

CAPITOLO 4

PROBLEMI CHE UN CALCOLATORE NON POTRA' MAI RISOLVERE

Problema 10. *Data una equazione diofantina con un qualunque numero di incognite ed a coefficienti interi individuare un processo che permetta di decidere con un numero finito di operazioni se l'equazione ammette soluzioni intere o meno (Hilbert 1900).*

1. Problemi a cui un calcolatore può fornire risposta: la nozione di problema decidibile.

Nel problema ora citato con l'espressione "equazione diofantina" si intende una equazione algebrica a coefficienti interi. Il problema, che riprenderemo nel prossimo paragrafo, consiste nell'individuare una regola che permetta di verificare, conoscendo i coefficienti dell'equazione, se esistono radici intere o meno. Tuttavia tale problema prima ancora di essere affrontato deve essere riformulato in maniera rigorosa. Infatti si dovrebbe dare una precisa definizione matematica di cosa si debba intendere per

"processo che permetta di decidere con un numero finito di operazioni"

nozione che ai tempi di Hilbert non era stata ancora data. D'altra parte, poiché nel capitolo precedente abbiamo definito la nozione di "funzione computabile" possiamo tentare di fornire una definizione di tale nozione utilizzando quella di funzione computabile. Per fare questo, ricordiamo che un sottoinsieme X di un insieme S può essere anche individuato dalla sua *funzione caratteristica* c_X che viene definita ponendo

$$c_X(x) = \begin{cases} 1 & \text{se } x \in X \text{ (cioè se } x \text{ verifica } P) \\ 0 & \text{se } x \notin X \text{ (cioè se } x \text{ non verifica } P) \end{cases}$$

Definizione 1.1. Un sottoinsieme X di N si dice *decidibile* se la sua funzione caratteristica è computabile. Più in generale, dato un intero positivo n chiamiamo *decidibile* una relazione n -aria se la sua funzione caratteristica è decidibile.

La nozione di insieme decidibile corrisponde all'idea che esista un programma (cioè un "processo che permetta di decidere") per stabilire, dato un qualunque elemento x , se x appartiene all'insieme in questione o meno. Infatti, dato l'input x ,

- se l'output è 1 allora possiamo dire che x appartiene ad X ,
- se l'output è 0, allora possiamo dire che x non appartiene ad X .

Allora se si accetta una tale definizione il problema di Hilbert può essere rigorosamente riformulato al modo seguente:

"L'insieme dei coefficienti delle equazioni diofantine che ammettono soluzioni intere è decidibile?"

Equivalentemente:

"esiste un programma che ricevendo in input i coefficienti di una equazione fornisce come output il valore 1 se la corrispondente equazione ammette soluzioni intere, il valore 0 altrimenti?"

Forniamo ora qualche esempio di insieme decidibile.

Esempio 1. L'insieme dei numeri pari è decidibile in quanto la sua funzione caratteristica è computata dal programma :

read(x); i := 0

1: *i := i+2 ; IF i < x THEN GOTO 1;*

IF i = x THEN y:=1 ;

IF i > x THEN y:= 0

write(y).

Tale programma incrementa la variabile i di due unità fino a quando i si mantiene minore di x . Se in tale modo di procedere i raggiunge il valore x allora x è pari, altrimenti x è dispari.

Esempio 2. La relazione " x è congruo ad y modulo 3" è decidibile perché la sua funzione caratteristica è computata dal programma

```

read(x); read(y); i := x
1:  i := i+3; IF i < y THEN GOTO 1;
    IF i = y THEN z:=1;
    IF i > x THEN z:=0
write(y).

```

A volte invece di dire che un insieme è decidibile diciamo che un problema o una proprietà è decidibile.

Definizione 1.2. Una proprietà $\alpha(x_1, \dots, x_n)$ viene detta *decidibile* se la sua estensione

$$X = \{(x_1, \dots, x_n) \in \mathbb{N}^n : \alpha(x_1, \dots, x_n) \text{ è vera}\}$$

è decidibile. In tale caso possiamo anche dire che *il problema* di stabilire per i numeri x_1, \dots, x_n se $\alpha(x_1, \dots, x_n)$ è vera o meno è decidibile.

Ad esempio, invece di dire che l'insieme dei numeri pari è decidibile possiamo dire che la proprietà di essere pari è decidibile oppure che il problema di stabilire se un numero è pari o meno è decidibile. Dato un intero m , invece di dire che l'insieme $\{(x, y) \in \mathbb{N} \times \mathbb{N} : x \text{ è congruo ad } y \text{ modulo } m\}$ è decidibile, possiamo dire che la relazione di congruenza è decidibile o che il problema "verificare se una coppia di numeri sono congrui modulo 5" è decidibile.

Come mostra la seguente proposizione tutti gli insiemi finiti sono decidibili.

Proposizione 1.3. L'insieme vuoto ed \mathbb{N} sono decidibili. Ogni insieme finito è decidibile.

Dim. L'insieme vuoto è decidibile in quanto la sua funzione caratteristica è computata dal programma

```

read(x);
y:=0;
write(y).

```

\mathbb{N} è decidibile in quanto la sua funzione caratteristica è computata dal programma

```

read(x);
y:=1;
write(y).

```

Sia X un insieme finito costituito dagli elementi n_1, \dots, n_h , allora il programma

```

read(x); y:=0
IF x = n1 THEN y := y+1;
.....
IF x = nh THEN y := y+1;
write(y).

```

computa la funzione caratteristica di X .

Esercizio. Scrivere un programma che calcoli la funzione caratteristica di $\{2, 5, 4\}$.

Teorema 1.4. Il complemento di un insieme decidibile è un insieme decidibile. L'unione e l'intersezione di due insiemi decidibili è un insieme decidibile. Pertanto la classe $Dec(\mathbb{N})$ delle parti decidibili di \mathbb{N} costituisce un'algebra di Boole numerabile, più precisamente una sotto-algebra $(Dec(\mathbb{N}), \cup, \cap, -, \emptyset, \mathbb{N})$ dell'algebra di Boole $(P(\mathbb{N}), \cup, \cap, -, \emptyset, \mathbb{N})$.

Dim. Supponiamo che X sia decidibile e che c_X sia la funzione caratteristica di X . Allora $1 - c_X$ è la funzione caratteristica del complemento $-X$ di X . D'altra parte se la prima funzione è computabile lo è anche la seconda e ciò prova che $-X$ è decidibile. Un altro modo di provare la stessa cosa è supporre che

```

π read(x); . . . ; write(y)

```

sia un programma per decidere se un elemento x appartiene o meno ad X . Allora inserendo prima di $write(y)$ l'istruzione $y:=1-y$ si ottiene un programma per decidere se un elemento x appartiene o meno a $-X$.

Siano c_1 e c_2 le funzioni caratteristiche degli insiemi decidibili X_1 e X_2 , allora è evidente che la funzione c definita da $c(x)=\max\{c_1(x),c_2(x)\}$ è la funzione caratteristica di $X_1\cup X_2$. Poiché tale funzione è computabile, $X_1\cup X_2$ è decidibile. D'altra parte ponendo $c(x)=\min\{c_1(x),c_2(x)\}$ si ottiene la funzione caratteristica della intersezione ed è evidente che tale funzione risulta essere computabile. In termini di programmi e riferendoci alla intersezione, supponiamo che

π_1 $read(x)$; P_1 ; $write(y_1)$ sia un programma per X_1

π_2 $read(x)$; P_2 ; $write(y_2)$ sia un programma per X_2 ,

in cui si sono usate due variabili diverse y_1 e y_2 per evitare confusione. Allora il programma che si ottiene

- ponendo π_1 e π_2 uno dopo l'altro

- cancellando un $read(x)$, $write(y_1)$, $write(y_2)$

- ponendo alla fine l'istruzione

IF $y_1=1$ AND $y_2=1$ THEN $y:=1$ ELSE $y:=0$

- ponendo ancora $write(y)$

computa la funzione caratteristica dell'intersezione.

Nel caso dell'unione si considera invece l'istruzione

IF $y_1=1$ OR $y_2=1$ THEN $y:=1$ ELSE $y:=0$.

Possiamo anche esprimere tale teorema in termini logici. Ricordiamo che date due proprietà α e β , in logica matematica se ne indica con $\alpha\wedge\beta$ la congiunzione, con $\alpha\vee\beta$ la disgiunzione mentre $\neg\alpha$ indica la negazione di α .

Teorema 1.5. Se α e β sono proprietà decidibili allora anche $\alpha\wedge\beta$, $\alpha\vee\beta$ e $\neg\alpha$ sono proprietà decidibili.

Esercizio. Chiamiamo *cofinito* un sottoinsieme di N il cui complemento sia finito. Dimostrare che gli insiemi cofiniti sono decidibili.

Nota: La nozione di "problema indecidibile" può dare luogo a qualche confusione. Infatti ci si deve sempre riferire a problemi $P(n_1,\dots,n_t)$ che dipendano da parametri n_1,\dots,n_t e quando si afferma che il problema è decidibile ci si riferisce all'esistenza di un programma che, ricevendo in input tali parametri n_1,\dots,n_t ci dica se $P(n_1,\dots,n_t)$ risulta vera oppure no. D'altra parte, come abbiamo già detto, la decidibilità di $P(n_1,\dots,n_t)$ significa la decidibilità dell'insieme $\{(n_1,\dots,n_t) : P(n_1,\dots,n_t) \text{ sia vera}\}$. Consideriamo invece un problema di matematica P che, per quanto complicato, non dipenda da parametri. Ad esempio consideriamo il famoso problema di Fermat che consiste nello stabilire se l'asserzione $\forall m\exists x\exists y\exists z(x^m+y^m=z^m)$ sia vera oppure no. Per tali tipi di problema non ha senso porsi questioni di decidibilità. Ha invece senso chiedersi se sia decidibile il problema $\exists x\exists y\exists z(x^m+y^m=z^m)$ che dipende dal parametro m , cioè se sia decidibile l'insieme

$$\{m\in N : \text{è vero che } \exists x\exists y\exists z(x^m+y^m=z^m)\}.$$

2. Problemi che un calcolatore può risolvere a metà: la nozione di problema semi-decidibile.

Nella teoria degli insiemi si dice *enumerabile* un sottoinsieme S che abbia potenza minore del numerabile, cioè un insieme S che sia vuoto, finito o numerabile. Equivalentemente S è enumerabile se è vuoto o è l'insieme dei valori di una successione (non necessariamente iniettiva). Infatti se S è finito, ad esempio $S = \{a_1,\dots,a_p\}$, allora S è l'insieme dei valori della funzione $f:N\rightarrow S$ definita ponendo $f(1)=a_1,\dots,f(p)=a_p$, $f(m)=a_p$ per ogni $m>p$. Se invece S è infinito allora è equipotente ad N e quindi per definizione esiste $f:N\rightarrow S$ biettiva.

In ambito costruttivo tale nozione di enumerabilità perde di interesse in quanto, poiché ci si riferisce sempre a sottoinsiemi di N , tutti gli insiemi presi in considerazione sono enumerabili. Risulta invece interessante parlare di "effettiva" enumerabilità per un sottoinsieme S di N , concetto che si ottiene richiedendo che la successione enumerante S sia effettivamente computabile.

Definizione 2.1. Un sottoinsieme S di N si dice *effettivamente enumerabile* se è vuoto o se è il codominio di una funzione totale e computabile f . In tale caso diremo che f è una *funzione enumerante* S .

Più in generale, una relazione p -aria si dice *effettivamente numerabile* se è la relazione vuota oppure se è il codominio di una funzione totale computabile $f: N \rightarrow N^p$.

Si può vedere un insieme effettivamente enumerabile come un insieme i cui elementi si possono stampare, uno alla volta, in modo effettivo. Infatti supponiamo che f sia calcolata da un opportuno programma, allora il programma

```
Function f . . . ;
n := 1 ;
1 write(f(n)) ;
  n := n+1 ; GOTO 1
```

stampa appunto tutti (e soli) gli elementi di S .

Nota. Per evitare di scrivere programmi troppo lunghi, nel seguito supporremo che il linguaggio di programmazione permetta anche la definizione di funzioni (cosa che è vera ormai per tutti i linguaggi). Ciò significa che se una funzione è computabile allora è possibile assegnare ad essa un nome f in modo che in un qualunque punto del programma il simbolo $f(n)$ denoterà il valore di f in n . Per indicare che in un programma è stata definita una funzione f scriveremo in breve

Function f . . . ;

Proposizione 2.2. L'insieme dei quadrati perfetti, l'insieme dei numeri pari, l'insieme dei numeri dispari costituiscono tutti esempi di insiemi effettivamente enumerabili.

Dim. Infatti tali insiemi sono i codomini rispettivamente delle funzioni x^2 , $2x$, $2x+1$.

Tutti gli insiemi decidibili sono effettivamente enumerabili (come mostreremo nella prossima proposizione) mentre esistono insiemi effettivamente enumerabili che non sono decidibili (come proveremo nei prossimi paragrafi).

Teorema 2.3. Sia S un insieme allora

S decidibile $\Leftrightarrow S$ e $-S$ sono effettivamente enumerabili.

Dim. Sia S un insieme decidibile, allora nel caso $S = \emptyset$ per la definizione di effettiva enumerabilità, S è anche effettivamente enumerabile. Supponiamo che $S \neq \emptyset$, allora esiste $b \in S$ e possiamo definire la funzione f ponendo

$$f(n) = \begin{cases} b & \text{se } n \notin S \\ n & \text{se } n \in S. \end{cases}$$

Tale funzione ha come codominio S ed è evidente che tale funzione è computabile. D'altra parte, se c è la funzione caratteristica di S , la funzione f può essere computata dal programma

```
Function c ....;
read(n) ;
IF c(n) = 0 THEN y := b ELSE y := n;
write(y)
```

e ciò prova che S è effettivamente enumerabile. D'altra parte se S è decidibile anche $-S$ decidibile e quindi, per quanto ora provato, anche $-S$ è effettivamente enumerabile. Viceversa, supponiamo che S sia enumerato dalla funzione computabile f e $-S$ dalla funzione computabile g . Allora la funzione

{

$$c(x) = \begin{cases} 1 & \text{se esiste } n \text{ tale che } f(n) = x \\ 0 & \text{se esiste } n \text{ tale che } g(n) = x \end{cases}$$

coincide con la funzione caratteristica di S . Tale funzione può essere computata dal seguente programma

```
Function f .... ; Function g ....;
read(x) ; n:=1 ; y:=2 ;
1  IF x = f(n) THEN y:=1 ;
   IF x = g(n) THEN y:=0 ;
   n := n+1 ;
   IF y:=2 THEN GOTO 1
write(y).
```

pertanto S è decidibile. Come al solito abbiamo denotato con Function f e Function g i programmi con cui è possibile computare le funzioni f e g . Naturalmente abbiamo potuto fare questo in base all'ipotesi per cui f e g sono computabili.

Ad esempio, per provare che l'insieme P dei numeri pari è decidibile è sufficiente osservare che P è effettivamente enumerabile in quanto codominio della funzione $f(x) = 2 \cdot x$. Inoltre il suo complemento, l'insieme D dei numeri dispari, è effettivamente enumerabile in quanto codominio della funzione $g(x) = 2 \cdot x + 1$. In definitiva un programma per verificare se x è pari oppure no è il seguente

```
read(x) ; n:=1 ; y:=2 ;
1  IF x = 2n THEN y:=1 ;
   IF x = 2n+1 THEN y:=0 ;
   n := n+1 ;
   IF y:=2 THEN GOTO 1
write(y).
```

Abbiamo chiamato effettivamente enumerabile ogni insieme che sia codominio di una funzione totale computabile e di una variabile. Un esempio è l'insieme $\{p^2 : p \in \mathbb{N}\}$ dei quadrati perfetti. Esaminiamo ora che cosa si ottiene nel caso di insiemi che siano il condominio di funzioni di più variabili. Ad esempio consideriamo l'insieme $\{p^2 + q^2 : p \in \mathbb{N} \text{ e } q \in \mathbb{N}\}$ dei numeri che sono somma di due quadrati perfetti (ad esempio, 1, 4, 5, 13, ...): esiste un programma capace di stamparne uno dopo l'altro tutti gli elementi di tale insieme? La risposta è positiva poiché il condominio di una funzione computabile di più variabili è effettivamente enumerabile.

Proposizione 2.4. Sia $p \in \mathbb{N}$ e sia $f : \mathbb{N}^p \rightarrow \mathbb{N}$ una funzione totale computabile di p variabili, allora il condominio di f è un insieme effettivamente enumerabile.

Dim. Per semplicità consideriamo il caso $p = 2$. In un certo senso si tratta di *stampare* uno dopo l'altro gli elementi di $Cod(f)$ e per fare questo possiamo esplorare, un passo alla volta, tutti i valori della seguente tabella infinita secondo qualche strategia che non escluda nessuna coppia

(1,1), (1,2), (**1,3**), (1,4), ...
 (2,1), (2,2), (**2,3**), (2,4), ...
 (**3,1**), (**3,2**), (**3,3**), (3,4), ...
 ...

Contemporaneamente, ogni volta che si raggiunge una coppia (i, j) si deve calcolare il valore $f(i, j)$. Il problema si riconduce quindi al problema della numerabilità del prodotto cartesiano di due insiemi già affrontato da Cantor nella sua teoria degli insiemi. Un informatico affronterebbe questo problema con un paio di cicli *nested* cioè uno dentro l'altro. Infatti se considerassimo il programma

```
For i:=1 to 3
Print f(i,3)
Print f(3,i)
```

avremmo un modo per stampare tutti i valori che si collocano sui due lati del rettangolo di vertice (3,3) (si vedano le coppie scritte in grassetto). Basta allora porre al posto di 3 una variabile n ed inserire il tutto in un ciclo infinito in cui n viene incrementato di una unità.

```
n = 0
1: n := n+1
  For i :=1 to n
    For j := 1 to n
      Print f(i,j)
    goto 1
```

Un modo più “matematico” di affrontare la questione è trovare una funzione totale computabile e suriettiva $g : N \rightarrow N^2$. Infatti in tale caso il problema è risolto in quanto basta comporre g con f per ottenere una funzione totale computabile di una sola variabile con lo stesso condominio di f

$$f \circ g : N \rightarrow N^2 \rightarrow N$$

Ad esempio possiamo definire g ponendo $g(n) = (p, q)$ in modo che 2^p e 3^q siano i primi due fattori della scomposizione in fattori di n . In altri termini poniamo

$$p = \text{Max}\{i \in N : 2^i \text{ sia un fattore di } n\} \text{ e } q = \text{Max}\{i \in N : 3^i \text{ sia un fattore di } n\}.$$

Allora la funzione $g \circ f$ è una funzione totale computabile di una variabile con lo stesso codominio di g . Ciò mostra che il codominio di g è effettivamente enumerabile. \square

Da tale proposizione consegue che l'insieme $\{p^2 + q^2 : p \in N \text{ e } q \in N\}$ è effettivamente enumerabile. Per stampare uno dopo l'altro tutti i valori di f possiamo semplicemente enumerare uno dopo l'altro gli elementi di N^2 e poi man mano stampare i valori di f .

Abbiamo definito un insieme effettivamente enumerabile come il condominio di una funzione computabile. Vogliamo dimostrare che può anche essere definito come il dominio di una funzione parziale ricorsiva. Per fare questo abbiamo bisogno della nozione di *funzione vuota* che viene definita come il sottoinsieme vuoto di $N \times N$. Tale funzione può essere vista come la funzione calcolata da un programma che, qualunque sia l'input, cade sempre in un loop. Ad esempio il programma

```
read(x);
1: IF 1 = 1 THEN GOTO 1;
write(x)
```

computa la funzione vuota in quanto, essendo la condizione $1 = 1$ è sempre vera, allora l'istruzione 1 viene ripetuta infinite volte e non si arriva mai a stampare il valore di x .

Ricordiamo che, come già indicato nel capitolo 3, diciamo che in un dato un programma π è stato inserito un *contatore* se si considera un altro programma π^c che si ottiene:

- fissando una nuova variabile c che inizialmente è posta uguale a zero tramite l'istruzione $c := 0$
- antepoendo ad ogni istruzione di π l'assegnazione $c := c + 1$.

Diciamo che un programma si ferma in n passi se nel momento in cui il contatore si ferma c è uguale ad n .

Teorema 2.5. Sia S un insieme, allora le seguenti asserzioni sono equivalenti.

- i) S è effettivamente enumerabile.
- ii) S è il dominio di una funzione computabile f , cioè esiste un programma π tale che

$$S = \{x \in \mathbb{N} : \pi \text{ converge se l'input è } x\}.$$

Dim. i) \Rightarrow ii) Sia S effettivamente enumerabile, allora se $S = \emptyset$ la proposizione è ovvia poiché il programma π che computa la funzione vuota verifica l'equivalenza

$$x \in \emptyset \Leftrightarrow \pi \text{ converge in } x \text{ e fornisce } 1 \text{ come output.}$$

Infatti, per ogni modo di fissare x , entrambi i lati di tale equivalenza sono falsi. Supponiamo $S \neq \emptyset$ e che S sia effettivamente enumerata dalla funzione computabile f . Allora possiamo definire la funzione g definita dal seguente algoritmo

- dato l'input x si producano gli elementi $f(1), f(2), \dots$ fino a quando non si ottiene x , se ciò accade allora si pone come output 1. È immediato che g è computabile e che ha come dominio S . Notiamo esplicitamente che g può essere computata dal programma

```

Function f .... ;
read(x) ; n:=0 ;
1:  n := n+1 ;
   IF x = f(n) THEN GOTO 2 ELSE goto 1
2:  write(1)

```

e che quindi S è l'insieme degli interi x in cui tale programma converge.

ii) \Rightarrow i) Sia S il dominio della funzione computabile g , sia g computata dal programma π . Se $S = \emptyset$ allora S è effettivamente enumerabile per definizione. Se $S \neq \emptyset$ sia b un elemento di S e definiamo $h(x,n)$ tramite:

$$h(x,n) = \begin{cases} x & \text{se } \pi \text{ con input } x \text{ si ferma in meno di } n \text{ passi;} \\ b & \text{altrimenti.} \end{cases}$$

Tale funzione risulta totale e computabile. Infatti basta considerare il programma π^* che si ottiene da π

- aggiungendo a π inizialmente due istruzioni del tipo INPUT x , INPUT n
- aggiungendo a π un contatore c
- aggiungendo opportunamente una istruzione del tipo IF $c > n$ THEN fermati e stampa b
- sostituendo l'istruzione di output di π con l'istruzione "write(x)".

Poiché il codominio di h coincide con S , possiamo concludere che S è effettivamente enumerabile. □

La nozione di effettiva enumerabilità coincide con quella di semi-decidibilità.

Definizione 2.6. Un insieme S si dice *semi-decidibile*, se la sua funzione caratteristica è computabile per metà, cioè se esiste un programma π tale che

- $x \in S \Rightarrow \pi$ converge in x e fornisce output 1
- $x \notin S \Rightarrow \pi$ diverge in x

La dimostrazione del seguente teorema è immediata.

Proposizione 2.7. Un insieme è effettivamente enumerabile se e solo se è semi-decidibile.

3. Forma logica di un problema, decidibilità, semi-decidibilità.

Spesso è possibile capire se un problema è decidibile o semi-decidibile osservando l'espressione logica che lo definisce. Nel seguito ci riferiamo al linguaggio del primo ordine dell'aritmetica. Una definizione precisa di tale linguaggio verrà data nel Capitolo 6. Qui ci limitiamo a dire, un po' approssimativamente, che è l'insieme delle espressioni che si possono scrivere usando le operazioni aritmetiche di somma e prodotto, le variabili, i numerali, i connettivi logici ed i quantificatori. Useremo anche i simboli di relazioni binarie $=$ e \leq . In tale linguaggio possono essere scritti, ad esempio, tutti i polinomi e tutte le equazioni algebriche.

La seguente proposizione mostra che se una proprietà è definibile da una espressione in cui non sono presenti i quantificatori allora è decidibile.

Proposizione 3.1. Ogni proprietà aritmetica esprimibile da una formula atomica o, più in generale, da una formula priva di quantificatori è decidibile.

Dim. Le formule atomiche sono del tipo $p = q$, oppure $p \geq q$ oppure $p > q$ con p e q polinomi sugli interi. Consideriamo ad esempio l'insieme $X = \{(x_1, \dots, x_n) \in \mathbb{N}^n : p(x_1, \dots, x_n) = q(x_1, \dots, x_n)\}$ definito dal primo tipo di formule. Allora è evidente che X è decidibile perché un algoritmo per decidere circa l'appartenenza di (x_1, \dots, x_n) ad X consiste semplicemente nell'eseguire i calcoli presenti in p e q e di andare a verificare che il valore $p(x_1, \dots, x_n)$ sia uguale al valore $q(x_1, \dots, x_n)$. La stessa cosa si può dire se si considerano disequazioni al posto di equazioni.

Se α è una espressione priva di quantificatori, allora α è ottenuta da formule atomiche tramite i connettivi logici \wedge , \vee , \neg . Allora la relazione rappresentata da α è ottenuta tramite le operazioni insiemistiche di unione, intersezione e complemento a partire da relazioni atomiche (e quindi decidibili). Pertanto, tenendo conto del fatto che l'unione, l'intersezione di due relazioni decidibili è una relazione decidibile e dal fatto che il complemento di una relazione decidibile è ancora decidibile, possiamo concludere che la proprietà rappresentata da α è decidibile.

Ad esempio è decidibile le proprietà $x^2 + y^2 = z^2$ cioè è decidibile la relazione $\{(x, y, z) \in \mathbb{N}^3 : x^2 + y^2 = z^2\}$. Tuttavia è anche decidibile la proprietà $(x^2 + y^2 = z^2) \wedge (x \leq y) \wedge \neg(x \leq z)$.

E' interessante ora vedere che cosa succede quando invece vengono introdotti i quantificatori.

Teorema 3.2. Un insieme è semi-decidibile se e solo se è definibile da una asserzione del tipo

$$\exists n R(n, x)$$

con R relazione decidibile.

Dim. Sia $X = \{x : \text{esiste } n \in \mathbb{N} \text{ tale che } (n, x) \in R\}$ con R decidibile e consideriamo il seguente programma

Read(x),

$y := 1$

$n = 1$;

1. $n := n + 1$

IF NOT $R(n, x)$ THEN GOTO 1 ELSE GOTO 2

2. *Print*(1)

Allora è evidente che tale programma converge ad 1 se e solo se $x \in X$ e quindi che X è semi-decidibile.

Viceversa, supponiamo che X sia semi-decidibile. Nel caso in cui $X = \emptyset$ allora X è definibile, ad esempio, dalla asserzione $\exists n (x \cdot n = x \cdot n + 1)$ che risulta sempre falsa. Nel caso $X \neq \emptyset$, allora esiste una funzione computabile f il cui codominio coincide con X . Posto $R = \{(n, x) : f(n) = x\}$ è evidente che R è decidibile e che

$$X = \{x : \text{esiste } n \in \mathbb{N} \text{ tale che } f(n) = x\} = \{x : \text{esiste } n \in \mathbb{N} \text{ tale che } (n, x) \in R\}.$$

Non è difficile estendere tale teorema anche al caso di più quantificatori esistenziali.

Teorema 3.3. Un insieme è semi-decidibile se e solo se è definibile da una asserzione del tipo

$$\exists n_1 \dots \exists n_t R(x, n_1, \dots, n_t)$$

con R relazione decidibile.

Il termini di logica matematica, il teorema ora provato mette in rilievo che se un problema è ottenuto da un problema decidibile tramite quantificatori esistenziali allora il problema è semi-decidibile.

Esempio: Consideriamo il problema di verificare se un numero è il quadrato di un numero primo, problema che corrisponde all'insieme $\{m : \exists n \in \mathbb{N} \text{ tale che } n \text{ è primo e } m = n^2\}$. Osserviamo in proposito che la proprietà “ n è primo” è decidibile, che la relazione “ $m = n^2$ ” è decidibile e che quindi la congiunzione di tali proprietà “ n è primo ed $m = n^2$ ” è decidibile. Allora tale problema è semi-decidibile poiché consiste in una quantificazione esistenziale di un problema decidibile.

Esempio: Il problema di verificare per quali valori di n l'equazione $x^n + y^n = z^n$ ammette soluzioni intere, problema che corrisponde alla espressione $\exists x \exists y \exists z (x^n + y^n = z^n)$, è semi-decidibile perché la relazione $R = \{(x, y, z, n) : x^n + y^n = z^n\}$ è decidibile.

E' interessante osservare invece che se un problema è descrivibile, a partire da un problema decidibile, tramite una quantificazione esistenziale “limitata” allora il problema resta decidibile. La stessa proprietà è verificata dalla quantificazione universale limitata.

Teorema 3.4. Se $R(x, y)$ è una relazione decidibile ed f una funzione computabile, allora le proprietà $\exists n \leq f(x) R(n, x)$ e $\forall n \leq f(x) R(n, x)$ sono ancora decidibili.

Dim. Ad esempio la funzione caratteristica di $X = \{x \in \mathbb{N} : \exists n \leq f(x) R(n, x)\}$ è computata dal programma (che converge per qualunque input x)

```

Read(x)
y = 0
FOR n:= 1 TO f(x)
IF R(n,x) THEN c = 1
Print(y).

```

Nel caso in cui X sia definito dalla proprietà $\forall n \leq f(x) R(n, x)$ allora il suo complemento $\neg X$ sarà definito dalla negata $\neg(\forall n \leq f(x) R(n, x))$ che equivale a $\exists n \leq f(x) \neg R(n, x)$. Poiché $\neg R(n, y)$ è decidibile, per quanto ora dimostrato $\neg X$ risulta decidibile. Ciò prova che X è decidibile.

Se vogliamo scrivere in modo più esplicito il programma per verificare $\forall n \leq f(x) R(n, x)$ dobbiamo allora:

- scrivere il programma per $\exists n \leq f(x) \neg R(n, x)$ (che va bene per $\neg(\forall n \leq f(x) R(n, x))$)
- poi porre al posto di y il valore $1-y$.

Si otterrà il programma:

```

Read(x)
y = 0
FOR n:= 1 TO f(x)
IF  $\neg R(n,x)$  THEN y = 1
y := 1-y
Print(y).

```

Più in generale se $R(n_1, \dots, n_t, x)$ una relazione decidibile Q_1, \dots, Q_t simboli per un quantificatore esistenziale o universale, f_1, \dots, f_t funzioni computabili, allora la proprietà

$$Q_1 n_1 \leq f_1(x) (Q_2(x) \dots Q_t n_t \leq f_t(x) R(n_1, \dots, n_t, x))$$

è ancora decidibile. Per la rimanente parte del teorema si procede per induzione sul numero t . Infatti il teorema è stato ora dimostrato per $t = 1$. Supposto vero il teorema per $t-1$, abbiamo che per ipotesi di induzione la formula

$$Q_1 n_1 \leq f_1(x) Q_2 n_2 \leq f_2(x) \dots Q_{t-1} n_{t-1} \leq f_{t-1}(x) R(n_1, \dots, n_{t-1}, n_t, x)$$

è decidibile. Poi si procede in modo ovvio.

Esempio. Abbiamo già provato che l'insieme dei numeri pari è decidibile mostrando un programma per la sua funzione caratteristica. Tuttavia possiamo asserire che P è decidibile anche perché è descrivibile dalla formula $\exists n \leq x (2n = x)$ che è una quantificazione limitata di una proprietà decidibile.

Esempio. Consideriamo il problema di dire se un numero è divisibile per 5. Tale problema viene descritto dalla formula $\exists n \leq x (5 \cdot n = x)$ che si ottiene tramite una quantificazione limitata di un problema decidibile $5 \cdot n = x$. Pertanto il problema è decidibile.

Esempio. Torniamo al problema di verificare se un numero è il quadrato di un numero primo, problema che abbiamo osservato corrispondere all'insieme $\{m : \exists n \in \mathbb{N} \text{ tale che } n \text{ è primo e } m = n^2\}$. Allora tale problema è decidibile poiché corrisponde anche all'insieme $\{m : \exists n \leq m \text{ tale che } n \text{ è primo e } m = n^2\}$.

In definitiva:

mentre al quantificatore esistenziale corrisponde un ciclo condizionato del tipo WHILE (che può andare in loop), al quantificatore esistenziale limitato corrisponde un ciclo limitato del tipo FOR ..TO..

Esempio. Il problema di decidere se un numero x è primo si esprime tramite la formula $\exists n \exists m (n \neq 1 \wedge m \neq 1 \wedge (n \cdot m = x))$. Tuttavia i numeri n ed m di cui si cerca l'esistenza sono sicuramente minori di x . Quindi il problema può essere espresso tramite quantificazione limitata $\exists n \leq x (\exists m \leq x (n \neq 1 \wedge m \neq 1 \wedge (n \cdot m = x)))$. In conclusione il problema è decidibile.

4. Ridurre un problema ad un altro

Spesso capita che sia possibile ridurre un problema ad un altro. Questo fatto può essere formalizzato al modo seguente.

Definizione 4.1. Dati due insiemi X ed Y diciamo che X è *riducibile ad* Y e scriviamo $X \leq_r Y$ se esiste una funzione totale computabile h tale che, per ogni $x \in \mathbb{N}$,

$$x \in X \Leftrightarrow h(x) \in Y.$$

Se X è riducibile ad Y allora ogni processo di decisione circa l'appartenenza o meno dei numeri interi ad Y si può tradurre in un processo di decisione per X . Infatti per decidere se x appartiene ad X è sufficiente calcolare il numero $h(x)$ e poi andare a vedere se tale numero appartiene ad Y .

Proposizione 4.2. X è riducibile ad Y se e solo se

$$c_X(x) = c_Y(h(x)).$$

Dim. Supponiamo che $X \leq_r Y$ e che $c_X(x) = 1$. Allora $x \in X$ e quindi $h(x) \in Y$. Ne segue che $c_Y(h(x)) = 1$. Se invece $c_X(x) = 0$ allora $x \notin X$ e quindi $h(x) \notin Y$. Ne segue che $c_Y(h(x)) = 0$.

Il viceversa si prova in modo simile.

Pertanto possiamo anche dire che X è riducibile ad Y se esiste una funzione totale computabile h tale che $c_X = c_Y \circ h$. In un certo senso la relazione $X \leq_r Y$ significa che X è "meno decidibile" di Y . Vale infatti la seguente proposizione.

Proposizione 4.3. Siano X ed Y due insiemi tali che $X \leq_r Y$. Allora

- i) Y decidibile $\Rightarrow X$ decidibile.
- ii) Y semi-decidibile $\Rightarrow X$ semi-decidibile.

Dim. Per ipotesi sappiamo che esiste una funzione computabile h totale ed iniettiva tale che $c_X = c_Y \circ h$. Ne segue che se Y è decidibile, allora c_X , poiché si ottiene come composizione di due funzioni computabili, è computabile. Pertanto X è decidibile. Equivalentemente possiamo dire che, dato un elemento x , per decidere se x appartiene ad X oppure no dobbiamo calcolare $h(x)$ e verificare se $h(x)$ appartiene ad Y oppure no.

Sia Y semi-decidibile, allora sappiamo che esiste un programma π il quale nel caso in cui $y \in Y$, converge e fornisce come output 1 mentre altrimenti diverge. Consideriamo l'algoritmo π^* tale che, dato l'input x , calcola $h(x)$ e poi applica l'algoritmo π a partire dal valore $y = h(x)$. Poiché $x \in X$ se e solo se $h(x) \in Y$, π^* è tale che se $x \in X$ allora π^* converge e fornisce come output 1. Ciò prova che X è semi-decidibile. \square

Esempio. Sia P l'insieme dei numeri pari e supponiamo di aver dimostrato che P è decidibile, allora l'insieme D il cui quadrato è pari è decidibile in quanto

$$x \in D \Leftrightarrow x^2 \in P,$$

cioè $D \leq_r P$ tramite la funzione $h(x) = x^2$.

Per esaminare alcune proprietà della relazione \leq_r , ricordo che una relazione binaria \leq in un insieme S è chiamata *relazione di pre-ordine*, se è riflessiva e transitiva, cioè se:

$$x \leq x \quad \text{e} \quad x \leq y, y \leq z \Rightarrow x \leq z.$$

Non si richiede invece la proprietà anti-simmetrica

$$x \leq y \quad \text{e} \quad y \leq x \Rightarrow x = y$$

come viene fatto per le relazioni d'ordine.

Proposizione 4.4. La relazione \leq_r è una relazione di pre-ordine (ma non è una relazione d'ordine).

Dim. Per provare che $X \leq_r X$ è sufficiente porre h uguale all'applicazione identica. Supponiamo che $X \leq_r Y$ tramite la funzione computabile h e che $Y \leq_r Z$ tramite la funzione computabile k . Allora poiché $c_X(x) = c_Y(h(x))$ e $c_Y(y) = c_Z(k(y))$, posto $y = h(x)$, abbiamo che $c_X(x) = c_Z(k(h(x)))$, cioè, tenendo conto del fatto che $k \circ h$ è computabile, che $X \leq_r Z$.

Per mostrare che \leq non è una relazione d'ordine, supponiamo che P sia l'insieme dei numeri pari e che D sia l'insieme dei numeri dispari. Allora posto $h(x) = x-1$ risulta che $x \in P$ se e solo se $h(x) \in D$ e quindi $P \leq_r D$. Posto $k(y) = y+1$, risulta che $y \in D$ se e solo se $k(y) \in P$ e quindi $D \leq_r P$ (pur essendo $P \neq D$). \square

Se (S, \leq) è un a relazione di pre-ordine allora ponendo $x \equiv y$ se e solo se $x \leq y$ e $y \leq x$ si ottiene una relazione di equivalenza. Inoltre nel quoziente S/\equiv è possibile definire una relazione d'ordine ponendo

$$[x] \leq [y] \Leftrightarrow x \leq y.$$

In particolare la relazione di riducibilità determina una relazione di equivalenza.

Concludiamo questo paragrafo osservando che, come al solito, la nozione di riducibilità può essere estesa a sottoinsiemi di insiemi che non coincidono necessariamente con N . Ciò purché esista una opportuna codifica in N di tali insiemi. Ad esempio possiamo dire che un sottoinsieme X di N è riducibile ad un sottoinsieme Y di $N \times N$ se, detto $Cod(Y)$ l'insieme dei numeri di codice di Y risulta che X è riducibile a $Cod(Y)$.

5. Problemi che un calcolatore non potrà mai risolvere: il problema della fermata

L'esistenza di infinite funzioni non computabili, di infiniti insiemi non decidibili e di infiniti insiemi non effettivamente enumerabili segue da semplici considerazioni riguardanti le cardinalità. Nel seguito ci riferiremo a funzioni di N in N che possono anche essere parziali.

Teorema 5.1. L'insieme $Comp(N)$ delle funzioni computabili di N in N è numerabile, pertanto esistono funzioni di N in N non computabili (più precisamente la classe delle funzioni non computabili ha la potenza del continuo).

Dim. Già abbiamo visto come si possa codificare la classe dei programmi capaci di computare una funzione cioè come sia possibile associare ad ogni intero n un programma $\pi(n)$ in modo da enumerare tutti i programmi. Tale codifica si estende all'insieme delle funzioni computabili associando ad ogni intero n la funzione f_n computata dal programma π_n . Possiamo allora considerare la funzione che associa ad ogni funzione computabile il minimo tra gli indici dei programmi capaci di computare f . Tale funzione è iniettiva e ciò prova che $Comp(N)$ è numerabile.

Indichiamo con $Par(N)$ l'insieme delle funzioni parziali o totali di N in N . Per provare che tale insieme ha la potenza del continuo ricordiamo che:

- l'insieme delle parti di N ha la potenza del continuo
- poiché ogni parte di N può essere identificata con la sua funzione caratteristica, l'insieme delle parti di N è equipotente ad $\{0,1\}^N$ cioè all'insieme delle funzioni di N in $\{0,1\}$.

Ne segue che $Par(N)$, contenendo $\{0,1\}^N$, ha la potenza maggiore o uguale al continuo. D'altra parte, poiché ogni funzione parziale è una relazione, $Par(N)$ è contenuto nell'insieme $P(N \times N)$ delle relazioni in N che ha la potenza del continuo. Pertanto $Par(N)$ ha potenza minore o uguale al continuo. In conclusione $Par(N)$ ha la potenza del continuo.

Infine se l'insieme $Par(N) - Comp(N)$ delle funzioni non computabili avesse cardinalità finita o numerabile, allora $Comp(N) = Comp(N) \cup (Par(N) - Comp(N))$ sarebbe numerabile, in contrasto con quanto ora dimostrato. Ne segue che l'insieme delle funzioni non computabili ha la potenza del continuo.

Teorema 5.2. La classe $Dec(N)$ dei sottoinsiemi decidibili di N è numerabile, pertanto esistono insiemi non decidibili (più precisamente la classe dei sottoinsiemi che non sono decidibili ha la potenza del continuo).

Dim. La classe degli insiemi decidibili di N ha la potenza del numerabile in quanto è numerabile l'insieme dei programmi per calcolare le relative funzioni caratteristiche. Più precisamente potrei considerare la funzione che associa ad ogni insieme decidibile S il minimo tra i numeri di codice dei programmi che computano la funzione caratteristica di S . In tale modo ottengo una funzione iniettiva a valori in N e ciò prova che $Dec(N)$ è numerabile. Supponiamo per assurdo che la classe $P(N) - Dec(N)$ dei sottoinsiemi non decidibili abbia cardinalità zero, finita o numerabile. Allora $P(N)$ in quanto unione di un insieme numerabile $Dec(N)$ e di un insieme $P(N) - Dec(N)$ vuoto, finito o numerabile avrebbe cardinalità numerabile. Ciò è un assurdo in quanto in contrasto con il teorema di Cantor. \square

Da notare che le dimostrazioni ora date non forniscono concretamente esempi di funzioni non computabili o di insiemi non decidibili. Inoltre nessuna considerazione sulla cardinalità permette di verificare se esistono insiemi effettivamente enumerabili che non siano decidibili. Infatti sia la classe dei semi-decidibili che quella dei decidibili hanno la potenza del numerabile. Il seguente teorema permette di rispondere a tali domande.

Teorema 5.3. (Problema della Fermata). Poniamo

$$K_0 = \{(i,j) \mid \pi_i \text{ converge in corrispondenza dell'input } j\},$$

allora K_0 è un insieme semi-decidibile che non è decidibile.

Dim. Che K_0 sia semi-decidibile segue dal fatto che posso immaginare un programma in una macchina a registri che, per decidere circa l'appartenenza di una coppia (i,j) a K_0 , per prima cosa:

- decodifica i in modo da ottenere un programma π_i (programma che computa la funzione f_i) e scrive questo programma nei suoi registri di memoria
- manda in esecuzione il programma π_i fornendo come input il numero j
- se π_i converge in j fornisce come output 1 (altrimenti non converge).

Questo è un tipico processo di semi-decidibilità perché se (i, j) è un elemento di K_0 questo processo di calcolo converge ed ha una risposta (positiva), se invece (i, j) non appartiene a K_0 il processo di calcolo non termina. Un modo equivalente di provare la semi-decidibilità di K_0 è osservare che K_0 è il dominio della funzione $f: N \times N \rightarrow N$ definita dal porre $f(i, j) = f_i(j)$ e che tale funzione è computabile.

Per provare che K_0 non è decidibile, concentriamo la nostra attenzione sull'insieme "più semplice"

$$K = \{i \in N : \pi_i \text{ converge in corrispondenza dell'input } i\}.$$

Osserviamo che

$$i \in K \Leftrightarrow (i, i) \in K_0$$

e quindi che $K \leq_r K_0$. Per provare che K è indecidibile supponiamo per assurdo che la sua funzione caratteristica c_K sia computabile. Esistono allora opportune istruzioni capaci di calcolare c_K . Indichiamo con *Function* $c_K \dots$;

questo blocco di istruzioni ed aggiungiamo ad esse le istruzioni

read(x);

1: IF $c_K(x) = 1$ THEN GOTO 1;

write(x).

in modo da ottenere il programma π

Function $c_K \dots$;

read(x);

1: IF $c_K(x) = 1$ THEN GOTO 1;

write(x).

E' immediato vedere che

$$c_K(x) = 0 \Leftrightarrow \pi \text{ converge in } x,$$

Infatti se $c_K(x) = 0$ allora dall'istruzione con etichetta 1 si passa alla successiva *write*(x) e ci si ferma. Invece se $c_K(x) = 1$ allora si ripete infinite volte l'istruzione con etichetta 1. Calcoliamo ora il numero di codice d di tale programma (si ricordi che il numero d si calcola secondo la procedura indicata nel paragrafo 9 del capitolo 3), allora $\pi = \pi_d$ e posto $x = d$.

$$\pi_d \text{ converge in } d \Leftrightarrow d \in K \Leftrightarrow c_K(d) = 1 \Leftrightarrow \pi_d \text{ non converge in } d.$$

Ciò è assurdo e tale assurdo deriva dall'aver supposto K decidibile. Pertanto possiamo concludere che K non è decidibile. Fatto questo, poiché $K \leq_r K_0$, risulta che K_0 non può essere decidibile perché in tale caso dovrebbe esserlo anche K . □

Tale teorema è di importanza fondamentale e si esprime spesso dicendo che:

il problema della fermata non è decidibile.

È questo il primo limite che troviamo alle cose che una macchina a memoria infinita (un programma) può fare. Tale teorema ci dice che:

non sarà mai possibile costruire un programma capace di controllare gli altri programmi avvertendoci, dato un programma (di indice i) ed un input j , se il calcolatore si ferma e stampa un output o se prosegue all'infinito.

Si noti che, come sanno tutti quelli che hanno scritto qualche volta dei programmi, invece esistono procedimenti (cioè programmi) che avvertono se un programma è scritto in modo formalmente corretto o meno.

Un'altra notevole proprietà dell'insieme K_0 è espressa dalla seguente proposizione che mostra come K sia, in un certo senso, il problema "più difficile" che ci sia tra quelli semi-decidibili. Infatti

ogni problema semi-decidibile si può ridurre al problema della fermata.

Teorema 5.4. L'insieme K_0 è un massimo rispetto la relazione di riducibilità nella classe En degli insiemi effettivamente enumerabili. In altri termini ogni insieme effettivamente enumerabile è riducibile a K_0 .

Dim. Sia X un insieme effettivamente enumerabile, allora esiste una funzione computabile f_i il cui dominio è X , e pertanto $j \in X$ se e solo se π_i converge in j . Allora, detta h la funzione tale che $h(x) = (i, x)$, risulta che

$$j \in X \Leftrightarrow \pi_i \text{ converge in } j \Leftrightarrow (i, j) \in K_0 \Leftrightarrow h(j) \in K_0.$$

Ciò prova che $X \leq_r K_0$. □

Ogni esempio di insieme non decidibile fornisce anche un esempio di insieme non effettivamente enumerabile. Infatti se X non è decidibile allora sappiamo che o X non è effettivamente enumerabile oppure X non è effettivamente enumerabile. Ad esempio, poiché K_0 è effettivamente enumerabile ma non decidibile, il complemento $\neg K_0$ di K_0 non può essere effettivamente enumerabile. Un altro esempio di insieme non effettivamente enumerabile è dato dalla seguente proposizione.

Proposizione 5.5. L'insieme $Tot = \{i \in N \mid f_i \text{ totale}\}$ non è effettivamente enumerabile (pur essendo numerabile) e pertanto non è decidibile.

Dim. Supponiamo per assurdo che Tot sia enumerato tramite la funzione computabile totale $h : N \rightarrow N$, cioè che $h(N) = Tot$. In altre parole supponiamo che l'insieme di tutte le funzioni computabili si possa rappresentare tramite la successione $f_{h(1)}, f_{h(2)}, \dots$. Vogliamo giungere ad un assurdo mostrando che esiste una funzione totale computabile g che non appartiene a tale successione. A tale scopo basta fare in modo che $g(1) \neq f_{h(1)}(1)$, (perché questo comporta che $g \neq f_{h(1)}$), $g(2) \neq f_{h(2)}(2)$ (perché questo comporta che $g \neq f_{h(2)}$) e così via. Un modo per ottenere questo è porre $g(x) = f_{h(x)}(x) + 1$. La funzione g ottenuta in questo modo è computabile e totale in quanto per calcolare il suo valore in x è sufficiente attivare il seguente algoritmo:

1. dato l'input x si calcola $h(x)$
2. si decodifica il numero $h(x)$ in modo da ottenere il programma $\pi_{h(x)}$ il cui numero di codice è $h(x)$
3. si fornisce come input a tale programma il numero x
4. si aggiunge al risultato il numero 1

Sia d un indice per tale programma, in altre parole sia d tale che cioè $g = f_d$. Essendo g totale, abbiamo che $d \in Tot$ ed essendo Tot il codominio di h , esiste $n \in N$ tale che $h(n) = d$. Quindi $g(n) = f_d(n) = f_{h(n)}(n)$. D'altra parte, per il modo come è stata definita g , è anche $g(n) = f_{h(n)}(n) + 1$, e ciò è assurdo. □

6. Non può esistere una macchina capace di controllare le altre macchine

La proposizione ora provata afferma che:

non è possibile costruire un programma capace di "controllare" gli altri programmi nel senso di dire, dato un programma π , se computa una funzione totale o meno.

In altre parole non può esistere un programma capace di avvisarmi, una volta che io abbia scritto un programma, se un programma conduce ad una funzione totale oppure ad una funzione parziale che quindi per alcuni input potrebbe andare in un loop infinito. Tale proposizione, che è relativa alla proprietà "essere totale", è rappresentativa di in una situazione più generale poiché per nessuna proprietà significativa può esistere un tale programma. Per dimostrare tale fatto premettiamo una proposizione, che è nota come *s-m-n*-teorema, che afferma che se ho un programma che computa una funzione di due variabili $h(x, y)$ allora per ogni modo di fissare un valore i per y ottengo un programma per la funzione $f(x) = h(x, i)$. Inoltre il numero di codice $s(i)$ di tale programma può essere calcolato in modo effettivo.

Proposizione 6.1. Se $h : N \times N \rightarrow N$ è una funzione computabile di due variabili allora esiste una funzione computabile totale s tale che

$$h(x, i) = f_{s(i)}(x).$$

Dim. Sia π un programma che computi la funzione $h(x,y)$. Tale programma richiede due input x ed y e quindi avrà due istruzioni del tipo $read(x)$ e $read(y)$. Fissiamo ora un numero intero i ed attiviamo la seguente procedura:

- sostituiamo in π l'istruzione $read(y)$ con l'assegnazione $y := i$ ottenendo un programma π' ,
- calcoliamo il numero di codice di π' che indichiamo con $s(i)$ (in quanto tale numero di codice dipende dal modo di fissare i).

E' facile rendersi conto che la funzione s è computabile. Infatti il procedimento ora descritto che permette di ottenere $s(i)$ a partire da i , è un processo effettivo. E' evidente che il programma π' computa la funzione f tale che $f(x) = h(x,i)$. D'altra parte, essendo $s(i)$ l'indice del programma π' , sarà $\pi' = \pi_{s(i)}$ ed $f = f_{s(i)}$ e quindi $h(x,i) = f_{s(i)}(x)$ per ogni input x . \square

Ad esempio consideriamo un programma per la somma, cioè $f(x,y) = x+y$:

```

read(x); read (y)
c:=0 ; s:=x
1. c:=c+1
   s:=s+1
   If c≤y then goto 1 else goto 2
2. write s.
```

Possiamo allora scrivere un programma per la funzione $f(x) = x+5$ ponendo al posto di $read(y)$ l'assegnazione $y := 5$,

```

read(x) ; y=5
c:=0 ; s:=x
1. c:=c+1
   s:=s+1
   If c≤y then goto 1 else goto 2
2 write s.
```

Codificando tale programma ottengo un indice $s(5)$. Se al posto di 5 pongo 7 otterrò un programma per $f(x) = x+7$. Indicherò con $s(7)$ l'indice di tale programma. E' evidente che la funzione s definita in questo modo è computabile.

Teorema 6.2 (Teorema di Rice). Sia C è una qualunque classe di funzioni parziali computabili diversa dalla classe vuota e dalla classe di tutte le funzioni parziali computabili. Allora l'insieme

$$I(C) = \{i \in \mathbb{N} \mid f_i \in C\}$$

non è decidibile.

Dim. **1° caso.** Supponiamo che la funzione vuota non appartenga a C . Poiché C è diverso dal vuoto, esiste una funzione computabile $f \in C$. Sia S un qualunque insieme effettivamente enumerabile che non sia decidibile e sia $h : \mathbb{N} \rightarrow \mathbb{N}$ è una funzione computabile enumerante S . Scriviamo un programma che accetta in entrata due input x ed y e che si comporti al modo seguente:

- se esiste i tale che $h(i) = y$ allora parte la procedura per il calcolo di $f(x)$ e poi tale procedura converge si fornisce $f(x)$ come output
- altrimenti il programma entra in un ciclo infinito.

Ad esempio, se π è l'insieme di istruzioni per il calcolo di $f(x)$ da cui è stata tolta l'istruzione $read(x)$, tale programma può essere realizzato al modo seguente:

```

Definition of h
read(x) ; read(y)
i := 0
```

```

1.  $i:=i+1$ 
   IF  $h(i) \neq y$  THEN Goto 1
    $\pi$ 

```

Detta $g(x,y)$ la funzione computabile corrispondente a tale programma abbiamo che per la proposizione 6.1 esisterà una funzione s totale computabile tale che $g(x,y) = f_{s(y)}(x)$. Inoltre avremo che:

- se $y \notin S$ allora $f_{s(y)}$ coincide con la funzione vuota (e quindi non appartiene a C)
- se $y \in S$ allora $f_{s(y)}$ coincide con f (e quindi appartiene a C).

In definitiva

$$y \in S \text{ se e solo se } s(y) \in I(C)$$

e quindi $S \leq_p I(C)$. Se allora per assurdo $I(C)$ fosse decidibile lo sarebbe anche S , in contrasto con le ipotesi.

2° caso. Supponiamo ora che C contenga la funzione vuota. Allora il complemento $-C$ di C non contiene la funzione vuota e quindi, per quanto ora dimostrato, l'insieme $\{i \in N \mid f_i \in -C\}$ non è decidibile. Ma allora anche il suo complemento $\{i \in N \mid f_i \in C\}$ non è decidibile. Poiché tale complemento coincide con IC la proposizione è provata anche in questo caso. \square

Si noti che le ipotesi per cui C è una classe diversa dall'insieme vuoto e dall'insieme di tutte le funzioni parziali ricorsive è necessaria. Infatti nel primo caso $I(C) = \{i \in N \mid f_i \in \emptyset\}$ coinciderebbe con l'insieme vuoto e nel secondo caso $I(C) = \{i \in N \mid f_i \text{ è parziale ricorsiva}\}$ coinciderebbe con N . D'altra parte sia l'insieme vuoto che N sono decidibili e quindi il teorema non è valido in tali casi.

Il Teorema di Rice ha un significato filosofico fondamentale. Interpretando la classe C come la classe di tutte le funzioni verificanti una data proprietà P , il teorema può essere enunciato al modo seguente:

Data una proprietà P , non esiste un procedimento generale che, partendo dall'analisi della struttura dei programmi, permetta di dire se la funzione computata verifichi P o meno.

È importante sottolineare che il termine "generale" gioca un ruolo essenziale nei teoremi limitativi che abbiamo dato. Ad esempio supponiamo che C sia uguale alla classe delle funzioni computabili che siano totali. Allora riotteniamo la proposizione 5.5 per cui non è decidibile se un dato programma determina una funzione totale oppure no. Tuttavia niente esclude l'esistenza di un procedimento che, per particolari tipi di programmi, riesca a dire se la funzione computata è totale. Un tale procedimento potrebbe essere del tipo:

- se nel programma non esistono "goto" ma solo istruzioni di assegnazione e del tipo "for $x:= \dots$ ", cioè cicli limitati, allora la funzione computata è totale ;
- se nel programma la prima istruzione è del tipo ' n goto n ' allora la funzione non è totale.

Non sarebbe difficile implementare un tale procedimento (ed anche procedimenti un poco più significativi di questo) in modo da avere degli avvisi al momento della compilazione del programma i quali, in almeno alcuni casi, avvertano se la funzione è totale o meno.

Naturalmente il teorema di Rice permette di mostrare molti esempi di insiemi non decidibili. Ponendo C uguale alla sola funzione fattoriale otteniamo che l'insieme $I(C) = \{i \in N \mid f_i \text{ è la funzione fattoriale}\}$ non è decidibile. Ponendo C uguale alla classe delle funzioni primitive ricorsive otteniamo che l'insieme $I(C) = \{i \in N \mid f_i \text{ primitiva ricorsiva}\}$ non è decidibile.

Proposizione 6.3. Non è decidibile se un programma calcola una funzione primitiva ricorsiva oppure no.

In altri termini non esiste un metodo generale che consenta, esaminando la struttura di un programma, se la funzione computata dal programma è primitiva ricorsiva oppure no.

Proposizione 6.4. Non è decidibile se due funzioni computabili sono uguali o meno, più precisamente, l'insieme

$$S = \{(i,j) \in N \times N \mid f_i = f_j\}$$

non è decidibile.

Dim. Fissiamo un intero j e poniamo $C = \{f_j\}$. Allora per il teorema di Rice l'insieme $IC = \{i \in N \mid f_i \in C\}$ non è decidibile. Poiché

$$(i, j) \in S \Leftrightarrow i \in IC,$$

S risulta essere riducibile ad IC e quindi non decidibile. \square

7. Computabilità e decidibilità in teoria degli insiemi.

Spesso è possibile "riscrivere" una nozione o un teorema classico in termini di matematica costruttiva. Ricordiamo ad esempio il famoso teorema di Cantor in cui si prova che $P(N)$ non è numerabile. La sua dimostrazione parte dal supporre per assurdo che esista una funzione biettiva $h: N \rightarrow P(N)$, si considera poi l'insieme $S = \{n \in N \mid n \notin h(i)\}$. Essendo h suriettiva esisterà j tale che $h(j) = S$. Pertanto:

$$j \in S \Leftrightarrow j \in h(j) \Leftrightarrow j \notin S_j.$$

L'assurdo cui siamo pervenuti prova la non numerabilità di $P(N)$. Nella seguente proposizione riproponiamo tale tecnica in termini costruttivi.

Proposizione 7.1. L'insieme delle parti decidibili di N non è effettivamente enumerabile (pur essendo numerabile). Più precisamente, l'insieme

$$P_d(N) = \{i \in N \mid f_i \text{ funzione caratteristica di un sottoinsieme di } N\}$$

non è effettivamente enumerabile.

Dim. Assumiamo per assurdo l'esistenza di una funzione suriettiva computabile $h: N \rightarrow P_d(N)$. Allora la sequenza $f_{h(1)}, f_{h(2)}, \dots$ rappresenterebbe tutte e sole le funzioni caratteristiche degli insiemi decidibili. Indichiamo con S_i l'insieme decidibile la cui funzione caratteristica è $f_{h(i)}$. Posto $S = \{i \in N \mid i \notin S_i\}$, è immediato vedere che S è decidibile. Infatti la sua funzione caratteristica c_S è definita da $c_S(i) = 1 - f_{h(i)}(i)$. Essendo S decidibile, deve esistere j tale che $S = S_j$. Si perviene allora all'assurdo per cui

$$j \in S \Leftrightarrow c_S(j) = 1 - f_{h(j)}(j) = 1 \Leftrightarrow f_{h(j)}(j) = 0 \Leftrightarrow j \notin S_j. \quad \square$$

Da notare che avremmo potuto dimostrare tale proposizione anche utilizzando il teorema di Rice. In teoria degli insiemi si dimostra che l'unione di due insiemi numerabili è ancora un insieme numerabile e che l'intersezione di due insiemi numerabili è un insieme finito o numerabile. La versione costruttiva di tali affermazioni è data dalla seguente proposizione.

Proposizione 7.2. L'unione e l'intersezione di due insiemi effettivamente enumerabili è un insieme effettivamente enumerabile.

Dim. Se S_1 e S_2 sono enumerati rispettivamente dalle funzioni computabili f e g , allora la loro unione è enumerata dalla funzione h definita da

$$h(n) = \begin{cases} f(n/2) & \text{se } n \text{ è pari} \\ g((n+1)/2) & \text{se } n \text{ è dispari.} \end{cases}$$

Infatti, per ogni intero i , $f(i) = h(2i)$ e $g(i) = h(2i-1)$.

Sia ora Z l'intersezione di S_1 e S_2 ; se $Z = \emptyset$ è ovvio che Z è effettivamente enumerabile. In caso contrario sia a un elemento fissato di Z , allora Z è enumerato dalla funzione computabile h di due variabili definita da

$$h(n, m) = \begin{cases} g(n) & \text{se } g(n) \in \{f(1), \dots, f(m)\} \\ a & \text{altrimenti.} \end{cases}$$

Infatti tutti i valori $h(n,m)$ appartengono a Z ; viceversa se z è un elemento di Z , allora esisteranno n ed m tali che $z = g(n) = f(m)$ e quindi $z = h(n,m)$.

Infine, l'insieme K costruito nel dimostrare il teorema della fermata è un insieme effettivamente enumerabile. Se il complemento di K fosse effettivamente enumerabile, allora K sarebbe decidibile in contrasto con quanto dimostrato. \square

Possiamo esprimere quanto ora provato anche in termini della nozione di reticolo.

Teorema 7.3. La classe $En(N)$ dei sottoinsiemi effettivamente enumerabili costituisce un sotto-reticolo $(En(N), \cup, \cap, \emptyset, N)$ del reticolo $(P(N), \cup, \cap, \emptyset, N)$ che non è un'algebra di Boole. Tale reticolo è una estensione del reticolo $(Dec(N), \cup, \cap, \emptyset, N)$ degli insiemi decidibili.

Problema. Il reticolo $(En(N), \cup, \cap, \emptyset, N)$ è completo? In altri termini l'unione di una successione di insiemi effettivamente enumerabili è un insieme effettivamente enumerabile?

Non tutte le proprietà della teoria degli insiemi hanno una versione costruttiva. Ad esempio è ovvio che, essendo tutti i sottoinsiemi di N enumerabili, il complemento di un sottoinsieme enumerabile di N è ancora enumerabile. Dal punto di vista costruttivo risulta invece la seguente proposizione.

Proposizione 7.4. Il complemento di un insieme effettivamente enumerabile non è necessariamente effettivamente enumerabile.

Dim. Basta considerare un qualunque insieme che sia effettivamente enumerabile ma non decidibile (e quindi con complemento non effettivamente enumerabile).

Una nozione fondamentale in teoria degli insiemi è quella di equipotenza e di cardinalità. Possiamo ridefinire tali concetti al modo seguente.

Definizione 7.5. Diciamo che l'effettiva cardinalità di X è minore dell'effettiva cardinalità di Y , e scriviamo $X \leq_e Y$, se esiste una applicazione iniettiva e computabile di X in Y . Diciamo che X è equipotente ad Y in modo effettivo e scriviamo $X \equiv_e Y$, se esiste una applicazione biettiva e computabile tra X ed Y .

Da notare che dire che $X \leq_e Y$ equivale a dire che X è riducibile ad Y tramite una funzione computabile ed iniettiva. E' possibile provare un teorema analogo a quello di Cantor-Bernstein.

Teorema 7.6. Se X ed Y sono sottoinsiemi di N tali che $X \leq_e Y$ e $Y \leq_e X$ allora X ed Y sono effettivamente equipotenti.

Da tale teorema segue che se X è equipotente ad Y in modo effettivo allora

X è decidibile $\Leftrightarrow Y$ è decidibile.

X è effettivamente enumerabile $\Leftrightarrow Y$ è effettivamente enumerabile.

Da notare che dal punto di vista della teoria degli insiemi di Cantor, tutti i sottoinsiemi infiniti di N sono numerabili e quindi equipotenti tra loro. Invece nell'approccio costruttivo proposto in questo paragrafo un sottoinsieme (infinito) di N che sia non decidibile non può essere equipotente in modo effettivo ad un insieme decidibile, ad esempio, ad N . Quindi, ad esempio, l'insieme K_0 rappresentante il problema della fermata non può essere effettivamente equipotente ad N (pur essendo equipotente ad N in quanto numerabile).

Quanto detto in questo paragrafo suggerisce un campo di ricerca che si potrebbe chiamare "teoria degli insiemi costruttiva".

8. Computabilità e decidibilità in analisi ed in algebra

La teoria della computabilità che abbiamo fino ad ora esposto determina un tipo di matematica notevolmente diverso dalla matematica classica. Questo lo abbiamo visto già a livello di teoria degli insiemi quando, ad esempio, abbiamo evidenziato che la classe dei sottoinsiemi decidibili è numerabile. Potremmo chiamare *approccio costruttivo* un tale modo di vedere le cose. In questo paragrafo daremo dei brevi cenni su come circa l'impatto dell'approccio costruttivo sull'analisi matematica. Nel seguito supporremo sempre che i numeri reali siano rappresentati in base due ed in modo che il periodo 1 non sia mai presente. Ad esempio, invece di scrivere $1,0101111\dots$ scriveremo $1,0110000\dots$. Con tale precauzione abbiamo che due numeri sono diversi se e solo se le relative espansioni sono diverse.

Definizione 8.1. Un numero reale r è detto *computabile* se possiede una espansione binaria

$$r = n, r(1)r(2)\dots$$

con n numero intero relativo scritto in base due ed $r : \mathbb{N} \rightarrow \{0,1\}$ funzione totale computabile.

In altri termini un numero reale è computabile se esiste un algoritmo capace di calcolare, per ogni intero n , la cifra binaria n -esima. Sono computabili i numeri razionali, ed i numeri $\sqrt{2}$, π , e . Ma, ovviamente, sono computabili tutti i numeri usualmente utilizzati in matematica (altrimenti non sarebbero utilizzabili!). Nel seguito per comodità di esposizione ci riferiremo solo ai numeri reali computabili dell'intervallo $[0,1]$, cioè ai numeri del tipo $0, r(1)r(2)\dots$. Ciò significa che possiamo identificare tale classe con quella delle funzioni totali computabili $r : \mathbb{N} \rightarrow \{0,1\}$.

Teorema 8.2. La classe $Comp([0,1])$ dei numeri reali computabili dell'intervallo $[0,1]$ è numerabile, pertanto esiste una infinità non numerabile di numeri reali non computabili.

Dim. Il risultato dovrebbe risultare ovvio per quanto detto precedentemente, comunque, per una maggiore comprensione di quanto affermato, scriviamone esplicitamente la dimostrazione. Che la classe $Comp([0,1])$ sia numerabile deriva dal fatto che si può definire la corrispondenza $f : Comp([0,1]) \rightarrow \mathbb{N}$ che ad ogni reale computabile associa l'indice (ad esempio il minimo indice) di una funzione che ne calcola l'espansione decimale. Ciò assicura che la cardinalità di $Comp([0,1])$ è minore o uguale al numerabile. Poiché tutti i numeri razionali sono computabili, e l'insieme dei razionali a cardinalità uguale al numerabile, $Comp([0,1])$ ha anche cardinalità maggiore o uguale al numerabile. Pertanto possiamo concludere che $Comp([0,1])$ è numerabile. \square

Ne segue che usualmente nella matematica classica ci si riferisce ad un universo, quello dei numeri reali, di cui solo una piccola fetta numerabile (quella dei numeri computabili) sembra avere significato.

Proposizione 8.3. La classe $Comp([0,1])$ dei numeri reali computabili non è effettivamente enumerabile. Più precisamente non esiste una funzione totale computabile $h : \mathbb{N} \rightarrow \mathbb{N}$ tale che nella successione

$$r_1 = 0, f_{h(1)}(1) f_{h(1)}(2) f_{h(1)}(3) \dots$$

$$r_2 = 0, f_{h(2)}(1) f_{h(2)}(2) f_{h(2)}(3) \dots$$

\dots

$$r_i = 0, f_{h(i)}(1) f_{h(i)}(2) f_{h(i)}(3) \dots$$

\dots

siano rappresentati tutti gli elementi di $Comp([0,1])$.

Dim. Supponiamo per assurdo che una tale funzione h esista. Definiamo allora il numero reale $r = 0, f(1)f(2)\dots$ in modo che $f(i) \neq f_{h(i)}(i)$, in modo cioè che la cifra i -esima di r sia diversa dalla cifra i -esima del numero r_i . Ad esempio possiamo porre $f(i) = 1 - f_{h(i)}(i)$. Poiché la funzione f è effettivamente computabile, il numero r risulta essere computabile e quindi deve esistere un indice i tale che $r = r_i$. Ma ciò è assurdo poiché la cifra i -esima di r è $f(i) = 1 - f_{h(i)}(i)$ mentre la cifra i -esima di r_i è $f_{h(i)}(i)$. \square

Per trovare esempi particolari di numeri non computabili è sufficiente osservare che ogni sottoinsieme X di N possiamo associare un numero reale $n(X)$, compreso tra 0 ed 1, la cui espansione è $0, c_X(1)c_X(2) \dots$. Inoltre in tale corrispondenza X è decidibile se e solo se $n(X)$ è computabile. Allora, ad esempio, se c è la funzione caratteristica dell'insieme $K = \{i \in N \mid \pi_i \text{ converge in } i\}$ relativo al problema della fermata, il numero reale $0, c(1)c(2) \dots$ non è computabile.

È possibile trovare numerosi esempi di come la teoria della computabilità possa far luce sopra aspetti fondamentali della matematica. Di fondamentale importanza sono alcuni teoremi che mostrano come non sia possibile trovare una soluzione generale per molti problemi della matematica. In particolare, consideriamo il decimo problema di Hilbert con cui è iniziato questo capitolo.

Definizione 8.4. Chiamiamo *diofantina* una equazione $p(x_1, \dots, x_n) = 0$ dove p è un polinomio i cui coefficienti sono numeri interi e dove si è interessati solo a soluzioni intere di tale equazione.

Un esempio di equazione diofantina è il seguente:

$$2x-6=0$$

che ammette come soluzione il numero 3. Invece l'equazione $5x-6=0$ non ammette soluzioni intere. Più in generale potrei pormi il problema se un'equazione come $ax+b=0$ con a e b interi ammette soluzioni intere oppure se una equazione $ax^2+bx+c=0$, con a, b, c interi ammette soluzioni intere.

Proposizione 8.5. Se limitato alle equazioni di primo e secondo grado il problema di Hilbert è decidibile.

Dim. Dobbiamo provare che l'insieme

$$X = \{(a,b) \in Z^2 : \text{tale che esiste } x \text{ intero per cui } ax+b=0\}$$

è decidibile. La cosa non è difficile poiché $(a,b) \in X$ se e solo se $-b/a$ è un intero, cioè se e solo se b è divisibile per a . Dobbiamo inoltre provare che l'insieme

$$\{(a,b,c) \in Z^3 : \text{esiste } x \in Z \text{ tale che } ax^2+bx+c=0\}$$

è decidibile. In tale caso basta applicare la formula risolutiva delle equazioni di secondo grado e verificare se una delle due soluzioni sia intera oppure no.

Naturalmente il problema di Hilbert è più generale perché si riferisce ad una qualunque equazione ed a qualunque numero di gradi.

Decimo Problema. *Data una equazione diofantina con un qualunque numero di incognite ed a coefficienti interi individuare un processo che permetta di decidere con un numero finito di operazioni se l'equazione ammette soluzioni intere o meno (Hilbert 1900).*

Tale problema è stato risolto negativamente da Yuri Matijasevic, il quale dimostra che non esiste il procedimento richiesto da Hilbert, cioè che il problema è indecidibile.

Teorema 8.6. L'insieme

$$E = \{(a_0, \dots, a_h) \mid a_h x^h + a_{h-1} x^{h-1} + \dots + a_0 = 0 \text{ ammette soluzioni intere}\}$$

è parzialmente decidibile (cioè effettivamente enumerabile) ma non è decidibile.

Dim. (parte facile) Proviamo solo la parte facile della proposizione, cioè che E è parzialmente decidibile. Infatti, dato l'input (a_0, \dots, a_h) possiamo controllare man mano se i numeri 1, 2, 3, ... sono soluzioni dell'equazione. Se esiste la soluzione, con questo procedimento me ne accorgo, se non esiste, tale procedimento continua all'infinito. Più concretamente, consideriamo il programma

`read(a0, ..., ah) ; x := 0;`

1. `x := x+1 ;`

IF `ahxh+ah-1xh-1+ ... +a0 = 0` THEN GOTO 2 ELSE GOTO 1

2. `WRITE(1)`

Tale programma stampa 1 se il polinomio $a_h x^h + a_{h-1} x^{h-1} + \dots + a_0$ ammette radici intere, diverge altrimenti. In altre parole computa una funzione parziale che ha come dominio l'insieme E . Ciò prova che tale insieme è effettivamente enumerabile.

Lettura: **DESTINI PROGRAMMATI**

di

Giangiacomo Gerla

Pirati ed Integrati. Nel Paese del Destino, esistono due tipi di abitanti, i Pirati e gli Integrati. Un Pirata ha abitudini semplici. Ogni giorno passeggia come un vagabondo per le campagne o per le strade di città, gioca e naviga per la rete con il proprio portatile per poi addormentarsi all'aria aperta sotto una coperta (per fortuna non piove mai in questo paese). In ogni istante un Pirata è libero di scegliere cosa fare, se cambiare città, se restare a dormire, se prendere la strada a sinistra o quella a destra . . . e così via. Gli Integrati invece vanno a lavorare ed ogni giorno fanno quello che è scritto nel Libro dei Regolamenti del Proprio Mestiere.

Anche io abito in questo paese e, non me ne vergogno, sono un Pirata. Mi piace questa vita poiché si vedono strade, città, boschi, montagne sempre nuovi. Ogni mattina mi sveglio alle nove e trenta ed ogni mattina, dopo colazione, decido liberamente se andarmene a passeggiare o starmene sdraiato a leggere ed a riposare per tutta la giornata. La sera posso andare all'Internet-bar dove si può chiacchierare con altri Pirati, scambiandosi l'ultimo interessante indirizzo internet o raccontando come quella volta si è riusciti ad entrare in questo o quel sito segreto del Grande Fratello. Oppure posso restare a sedere sotto un albero con il mio fedele portatile per navigare in tutta l'amata-odiata rete. Niente di più bello del momento in cui si riesce a penetrare nei siti più protetti del mondo spargendo virus e scherzi. Solo così è possibile prendere in giro il Grande Fratello che cerca in tutti i modi di condizionarci.

Quella sera, come tutte le sere, mentre più soddisfatto del solito mi accoccolavo per addormentarmi accanto ad un muretto in una campagna, mi venne da mormorare nel silenzio della serata

"faccio una vita serena e sono un essere libero, libero ogni mattina di andare a destra o a sinistra come mi detta la fantasia".

Ma, come rispondendo alla mie parole, ecco una voce ironica provenire dall'altra parte del muretto

"non ti illudere, non ti illudere".

Era un GI (Giovane Integrato) emerso dal buio con il suo cravattino, la camicia celeste e la giacchetta blu. Lo guardai con fastidio (non correva buon sangue tra gli Integrati ed i Pirati) e pensai che, essendo domenica, doveva trattarsi del solito integrato che trascorreva un fine settimana tra le montagne. Era strano comunque che non avesse gli scarponi, gli occhiali scuri, la giacca a vento firmata e tutto quanto era previsto dal Regolamento Generale degli Integrati in Fine Settimana. Aveva invece in mano il suo piccolo portatile, per la verità più bello, veloce e moderno del mio (i Pirati non davano a vederlo, ma invidiavano sempre un po' i soldi degli Integrati).

"Non ti illudere sulla tua libertà", proseguì, "il tuo destino, come quello di tutti è già definito e basta caricare un semplice programma per prevedere quello che ciascun pirata farà giorno per giorno ed ora per ora".

Guardai il suo portatile un po' preoccupato. Era acceso ed il GI, con aria di sfida, si stava collegando con il computer centrale del suo ufficio. Ed ecco apparire sullo schermo la scritta

REGISTRO GENERALE DEI DESTINI

e sotto

**INSERIRE IL NUMERO DI CODICE DEL PIRATA
INPUT N**

"Facciamo un esempio a caso" disse GI, "inseriamo il numero di codice 23444443444"

Un ticchettio e . . .

ATTENDERE PREGO:

. . .

STIAMO CARICANDO IL PROGRAMMA PER IL DESTINO DEL PIRATA 23444443444

. . .

Ed ecco che dopo un po' appare una schermata di stile un po' anonimo con su scritto

*DESTINO DEL PIRATA DI CODICE 23444443444
INSERIRE LA DATA E L'ORA*

Provai a mettere la data del giorno dopo 03.07.2321 e le ore 10 ed ecco apparire la scritta
IL PIRATA 23444443444 IL GIORNO 03.07.2321 ALLE ORE 10 LASCERA' LA CITTA'

- *Ma, come si chiama il pirata con questo codice ?*

- *Non lo so, al momento della nascita ad ognuno viene assegnato un codice ma il numero di codice resta segreto per questione di privacy. In ogni caso, che tu ci creda oppure no il destino di ciascuno è determinato. Se ora sapessi il tuo codice potrei dirti esattamente quello che farai domani.*

Era terribile pensare che in quel momento c'era un Pirata di cui stavo leggendo il destino e che felice ed ignaro si illudeva ogni giorno di fare liberamente le sua scelte!

Come odiavo i computer e la Grande Rete che sembrava intrappolare tutte le persone del mondo !

E, principalmente, che tristezza sapere che forse il Giovane Integrato aveva ragione e che il mio destino poteva essere stampato in ogni momento da uno squallido programma in uno squallido computer!

"Che senso ha la vita di un Pirata", pensai, "se le sue scelte sono solo una illusione, se tutto è già stabilito? Non sarebbe meglio allora risparmiarsi i reumatismi, trovarsi un impiego e tutte le mattine recarsi ubbidienti al proprio posto di lavoro? Abbasso i computer con la loro diabolica abilità di effettuare calcoli !

Il sonno mi era completamente passato ed anche il mio portatile, che era sempre stato il mio orgoglio e la mia compagnia, cominciava a sembrarmi un oggetto odioso.

Eppure non mi fidavo: poteva darsi che il Giovane Integrato mi stesse mentendo e che un uomo libero potesse realmente esistere in questo Paese. Chi mi poteva assicurare che il programmino che avevo visto non era tutta una finzione ? Bisognava verificare, bisognava trovare una via di uscita; bisognava pensare, pensare, pensare . . .

Ed ecco, all'improvviso, un'idea !

Cominciai a ticchettare sul mio computer per collegarmi al Registro Generale dei Destini. Era difficile trovare il codice segreto d'ingresso ma ben presto ci riuscii: d'altra parte altrimenti non sarei stato il migliore pirata informatico del mondo !

Finalmente sereno, mi addormentai, stringendo in mano un foglietto con sopra il faticato codice di ingresso. Sapevo quello che avrei dovuto fare l'indomani ed il portatile che mi stava accanto ora mi sembrava un'ancora di salvezza, non più una trappola.

Il giorno dopo. Al mattino, come tutte le mattine, il leggero calore del sole mi svegliò ed io mi alzai ben riposato. L' Integrato se ne stava ancora lì a guardarmi, cosa questa alquanto strana visto che era lunedì e che ogni Integrato il lunedì sta al suo posto di lavoro. Non me ne preoccupai, anzi ero ansioso di iniziare la mia passeggiata quotidiana e di dimostrargli quanto si sbagliasse. Come tutte le mattine, avevo davanti a me tutta la giornata e potevo scegliere liberamente cosa fare. Accesi il mio portatile e mi collegai al Registro Generale dei Destini, forte ormai della conoscenza del codice segreto. Mi apparve subito il programma *Destini* e la richiesta di inserire il numero di un Pirata. Misi il numero 1, e mi apparve la solita schermata in cui si chiedeva la data del giorno e l'ora. Visto che erano quasi le dieci, inserii, appunto le ore 10. Dopo poco apparve la risposta

*IL DESTINO DEL PIRATA 1
NEL GIORNO 03.07.2321 ALLE ORE 10*

E' :

ANDARSENE A PASSEGGIARE PER LE CAMPAGNE

Sapevo cosa fare: visto che il Pirata numero 1 se ne andrà a passeggiare per le campagne, oggi me ne resto sdraiato a leggere e mangiucchiare per tutta la giornata. In tale modo posso almeno evitare che il mio destino sia quello del Pirata 1.

E così feci restando a pancia all'aria chiacchierando al fianco del Giovane Integrato per poi addormentarmi contento. Il mattino dopo, al risveglio, dopo avere aspettato fino a poco prima delle dieci, accesi di nuovo il mio portatile, e chiesi al Registro Generale cosa avrebbe fatto il Pirata 2 alle ore 10. La risposta fu

"IL DESTINO DEL PIRATA 2

NEL GIORNO 04.07.2321 ALLE ORE 10

È :

STARE SDRAIATO A LEGGERE

Aspettai alle ore 10 in punto poi, svelto, mi misi a camminare. Ero contento poiché in tale modo il mio destino non poteva essere nemmeno quello del Pirata 2. Insomma, avevo trovato il metodo sicuro perché il mio destino non potesse essere quello del Pirata 3, né quello del Pirata 4 e così via. In definitiva, nessuno dei destini previsti dall'Istituto Centrale Destini poteva essere il mio e ciò mi metteva in allegria e mi spingeva a prendere in giro il Giovane Integrato per le sue fandonie sul destino. Ero anzi convinto che il Giovane Integrato si fosse convertito alla vita dei Pirati, visto che tutti i giorni bighellonava con me lungo le strade.

Ma anche il Giovane Integrato era contento ed io non capivo perché. Egli pensava al suo lavoro, che come tutti gli Integrati amava molto:

"questo lavoro non è poi tanto difficile, peccato che mi obblighi a stare tanto con quei buzzurri ignoranti dei pirati"

e guardandomi pensava:

"Tu ritieni di essere libero, eppure io, giorno per giorno, conosco esattamente ciò che farai alle ore 10 di domani!"

NOTA SERIA

Il racconto mostra che, per quanto sia potente un computer e per quanti possibili destini sia capace di descrivere è sempre possibile costruirsi un nuovo destino che non è previsto da tale computer. Per evidenziare gli aspetti matematici di un tale fenomeno, indichiamo con 0 l'azione del "passeggiare" e con 1 l'azione del "restare a leggere". In questo modo possiamo chiamare *destino* una qualunque funzione computabile $d : N \times N \times N \rightarrow \{0,1\}$ il cui significato è che $d(a,g,o)$ è l'azione che sarà fatta nell'anno a , nel giorno g ed all'ora o . Indichiamo con *Prog* l'insieme di tutti i possibili programmi che il Grande Fratello può contenere e con *Dest* l'insieme di tutti i possibili programmi capaci di calcolare un destino. Secondo il GI il calcolatore sarebbe capace, dato il numero di matricola n di un qualunque pirata, di trovare nella sua memoria un programma $\pi(n) \in \text{Dest}$ capace di calcolare il destino d_n di n . Il Pirata invece riesce a trovare un algoritmo, e quindi un programma $\pi \in \text{Prog}$, per una funzione-destino $d : N \times N \times N \rightarrow \{0,1\}$, tale che d è diversa da $\pi(n)$ per ogni indice n . Allora un qualunque "processo" che associa ad ogni intero n un elemento $\pi(n)$ in *Dest* lascerà sempre fuori qualche elemento di *Dest*. In altre parole non esiste una funzione computabile $\pi : N \rightarrow \text{Dest}$ che sia suriettiva. In termini informatici questo significa che *Dest* non è un insieme effettivamente enumerabile (pur essendo ovviamente numerabile). Più precisamente il racconto mostra che *Dest* è un insieme *produttivo* cioè che ogni algoritmo che tenti di enumerare gli elementi di *Dest* è destinato a fallire poiché è sempre possibile costruire un elemento di *Dest* che non appartiene a tale enumerazione. Naturalmente *Dest*, non essendo effettivamente numerabile non è neanche decidibile.

Un altro modo di interpretare il racconto è come prova che non può esistere un linguaggio di programmazione capace di calcolare tutte le funzioni totali che sono intuitivamente computabili.

Ma nel racconto la situazione è più complicata perché il metodo trovato dal pirata è a sua volta un algoritmo che può essere implementato da un programma. Niente proibisce di aggiungere tale programma nel calcolatore ... Questo è un fenomeno tipico degli insiemi produttivi. Ad esempio se VA è l'insieme delle proposizioni vere dell'aritmetica allora non esiste un modo di produrre (cioè enumerare) in modo effettivo tutti gli elementi di VA . Infatti ogni volta che si tenta un enumerazione effettiva $\alpha_1, \alpha_2, \dots$ esisterà, come Goedel insegna, una asserzione vera α che non appartiene a tale enumerazione. Niente proibisce comunque di costruire una nuova enumerazione che include α . In definitiva si riesce a dimostrare che per ogni successione effettiva $\alpha_1, \dots, \alpha_n, \dots$ di elementi di VA esiste α in VA non appartenente a tale successione e non che esiste un α in VA che non appartiene e nessuna successione effettiva di elementi di VA .