la stessa cosa scritta in maniera differente)

$g(x,y) =$

- $\uparrow$ $x \in \overline{K}$ e $y \notin dom(\vartheta)$ ($\vartheta(y)$ è stato sostituito con $f(y)$ poiché
- $f(y)$ $x \in \overline{K}$ e $y \in dom(\vartheta)$ ϑ è un sottoinsieme di f, quindi dove
- $f(y)$ $x \in K$ f è definito anche ϑ lo è)

$\Downarrow$

$g(x,y) =$

- $f(y)$ $x \in K$ oppure $y \in dom(\vartheta)$
- $\uparrow$ altrimenti

$Q(x,y) =$ "$x \in K$" $\vee$ "$y \in dom(\vartheta)$" semi-decidibile (sia $x \in K$ che $y \in dom(\vartheta)$ sono semi-decidibili)

$= f(y) * SC_Q(x,y)$ calcolabile

Per il teorema SMN $g(x,y) = \varphi_{S(x)}(y)$ per la funzione $S : N \to N$ calcolabile totale.

S è funzione di riduzione.

$\overline{K} \leq_m A$

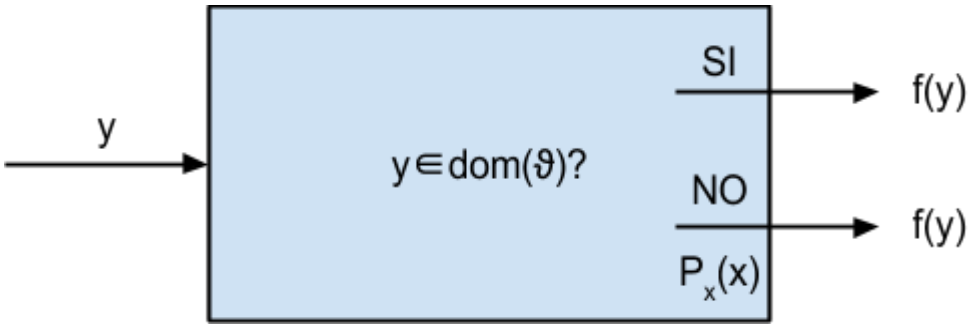
- $x \in \overline{K} \Rightarrow \varphi_{S(x)}(y) = g(x,y) =$
  - $f(y)$ se $y \in dom(\vartheta)$
  - $\uparrow$ altrimenti

  $= S(x) \in A$
- $x \in \overline{K} \Rightarrow \varphi_{S(x)}(y) = g(x,y) = f(y)\ \forall y$

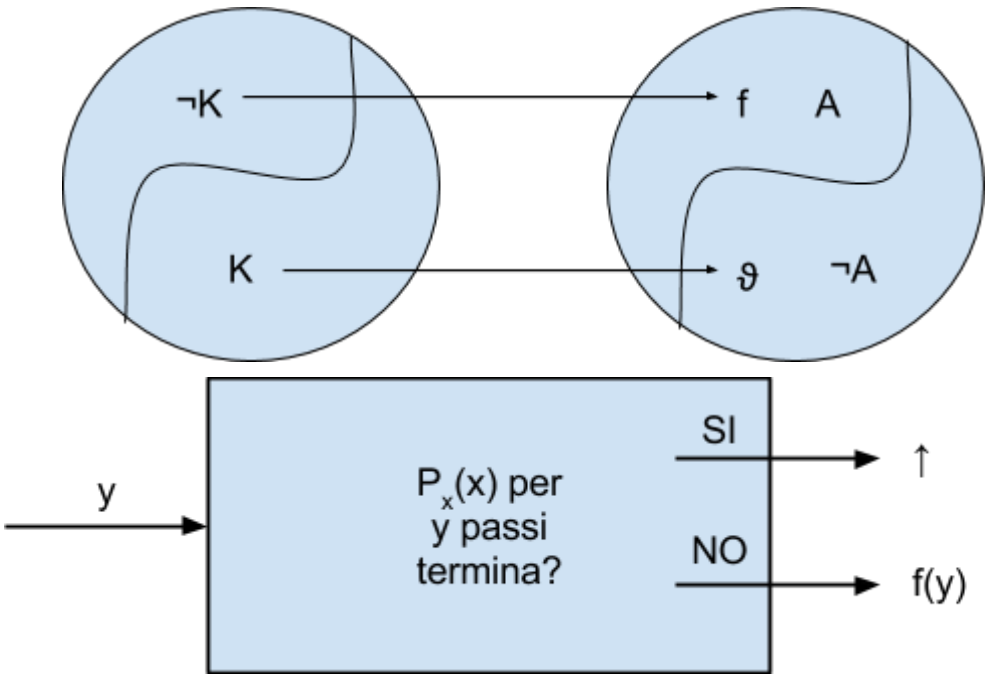
  $\Rightarrow \varphi_{S(x)} = f \notin A$

  $\Rightarrow S(x) \notin A$

  $\Rightarrow \overline{K} \leq A \Rightarrow A$ non r.e.



2. $\exists f\ f \in A$ e $\exists \vartheta \subseteq f\ \ \vartheta \notin A \ \Rightarrow\ A$ non r.e.
   $f \in$ Å e per ogni $\vartheta \subseteq f\ \ \vartheta \notin$ Å
   $\overline{K} \leq A$



$g(x,y) =$

- $f(y)$ se $\neg H(x,x,y)$
- $\uparrow$ altrimenti

$\Downarrow$

$g(x,y) = f(y) * 1(\mu w.\ \chi_H(x,x,y))$

Per SMN esiste una funzione $S : N \to N$ calcolabile totale tale che $\varphi_{S(x)} = g(x,y)$

S è una funzione di riduzione di $\overline{K} \leq A$

$\to x \in \overline{K} \Rightarrow \varphi_{S(x)} = f(y)\ \forall y$

$\Uparrow$

$\forall y\ \neg H(x,x,y)$

$\Rightarrow \varphi_{S(x)} = f \in$ Å $\Rightarrow S(x) \in A$

Quindi se $x \in K \Rightarrow \exists y_0$ tale che $H(x,x,y_0)$

prendo il minimo, ovvero:

$\neg H(x,x,y)\ \forall y < y_0$ ed $H(x,x,y)\ \forall y \geq y_0$

$\varphi_{S(x)}(y) = g(x,y) =$

- $f(y)$ se $y < y_0$
- $\uparrow$ altrimenti

$\varphi_{S(x)} \subseteq f$ ed è finito ($dom(\varphi_{S(x)} \subseteq [0,y_0) \Rightarrow \varphi_{S(x)} \notin$ Å $\Rightarrow S(x) \notin A$)

Quindi A non è r.e.

14 Teoremi di ricorsione

1° Teorema di Ricorsione:
Per arrivare alla definizione del teorema bisogna prima introdurre il concetto di funzione funzionale.

Funzione Funzionale:
Si tratta di una funzione calcolabile totale $\varphi : F(N^k) \to F(N^k)$ (da insieme di funzioni ad insieme di funzioni)
dove $F(N^k) = \{f \mid f : N^k \to N^k\}$
Una funzione funzionale è ricorsiva (calcolabile) se esiste un φ calcolabile:

$\varphi(f)(x) = \varphi(\hat{\vartheta}, x) \qquad \vartheta \subseteq f$ per qualche $\vartheta \subseteq f$ finito
(definito su un numero finito di punti)

- Funzioni finite → Numeri

$\hat{\vartheta} =$

- 0 se $\vartheta = 0$
- $\prod_{x \in dom(\vartheta)} P_x^{\vartheta(x)+1}$ altrimenti

$z \in N \qquad z = \hat{\vartheta}$

- $x \in dom(\vartheta) \Leftrightarrow P_x \mid z$
- $app(z, x) =$
  - $(z)_x - 1$ se $x \in dom(z)$
  - $\uparrow$ altrimenti

φ ricorsivo se esiste una funzione $\varphi : N^{h+1}$ tale che per ogni $f \in F(N^k)$ e per ogni $x^{\rightarrow} \in N^h,\ y \in N$:

$\varphi(f)(x^{\rightarrow}) = y$ se e solo se esiste $\vartheta \subseteq f$ finito
$\varphi(\hat{\vartheta}, x^{\rightarrow}) = y$
(Guardo solamente dove vale su f un numero finito di punti)

Esempio:

$\varphi : F(N^1) \to F(N^1)$
$\varphi(f)(x) = f(x) + 1$
$\varphi(\hat{\vartheta}, x) = \vartheta(x) + 1 = applicazione(\hat{\vartheta}, x) + 1$

Esempio: funzione di Ackermann

$\varrho : F(N^2) \to F(N^2)$
$\varrho(f)(0, y) = y + 1$
$\varrho(f)(x + 1, 0) = f(x, 1)$
$\varrho(f)(x + 1, y + 1) = f(x, f(x + 1, y))$

Proposizione:
Sia $\varphi : F(N^k) \to F(N^k)$ ricorsivo ed $f : N^k \to N$ calcolabile, allora $\varphi(f) : N^k \to N$ calcolabile.

Myhill-Shepherdson

φ ricorsivo ed f calcolabile $\Rightarrow$ $\varphi(f)$ calcolabile

φ_e $\varphi(\varphi_e)$ calcolabile

$\Downarrow$

$\varphi_{e'} = \varphi_{h(e)}$

Funzione Estensionale:

Diciamo che la funzione $h : N \to N$ è estensionale se $\forall e, e'$ tali che $\varphi_e = \varphi_{h(e')}$

(Una funzione estensionale è una funzione basata sul significato della funzione, non sulla sua sintassi)

Teorema di Myhill-Shepherdson:

Parte 1:

Sia $\varphi : F(N^k) \to F(N^k)$ una funzione funzionale ricorsiva, allora esiste una funzione $h : N \to N$ calcolabile totale estensionabile tale che:

$\forall e \in N$ $\varphi(\varphi_e) = \varphi_{h(e)}$

Parte 2:

Sia $h : N \to N$ una funzione calcolabile totale ed estensionabile, allora esiste un unico funzionale ricorsivo $\varphi : F(N^k) \to F(N^k)$ tale che $\varphi(\varphi_e^{(k)}) = \varphi_{h(e)}^{(h)}$

Primo Teorema di ricorsione: (pagina 192)

Sia $\varphi : F(N^k) \to F(N^k)$ funzionale ricorsivo, allora φ ha un minimo punto fisso $f_\varphi : N^k \to N^k$ calcolabile

1) $\varphi(f_\varphi) = f_\varphi$
2) $\forall g \ \varphi(g) = g \Rightarrow f_\varphi \subseteq g$
3) f_φ calcolabile

Inoltre $f_\varphi = \bigcup_{k \in N} \varphi^n(0)$

Esempio: x-1

$\varphi(f)(x) =$

- 0 se $x = 0, 1$
- $f(x-1)+1$ altrimenti

Quindi il teorema ci dice che esiste un punto fisso $\varphi f_\varphi(n)$

$\varphi^n(0)(3) = \varphi^{n-1}(0)(3-1)+1 = (\varphi^{n-2}(0)(2-1)+1)+1 = 2$

$\varphi^n(0)(1) = \varphi^n(0)(0) = 0$

(Dire che f_φ è un punto fisso vuol dire applicare f_φ un certo numero di volte fino ad arrivare ad un caso base)

Esempio: Ackermann

$\psi(f)(0, y) = y + 1$

$\psi(f)(x+1, 0) = f(x, 1)$

$\psi(f)(x+1, y+1) = f(x, f(x+1, y))$

Ricorsivo $\rightarrow$ il punto fisso è calcolabile $\psi_{ack} \subseteq f$

$f : N^k \rightarrow N \qquad g : N^{k+2} \rightarrow N$

$\varphi\ (h)\ (x^{\rightarrow}, 0) = f(x^{\rightarrow})$

$\varphi\ (h)\ (x^{\rightarrow}, y+1) = g(x^{\rightarrow}, y, h(x^{\rightarrow}, y))$

$h = \varphi(h)$

Esempio: Minimalizzazione

$\mu y.\, f(x^{\rightarrow}, y)$

$h(x^{\rightarrow}, y) =$

- y se $h(x^{\rightarrow}, y) = 0$
- $h(x, y+1)$ se $h(x^{\rightarrow}, y)\downarrow \neq 0$

Si tratta di un funzionale ricorsivo poiché usa h su un numero finito di punti.

$\mu y.\, f(x^{\rightarrow}, y) = f_{\varphi_\mu}(x^{\rightarrow}, 0)$

φ_μ (h)($x^{\rightarrow}, y$)=

- y se $h(x^{\rightarrow}, y) = 0$
- $h(x, y+1)$ se $h(x^{\rightarrow}, y)\downarrow \neq 0$

2° Teorema di Ricorsione:

Il Primo teorema diceva che:

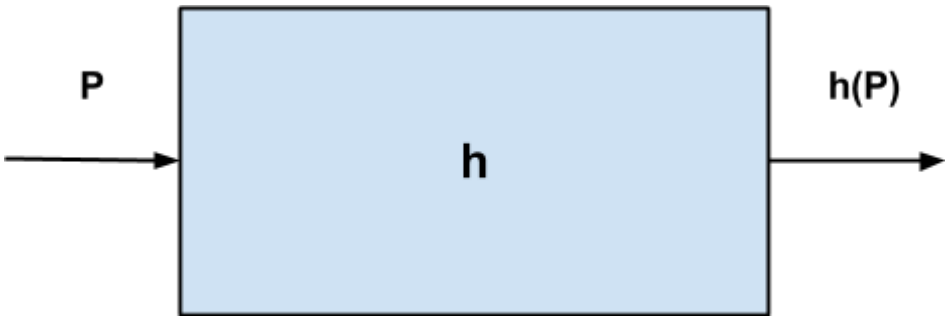
$e \in N \qquad h : N \to N$ calcolabile totale estensionabile

$\varphi_h(\varphi_e) = \varphi_{h(e)}$

esiste $e \qquad \varphi_e = \varphi_h(\varphi_e) = \varphi_{h(e)}$

Il Secondo teorema dice che il presupposto di estensionabilità non è necessario.

$$P_e \to P_{h(e)}$$



Secondo Teorema di Ricorsione: (pagina 200)

Sia $h : N \to N$ calcolabile totale, allora esiste $n \in N$ tale che $\varphi_{h(n)} = \varphi_n$

Dimostrazione:

$g(x,y) = \varphi_{h(\varphi_x(x))}(y) = \psi_U(h(\psi_U(x,x)), y)$ calcolabile

$\Downarrow \qquad \Downarrow$

$\psi_U(x,x)$ Funzione universale

Per il teorema SMN esiste una funzione $S : N \to N$ calcolabile totale tale che:

$\psi_U(h(\psi_U(x,x)), y) = \varphi_{S(x)}(y)$

$\Downarrow$ sia e tale che $\varphi_e = S$

$= \varphi_{\varphi_e(x)}(y)$

Per $x = e$

$\varphi_{h(\varphi_e(e))}(y) = \varphi_{\varphi_e(e)}(y)$

dato $n = \varphi_{\varphi_e(e)}(y)$

$\varphi_{h(n)}(y) = \varphi_n(y) \ \forall y$

$\Downarrow$

$\varphi_{h(n)} = \varphi_n$

(n non può essere $\uparrow$ poiché $n = \varphi_e(e)$ ed S sono totali, quindi sempre definiti)

Se due programmi calcolano la stessa cosa, ossia con lo stesso input danno il medesimo output, allora entrambi i programmi calcolano la medesima funzione.

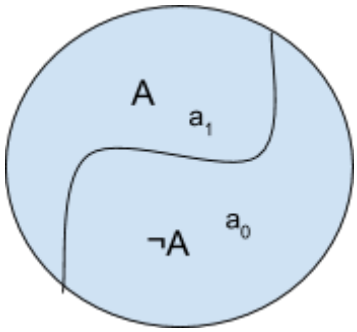
Esempio: Teorema di Rice

$A \neq 0,\ N$ saturato $\Rightarrow$ A non ricorsivo

Dimostrazione: consideriamo per assurdo che A sia ricorsivo

sia $a_1 \in A$

sia $a_0 \notin A$



$f(x) =$

- a_0 se $x \in A$
- a_1 se $x \notin A$

$\Downarrow$

$f(x) = a_0 * \chi_A(x) + a_1 * \chi_{\neg A}(x)$ calcolabile totale

Per il secondo teorema di ricorsione esiste un n tale che:

$$\varphi_n = \varphi_{f(n)}$$

- $n \in A \Rightarrow f(n) = a_0 \notin A \Rightarrow \varphi_n \neq \varphi_{f(n)}$ ASSURDO!!!
- $n \notin A \Rightarrow f(n) = a_1 \in A \Rightarrow \varphi_n \neq \varphi_{f(n)}$ ASSURDO!!!

Quindi A non ricorsivo.