Il Problema della Fermata

Prima di iniziare ad realizzare i nostri primi ALGORITMI vediamo a vedere un PROBLEMA INSOLUBILE, ovvero un problema che non può essere risolto da NESSUN algoritmo.

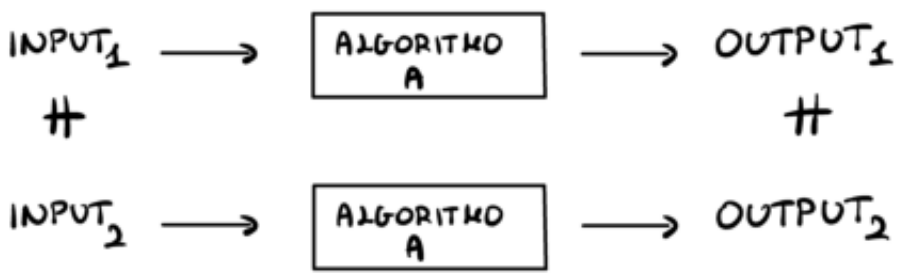
Prima di descrivere il problema ricordiamo alcuni concetti fondamentali:

- INPUT/OUTPUT
- TERMINAZIONE/LOOP INFINITO

INPUT/OUTPUT

Un ALGORITMO consiste in un insieme di ISTRUZIONI che ci permettono di trasformare un INPUT in un OUTPUT.

Fissato un algoritmo, cambiando l'INPUT è possibile che anche l'OUTPUT cambi.



In alcuni casi però anche se l'INPUT è diverso, l'OUTPUT potrebbe essere lo stesso, come nel caso del SORTING

In generale per capire come sono relazionati INPUT e OUTPUT dobbiamo analizzare attentamente l'algoritmo in questione.

TERMINAZIONE / LOOP INFINITO

Qualitativamente dato un ALGORITMO e un INPUT troviamo due COMPORTAMENTI diversi:

1) Nel primo caso l'algoritmo con quel particolare input TERMINA e produce un valore di output.

2) Altrimenti l'algoritmo con quel particolare input potrebbe NON TERMINARE MAI. In questo caso PATOLOGICO si dice che il particolare input porta l'algoritmo ad entrare in un LOOP INFINITO.

Qualche esempio...

```
CHECK PRIMENESS(integer n)
RESULT ← TRUE
FOR J←2 TO n-1
  IF J DIVIDES n
    RESULT ← FALSE
RETURN RESULT
```

ESCE DOPO n-2 ITERAZIONI

TERMINA ✓

```
LOOP()
WHILE TRUE
  PRINT "HELLO"

RETURN 0
```

NON ESCE MAI DA QUESTO CICLO

LOOP INFINITO ✗

PROBLEMI INDECIDIBILI

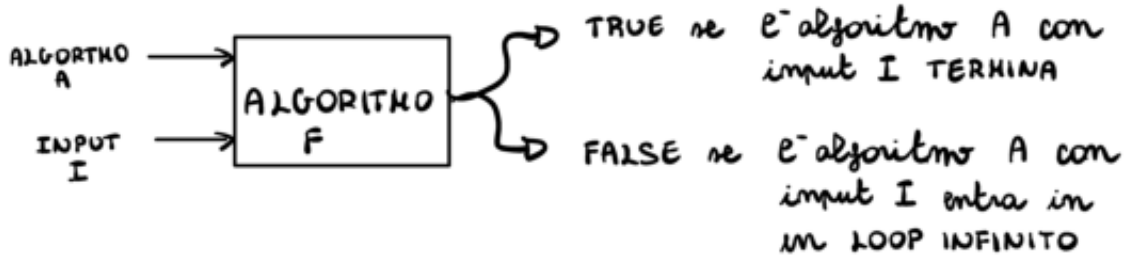
Siamo ora in grado di descrivere il nostro problema di interesse.

PROBLEMA DELLA FERMATA

Dato un ALGORITMO A e un INPUT I, stabilire se A con input I TERMINA oppure entra in un LOOP INFINITO.

Andiamo adesso a DIMOSTRARE che tale problema non può essere risolto. L'idea della dim. è la seguente: (REDUCTIO AD ABSURDUM ↕ DIMOSTRAZIONE PER ASSURDO)

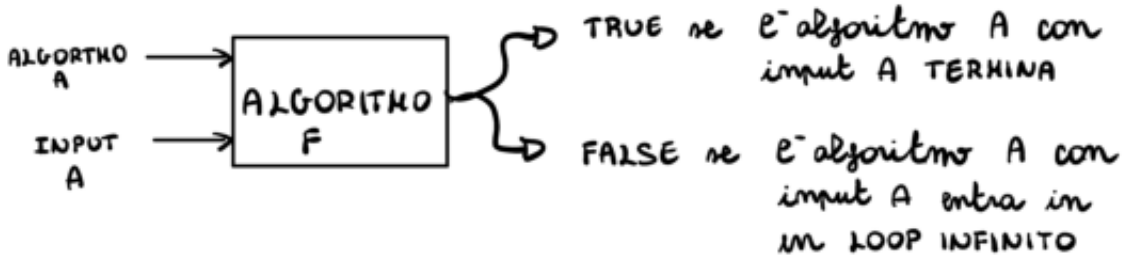
1) Assumiamo che il PROBLEMA DELLA FERMATA può essere risolto.
   In particolare assumiamo di avere un algoritmo F che risolve il nostro problema



2) A partire dall'algoritmo F COSTRUIAMO un nuovo algoritmo G.

3) A partire dall'algoritmo G deriviamo una CONTRADDIZIONE LOGICA.

Supponiamo quindi di avere l'algoritmo F



OSS: Non conosciamo i DETTAGLI IMPLEMENTATIVI dell'algoritmo F, ma solo come sono relazionati INPUT e OUTPUT.

A partire da F costruiamo l'algoritmo G

```
ALGORITMO G(A)
IF F(A,A) == TRUE
    WHILE TRUE
IF F(A,A) == FALSE
    RETURN
```

→

```
ALGORITMO G(G)
IF F(G,G) == TRUE
    WHILE TRUE
IF F(G,G) == FALSE
    RETURN
```

Consideriamo adesso che succede se proviamo a calcolare G(G).
Notiamo il seguente comportamento

- Se F(G,G) = TRUE, allora G(G) entra in un LOOP INFINITO.
  Ma dire F(G,G) = TRUE è equivalente a dire che l'algoritmo G con input G TERMINA.
  Dunque troviamo

  Se l'algoritmo G con input G TERMINA,
  allora l'algoritmo G con input G entra in un LOOP INFINITO.

- Se invece F(G,G) = FALSE, allora G(G) TERMINA.
  Ma dire F(G,G) = FALSE è equivalente a dire che
  l'algoritmo G con input G entra in un LOOP INFINITO.

  Dunque troviamo

  Se l'algoritmo G con input G entra in un LOOP INFINITO.
  allora l'algoritmo G con input G TERMINA

×

Notiamo che in entrambi i casi abbiamo trovato una CONTRADDIZIONE LOGICA.

Ora, dato che l'algoritmo G ci porta ad una CONTRADDIZIONE, questo significa che G NON PUÒ ESISTERE. Se analizziamo il codice di G notiamo come l'unica cosa che utilizziamo interamente è proprio l'algoritmo F. Troviamo quindi,

> Dato che G non può esistere, e dato che per costruire G necessitiamo solamente dell'esistenza di F, ne consegue che F non può esistere.

Graficamente,

Per finire ci basta notare che F è un qualsiasi algoritmo che risolve il PROBLEMA DELLA FERMATA. Vale quindi

TEOREMA (TURING, 1937)

Il PROBLEMA DELLA FERMATA è un PROBLEMA INDECIDIBILE.

NOTE STORICHE