RIASSUNTO

1. Cos'è un algoritmo?

2. Si possono enumerare

3. A_i calcola φ_i (l'algoritmo i-esimo calcola la funzione i-esima)

   ↕

   Tesi di Church Turing

   Per qualunque algoritmo si può scrivere una macchina di Turing

4. Se un formalismo esprime solo funzioni calcolabili totali allora non le esprime tutte

Th. 1 Le funzioni calcolabili sono $\#\mathbb{N}$

Esistono funzioni non calcolabili

Th. 2 Esistono un predicato e una funzione calcolabili totali e ricorsivi primitivi $T(i,x,y)$, $U(y)$ tali che

$$\forall i,x.\ \varphi_i(x) = U(\mu y.\ T(i,x,y))$$

↑

La funzione calcolata dalla i-esima macchina

Padding lemma $\forall i.\ \exists A_i$ t.c. $\forall j \in A_i.\ \varphi_i = \varphi_j$

Gli insiemi A_i si possono costruire con una funzione rp

Assurde: Per ogni indice io posso costruire con una funzione rp un insieme infinito di macchine equivalenti

TEOREMA di ENUMERAZIONE

(della MACCHINA di TURING UNIVERSALE)

Esiste z indice della mdT universale tale che prende in input una macchina e i suoi dati e restituisce lo stesso valore

$$\exists z.\ \forall (i,x).\ \varphi_i(x) = \varphi_z(i,x)$$

È un ragionamento analogo al fatto che un interprete sia un programma che prende in input un programma e dei dati sia dati sia programmi.

L'insieme dei dati non è isomorfo a quello delle funzioni se non si lavora con funzioni calcolabili

Poiché si lavora solo con $f: \mathbb{N} \to \mathbb{N}$, φ_z prende in input la codifica della coppia (i,x).

ATTENZIONE ALL'ORDINE dei QUANTIFICATORI

Dim:

$$\varphi_i(x) = \{\text{Th. 2}\}\ U(\mu y.\ T(i,x,y))$$

Questa è μ-ricorsiva, è calcolabile

$\Rightarrow$ {Th. Church Turing}

$\varphi_z(i,x)$ Q.E.D.

$\exists z.\ \varphi_z = U(\dots)$

Considerato che l'indice può diventare argomento, come fare il contrario?

TEOREMA del PARAMETRO (S-m-n)

1) $\exists s$ funzione calcolabile totale iniettiva t.c.

$\forall x, y. \quad \lambda z.\, \varphi_x(y, z)$ (la considero come una funzione in cui l'unica variabile è z)

$$= \varphi_{s(x,y)}(z)$$

Suppongo che y sia un parametro fissato e un valore qualunque z

invece che usare l'algoritmo x-esimo si può usare un algoritmo che dipende solo da z.

Dimo (sketch)

$\varphi_x(y, z)$

(a) $\hookrightarrow M_x$ presa attraverso x

(b) Scrivo $\langle y, z \rangle$ sul nastro di M_x

(a), (b) sono un algoritmo

{Church - Turing} $\Longrightarrow$ hanno un indice n

$$n = s(x, y)$$

$s(x, y)$ deve essere iniettiva

Per il padding lemma posso costruire s strettamente crescente

$\Rightarrow s$ iniettiva

Q.E.D.

2) $\forall x, y \dots y_m. \quad \lambda z \dots z_n. \varphi_x(y \dots y_m, z \dots z_n) = \varphi_{s(x, y \dots y_m)}(z \dots z_n)$

Esempio di valutatore parziale:

(1) prodotto := 0;

while $y>0$ do

prodotto := prodotto + 2;

$y := y - 1$;

Calcolo $\varphi_x(2, 2)$

(2) while $y>0$ do

C { prodotto := prodotto + 2;
  { $y := y - 1$;

prodotto := 0;

prodotto := prodotto + 2;

prodotto := prodotto + 2;

Trasformo while in if-then-else

(3) if $y>0$ then C; WHILE

else skip;

(4) prodotto := prodotto + 2;

$y = y - 1$;

WHILE

(5) $y := y - 1$ WHILE

(6) WHILE

MEMORIA

y
σ_0 : 2
σ_1 : 2
σ_2 : 1
σ_3 : 0

(7) if $y>0$ then C; WHILE

else skip;

(8) probHo := probHo + 2;

$y := y-1$;

WHILE

(9) $y := y-1$ WHILE

(10) WHILE

(11) if $y>0$ then C; WHILE

else skip;

(12) skip $\longrightarrow \sigma_3$