

Associamo a ciascun programma P del linguaggio \mathcal{L} un numero, che indicheremo con $\#(P)$, in modo tale che il programma può essere integralmente ricostruito a partire da $\#(P)$.

Ordiniamo le variabili come segue

$Y, X_1, Z_1, X_2, Z_2, \dots$

e le etichette:

$A_1, B_1, C_1, D_1, E_1, A_2, B_2, C_2, \dots$

Scriviamo $\#(v)$, $\#(L)$ per la posizione di una variabile o di un'etichetta data nell'ordinamento appropriato.

Avremo quindi:

$$\#(X_2) = 4, \quad \#(Z_1) = \#(Z) = 3,$$

$$\#(E) = 5, \quad \#(B_2) = 7$$

Sia ora I un'istruzione (etichettata o non etichettata) del linguaggio S .

Scriviamo allora

$$\#(I) = \langle a, \langle b, c \rangle \rangle$$

dove:

1. Se I non è etichettata allora $a = 0$
Se I è etichettata L allora $a = \#(L)$
2. Se la variabile V compare in I
allora $c = \#(V) - 1$
3. Se l'enunciato in I è
 $V \leftarrow V$ oppure $V \leftarrow V + 1$ oppure $V \leftarrow V - 1$
allora, rispettivamente, $b = 0$ oppure
 $b = 1$ oppure $b = 2$.
4. Se l'enunciato in I è
 $\text{IF } V \neq 0 \text{ GOTO } L'$
allora $b = \#(L') + 2$

ESEMPI

Programma

[A] $X \leftarrow X+1$

I_1

IF $X \neq 0$ GOTO A

I_2

(calcola la funzione (calcolabile in S) non definita in nessun punto).

calcoliamo il numero di codice dell'istruzione I_1 abbiamo che:

$$\#(I_1) = \langle 1, \langle 1, 1 \rangle \rangle$$

↑
perché
l'etichetta A
ha posto 1

↑
perché
l'istruzione
è del tipo
 $X \leftarrow X+1$

←
perché la variabile che compare è X che si trova al secondo posto ed il valore di c è dato da:
 $c = \#(V) - 1$

effettuando il calcolo si ha:

$$\langle 1, 1 \rangle = 2^1 (2 \cdot 1 + 1) - 1 = 5$$

$$\langle 1, 5 \rangle = 2^1 (2 \cdot 5 + 1) - 1 = 21$$

quindi $\#(I_1) = 21$

$$\langle x, y \rangle = 2^x (2^y + 1) - 1$$

$I_2 : IF X \neq 0 \text{ GOTO } A$

$$\#(I_2) = \langle a, \langle b, c \rangle \rangle =$$

$$= \langle 0, \langle 3, 1 \rangle \rangle$$

perché
è istruzione
NON è
etichettata

essendo
l'istruzione
di "salto"
il suo numero
è $b = \#(L) + 2 = 1 + 2 = 3$
poiché $\#(A) = 1$

perché la variabile
che compare è X

effettuando il calcolo si trova che:

$$\#(I_2) = 46$$

poiché:

$$\langle 3, 1 \rangle = 2^3 (2 \cdot 1 + 1) - 1 = 8(2 + 1) - 1 = 23$$

$$\text{e } \langle 0, 23 \rangle = 2^0 (2 \cdot 23 + 1) - 1 = 46$$

Il numero del programma è quindi dato da:

$$\#(P) = [\#(I_1), \#(I_2)] - 1 = [21, 46] - 1 =$$

$$= 2^{21} \cdot 3^{46} - 1$$

Si osservi che per ogni numero dato q vi è un'unica istruzione I con $\#(I) = q$. Calcoliamo prima $l(q)$. Se $l(q) = 0$, I è non etichettata; in caso contrario I ha la $l(q)$ -esima etichetta della nostra lista. Per trovare la variabile che compare in I , calcoliamo $i = r(r(q)) + 1$ e localizziamo la i -esima variabile V nella nostra lista. L'enunciato in I sarà quindi

$V \leftarrow V$ se $l(r(q)) = 0$,

$V \leftarrow V + 1$ se $l(r(q)) = 1$,

$V \leftarrow V - 1$ se $l(r(q)) = 2$,

IF $V \neq 0$ GOTO L se $j = l(r(q)) - 2 > 0$

ed L è la j -esima etichetta nella nostra lista.

Infine, un programma P consista delle istruzioni I_1, I_2, \dots, I_k . Poniamo allora

$$\#(P) = [\#(I_1), \#(I_2), \dots, \#(I_k)] - 1..$$

$$\langle \langle a, \langle b, c \rangle \rangle \rangle = q$$

Calcoliamo il numero di ~~istruzioni~~ ^{codice} dell'istruzione non etichettata:

$$Y \leftarrow Y \quad (*)$$

Ricordando che

- Se I non è etichettata $a = 0$
- Se I è del tipo $V \leftarrow V$ $b = 0$
- Se la variabile nominata in I è Y allora $c = 0$

$$(c = \#(V) - 1 = \#(Y) - 1 = 1 - 1 = 0)$$

il numero associato all'istruzione (*) è $\langle 0, \langle 0, 0 \rangle \rangle = 0$

Ricorriamo:

Unica ambiguità codifiche di Gödel, gli zeri alla fine -

Ambiguità eliminabile:

Decretando che nessun programma può terminare con l'istruzione $Y \leftarrow Y$.

TEOREMA DELLA FERMATA

Per un y dato, sia ora P il programma tale che $\#(P) = y$ allora $\text{HALT}(x,y)$ è vero se $\Psi_P^{(1)}(x)$ è definito e falso se $\Psi_P^{(1)}(x)$ non è definito. In breve:

$\text{HALT}(x,y) \iff$ il programma numero y si fermerà, prima o poi, sull'ingresso x .

Teorema. $\text{HALT}(x,y)$ non è un predicato computabile.

Dimostrazione. Supponiamo che $\text{HALT}(x,y)$ sia computabile. Potremmo allora costruire il programma \textcircled{Q} :

[A] IF $\text{HALT}(X,X)$ GOTO A

(Naturalmente \textcircled{Q} deve essere la macro espansione di questo programma.)

Ovviamente \textcircled{Q} è stato costruito in modo tale che:

$$\Psi_{\textcircled{Q}}^{(1)}(x) = \begin{cases} \text{indefinito} & \text{se } \text{HALT}(x,x) \\ 0 & \text{se } \sim \text{HALT}(x,x). \end{cases}$$

Sia $\#(\textcircled{Q}) = y_0$. Usando allora la definizione del predicato HALT ,

$$\text{HALT}(x,y_0) \iff \sim \text{HALT}(x,x).$$

Poichè quest'equivalenza è vera per ogni x , possiamo porre $x = y_0$:

$$\text{HALT}(y_0,y_0) \iff \sim \text{HALT}(y_0,y_0).$$

Ma questa è una contraddizione.

Ed il teorema è dimostrato.

Questo teorema ci fornisce, intanto, un esempio di funzione che non è computabile mediante nessun programma del linguaggio S.

Ma possiamo dire di più:

Dato un programma di S ed un ingresso a tale programma, non esiste nessun algoritmo in grado di determinare se il programma dato si fermerà o no, prima o poi, sull'ingresso dato.

Sotto questa forma il risultato è chiamato la insolubilità del problema della fermata.

Infatti, se un tale algoritmo esistesse, potremmo usarlo per verificare la verità o falsità di $HALT(x,y)$ per x ed y dati, ottenendo prima il programma Q , con $\#(Q) = y$, e verificando quindi se Q , prima o poi, si ferma sull'ingresso x .

Chiediamoci adesso se il predicato
 $\text{HALT}(x, x)$ di una sola variabile
è calcolabile:

Supponiamo che lo sia. Allora come
nel caso precedente potremmo costruire
il programma P

[A] IF $\text{HALT}(x, x)$ GOTO A

e no, come prima, $y_0 = \#(P)$

Poniamoci adesso la domanda:

P si ferma sulla variabile
d'ingresso y_0 ?

Avremo:

$$\text{HALT}(y_0, y_0) \Leftrightarrow \neg \text{HALT}(y_0, y_0)$$

Lo mostra che in realtà la dimostrazione
precedente della non calcolabilità di
 $\text{HALT}(x, y)$ è, in nuce, la dimostrazione
della non calcolabilità di $\text{HALT}(x, x)$.

CONGETTURA DI GOLDBACH

OGNI NUMERO PARI ≥ 4 È LA
SOMMA DI DUE NUMERI PRIMI

È facile verificarlo per numeri piccoli

$$4 = 2 + 2$$

$$6 = 3 + 3$$

$$20 = 7 + 13$$

$$36 = 17 + 19$$

È possibile scrivere un programma
che verifica la congettura cercando
controesempi -

Il programma si ferma quando ha
trovato un n pari che non soddisfa
la congettura -

Collegamento col teorema delle fermat

Cosa potremmo dire se il teorema
della fermat non fosse vero?