Informatica per le discipline umanistiche

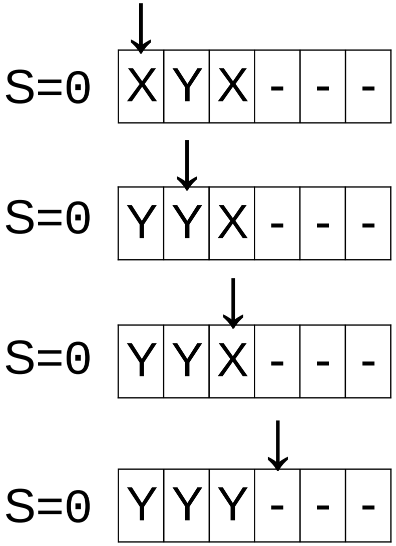
Lezione 4 – La macchina universale

cristiano.longo@unict.it

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che



Regole di transizione

`(0,X) → (0,Y,>)`

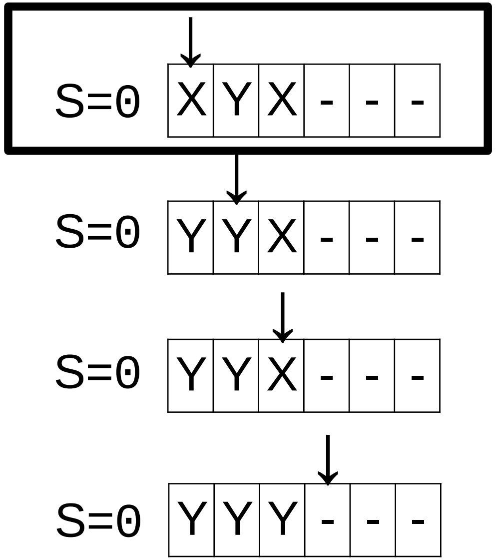
`(0,Y) → (0,Y,>)`

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),



Regole di transizione

(0,X) → (0,Y,>)

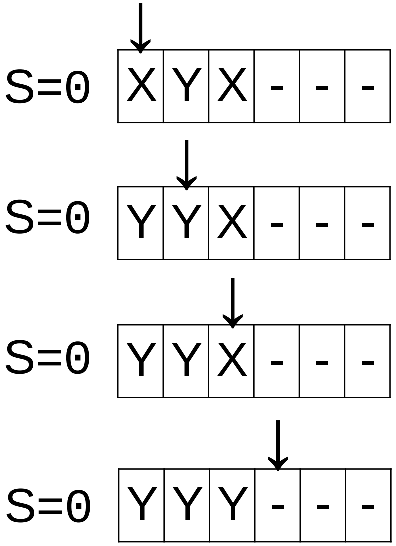
(0,Y) → (0,Y,>)

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),
- due configurazioni consecutive seguano le regole di transizione,



Regole di transizione

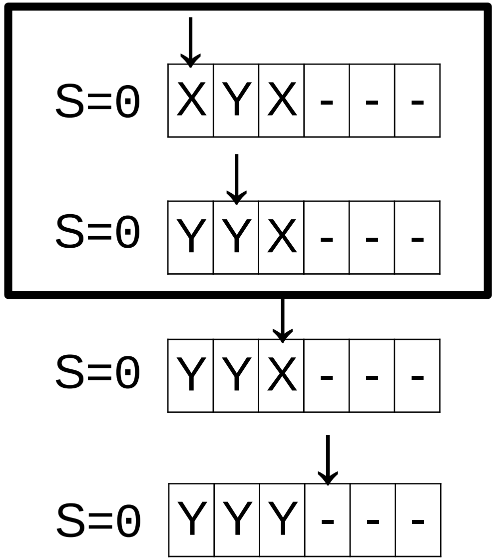
(0,X) → (0,Y,>)

(0,Y) → (0,Y,>)

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),
- due configurazioni consecutive seguano le regole di transizione,



Regole di transizione

(0,X) → (0,Y,>)

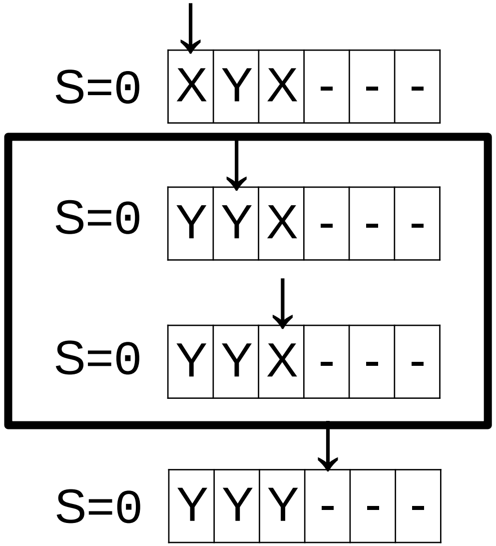
(0,Y) → (0,Y,>)

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),
- due configurazioni consecutive seguano le regole di transizione,



Regole di transizione

`(0,X) → (0,Y,>)`

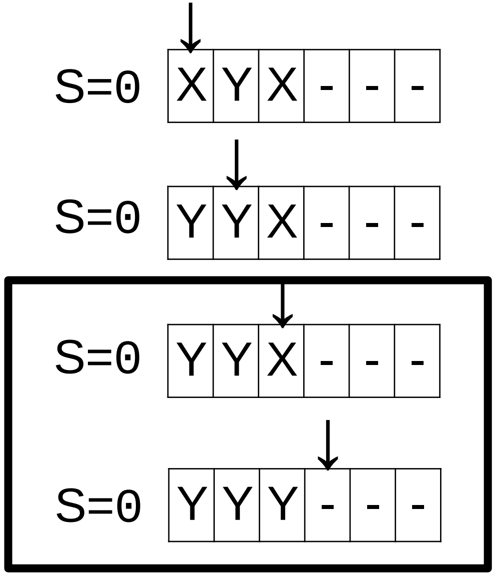
`(0,Y) → (0,Y,>)`

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),
- due configurazioni consecutive seguano le regole di transizione,



Regole di transizione

(0,X) → (0,Y,>)

(0,Y) → (0,Y,>)

Computazione

Richiamo: Una **configurazione** è caratterizzata dal contenuto del nastro (omettendo gli infiniti blank nella parte terminale del nastro), dalla posizione della testina, e dallo stato corrente.

Una **computazione** (o esecuzione) di una macchina di turing *T* è una sequenza di configurazioni tale che

- inizi con una **configurazione iniziale** (stato iniziale e testina all'inizio),
- due configurazioni consecutive seguano le regole di transizione,
- se finita, termini con una **configurazione finale** nella quale nessuna regola è applicabile.

	↓					
S=0	X	Y	X	-	-	-

		↓				
S=0	Y	Y	X	-	-	-

			↓			
S=0	Y	Y	X	-	-	-

				↓		
S=0	Y	Y	Y	-	-	-

Regole di transizione

`(0,X) → (0,Y,>)`

`(0,Y) → (0,Y,>)`

Computazione non terminante

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.

S=0	X	X	X	-	-	-

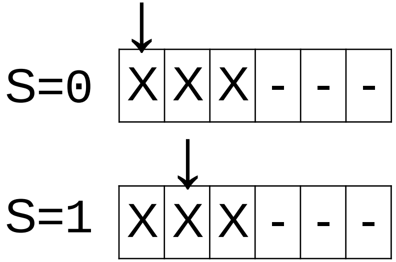
Regole di transizione

$(0,X) \rightarrow (1,X,>)$

$(1,X) \rightarrow (0,X,<)$

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



.
.
.

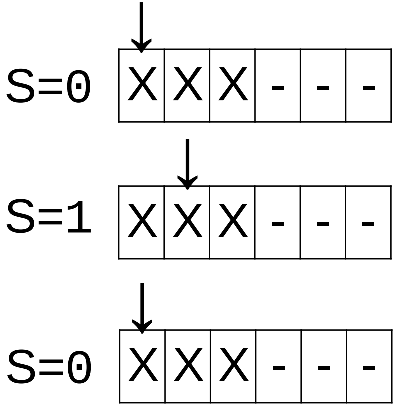
Regole di transizione

(0,X) → (1,X,>)

(1,X) → (0,X,<)

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



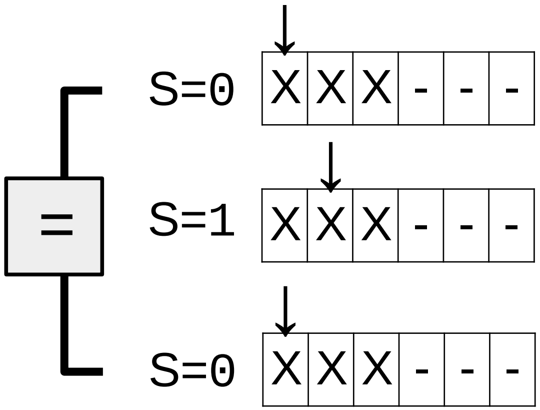
Regole di transizione

(0,X) → (1,X,>)

(1,X) → (0,X,<)

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



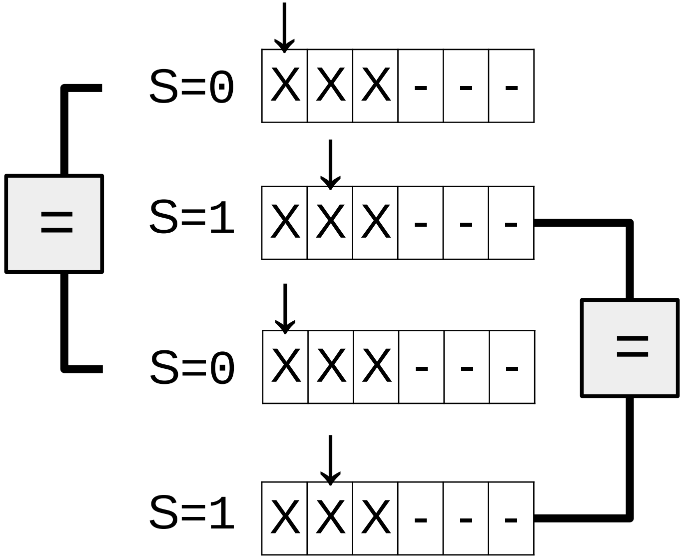
Regole di transizione

(0,X) → (1,X,>)

(1,X) → (0,X,<)

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



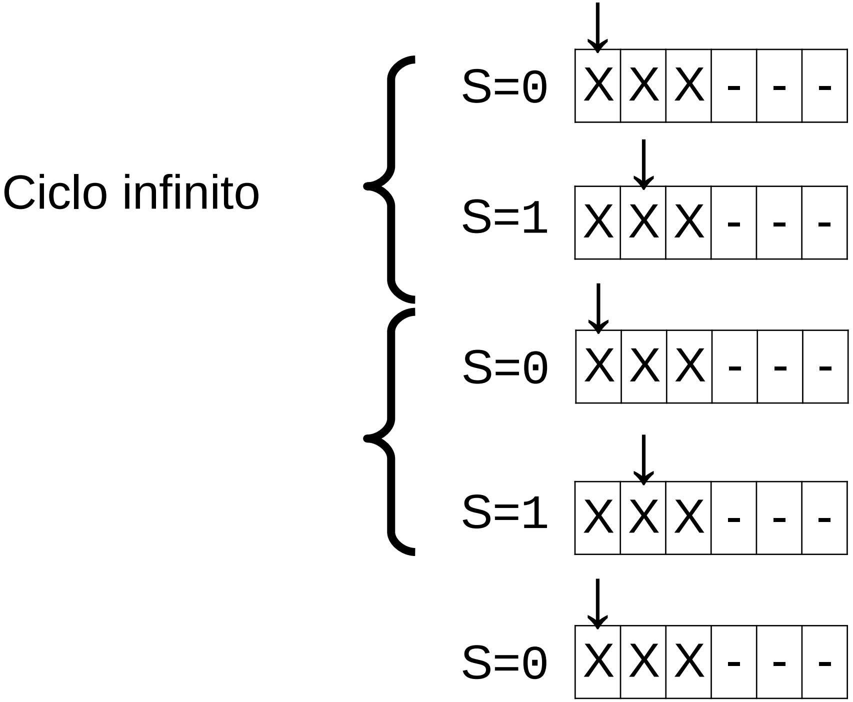
Regole di transizione

(0,X) → (1,X,>)

(1,X) → (0,X,<)

Computazione non terminante: esempio 1

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

`(0,X) → (1,X,>)`

`(1,X) → (0,X,<)`

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.

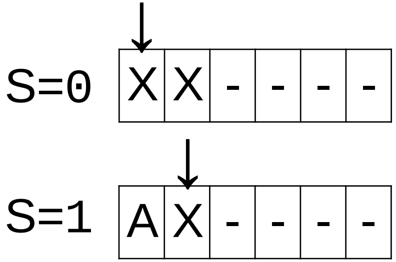
S=0	↓ X	X	-	-	-	-

Regole di transizione

```
(0,X) → (1,A,>)
(1,X) → (0,B,<)
(0,A) → (0,A,>)
(0,B) → (0,B,<)
```

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

(0,X) → (1,A,>)

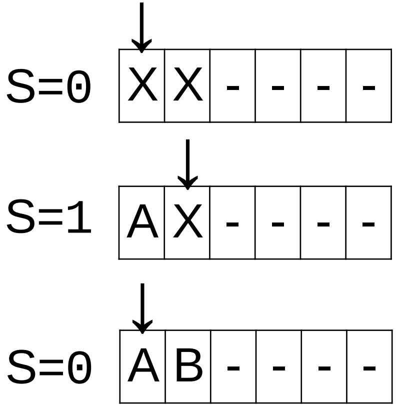
(1,X) → (0,B,<)

(0,A) → (0,A,>)

(0,B) → (0,B,<)

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

(0,X) → (1,A,>)

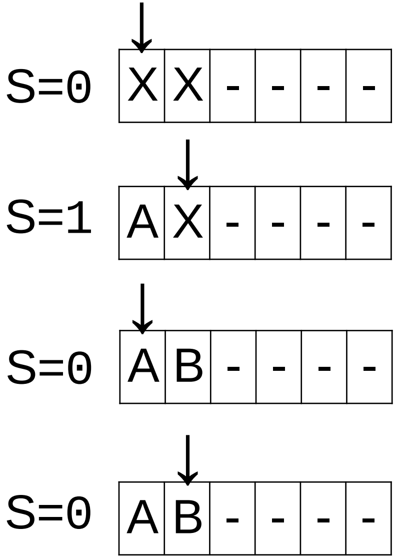
(1,X) → (0,B,<)

(0,A) → (0,A,>)

(0,B) → (0,B,<)

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

(0,X) → (1,A,>)

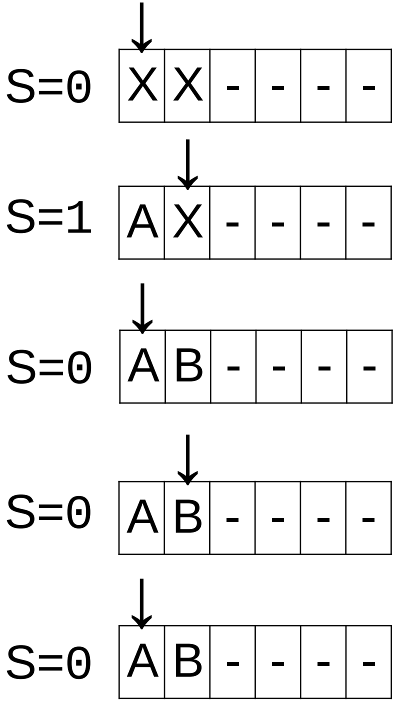
(1,X) → (0,B,<)

(0,A) → (0,A,>)

(0,B) → (0,B,<)

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

(0,X) → (1,A,>)

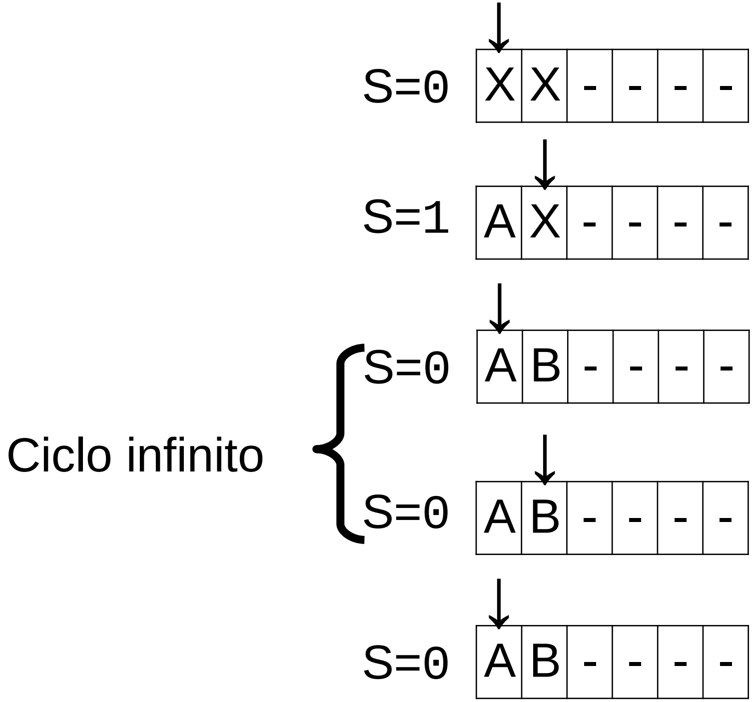
(1,X) → (0,B,<)

(0,A) → (0,A,>)

(0,B) → (0,B,<)

Computazione non terminante: esempio 2

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione

(0,X) → (1,A,>)

(1,X) → (0,B,<)

(0,A) → (0,A,>)

(0,B) → (0,B,<)

Computazione non terminante: esempio 3

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.

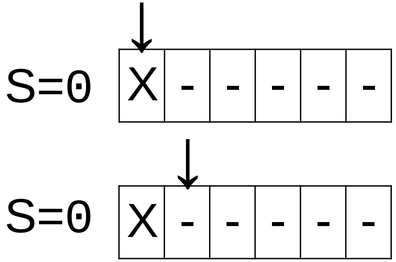
S=0

X	-	-	-	-	-

Regole di transizione
(0,X) → (0,X,>)
(0,-) → (0,X,>)

Computazione non terminante: esempio 3

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



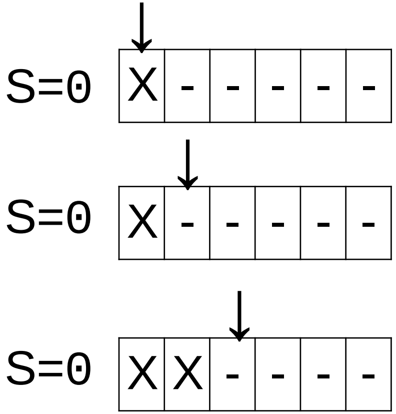
Regole di transizione

(0,X) → (0,X,>)

(0,-) → (0,X,>)

Computazione non terminante: esempio 3

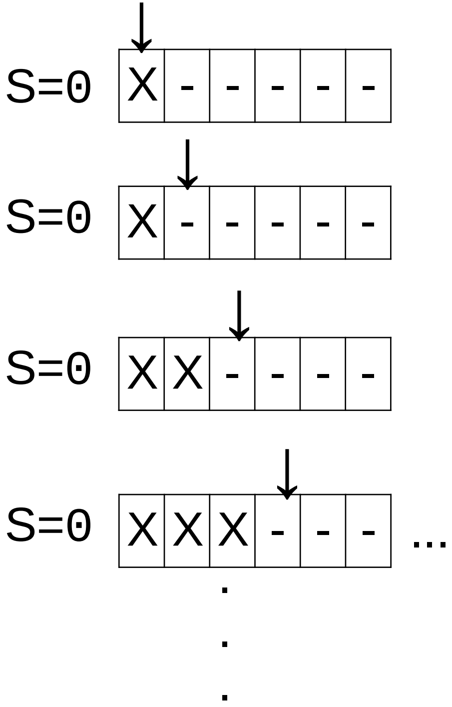
Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione
`(0,X) → (0,X,>)`
`(0,-) → (0,X,>)`

Computazione non terminante: esempio 3

Esistono macchine di Turing che, a partire da determinate configurazioni iniziali, danno luogo a computazioni infinite.



Regole di transizione
(0,X) → (0,X,>)
(0,-) → (0,X,>)

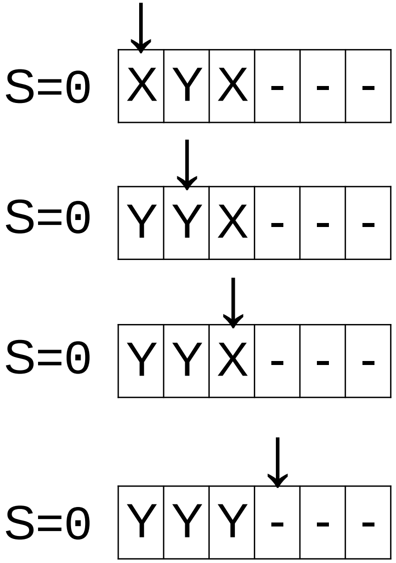
Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.

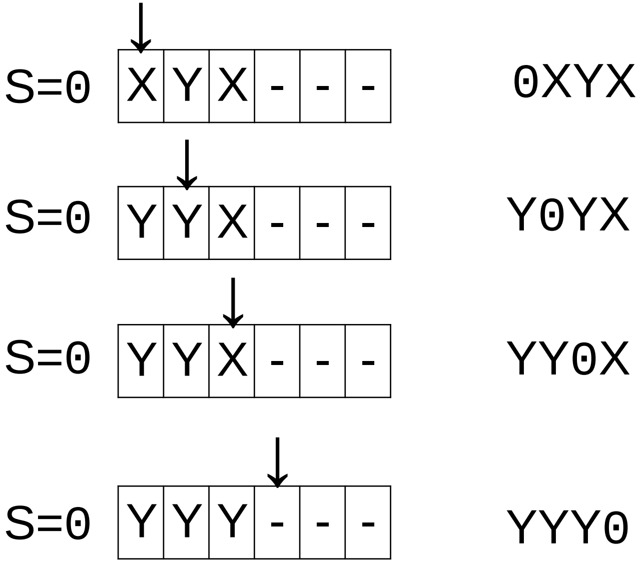


Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

vSw

con
- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.



Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- ***v* la parte di nastro che precede la testina,**
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.

S=0	X	Y	X	-	-	-

Se la testina è all'inizio, *v* è vuota.

Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- **S lo stato corrente,**
- *w* la parte di nastro a partire dalla testina.

S=**0**

X	Y	X	-	-	-

0

Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- ***w* la parte di nastro a partire dalla testina.**

S=0

X	Y	X	-	-	-

0XYX

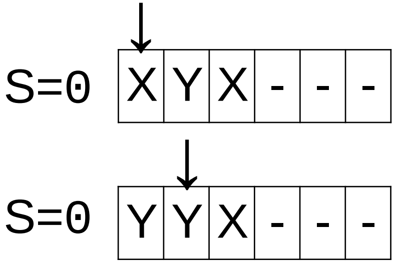
Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.



0XYX

Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- ***v* la parte di nastro che precede la testina,**
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.

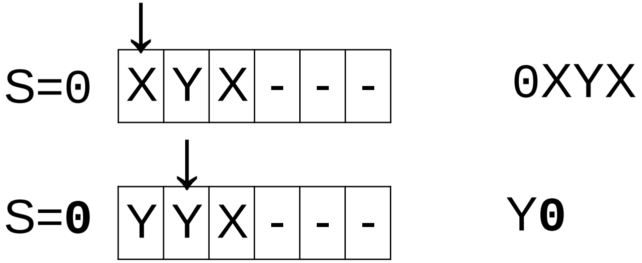
S=0	X	Y	X	-	-	-	0XYX
S=0	**Y**	Y	X	-	-	-	**Y**

Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

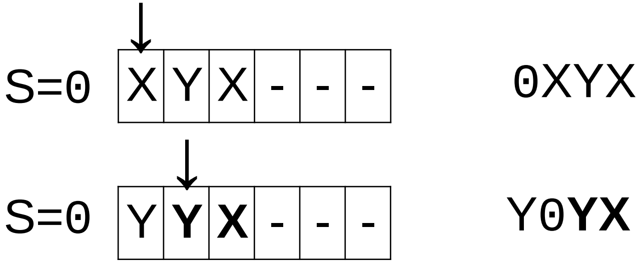
con
- *v* la parte di nastro che precede la testina,
- **S lo stato corrente**,
- *w* la parte di nastro a partire dalla testina.



Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$



con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- ***w* la parte di nastro a partire dalla testina.**

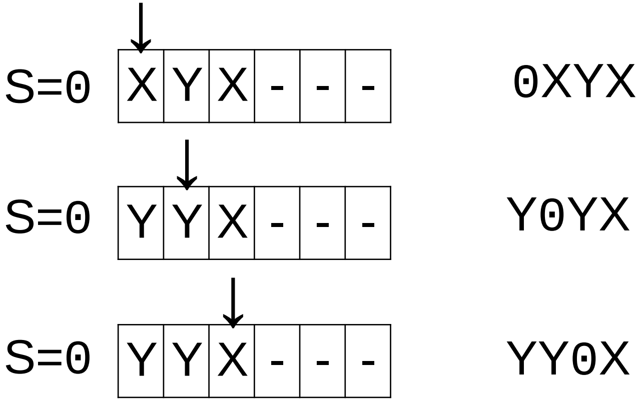
Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.



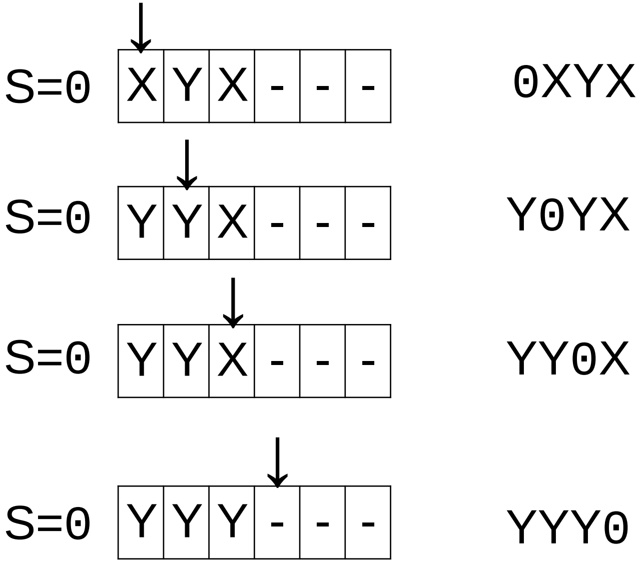
Notazione compatta per le configurazioni

Assumendo che l'insieme degli stati e l'alfabeto siano disgiunti, una configurazione può essere rappresentata come

$$vSw$$

con

- *v* la parte di nastro che precede la testina,
- S lo stato corrente,
- *w* la parte di nastro a partire dalla testina.



Hardware

Secondo l’enciclopedia Treccani

Hardware. termine di uso internazionale per indicare la "parte fisica", non modificabile, di un calcolatore.

Hardware

Secondo l’enciclopedia Treccani

Hardware. termine di uso internazionale per indicare la "parte fisica", non modificabile, di un calcolatore.

Hardware di una macchina di Turing:

↓

X	Y	X	-	-	-	...

Stato 0

Regole di transizione

(0,X) → (0,Y,>)

(0,Y) → (0,Y,>)

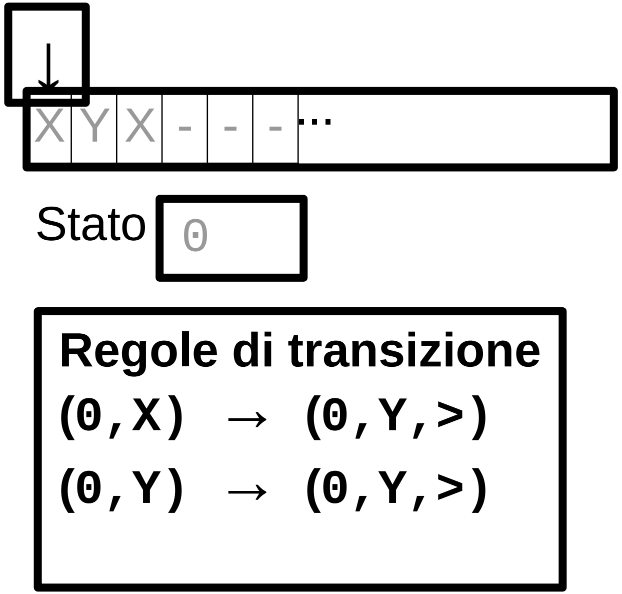
Hardware

Secondo l’enciclopedia Treccani

Hardware. termine di uso internazionale per indicare la "parte fisica", non modificabile, di un calcolatore.

Hardware di una macchina di Turing:

- nastro;
- testina;
- (contenitore per lo) stato corrente;
- Regole di transizione.



Hardware e dati

Hardware. Tutto quello che non può essere modificato durante la computazione: nastro, testina, contenitore per lo stato corrente, regole di transizione.

Dati. Quanto riportato nelle configurazioni.

configurazione iniziale

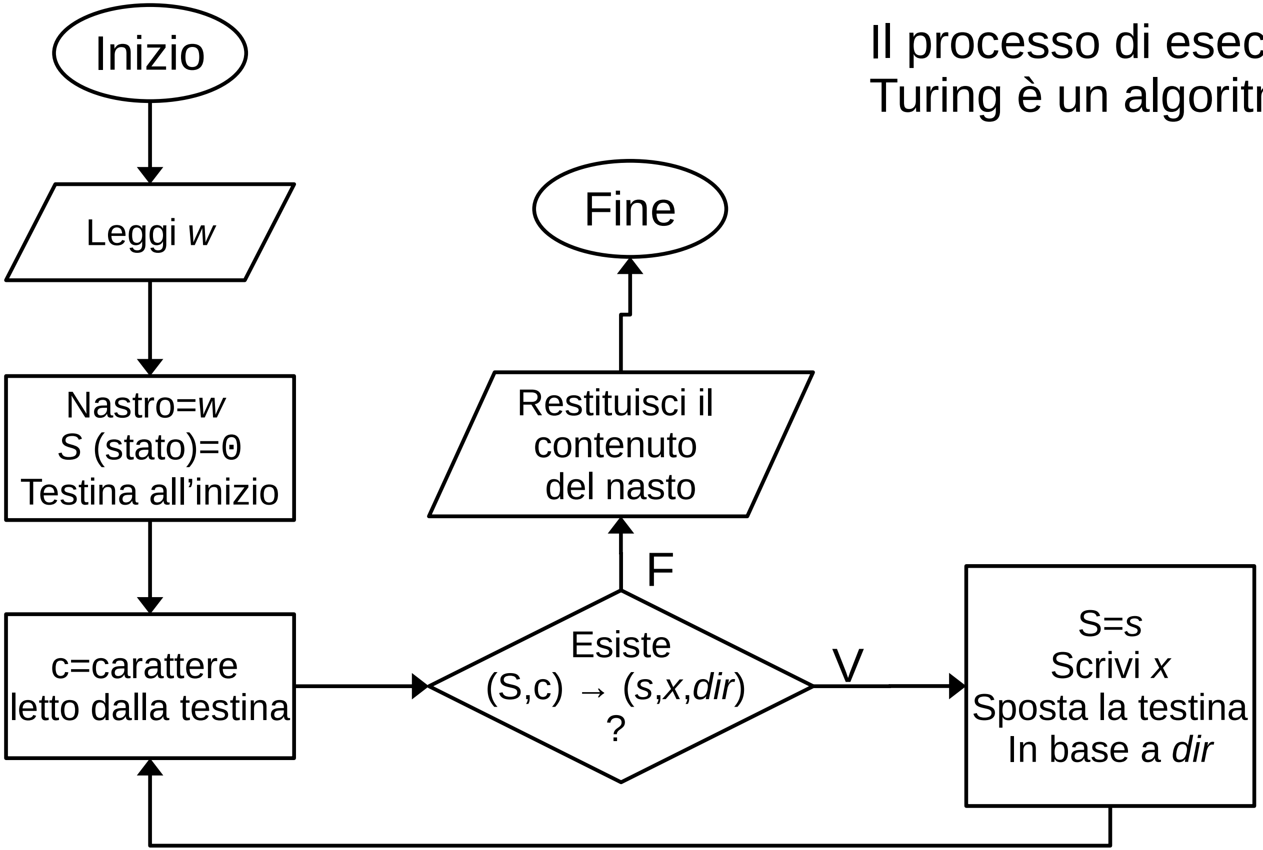
computazione

```
0XYX    dati in ingresso (input)
Y0YX
YY0X    dati intermedi
YY0Y
YYY0    risultato (output)
```

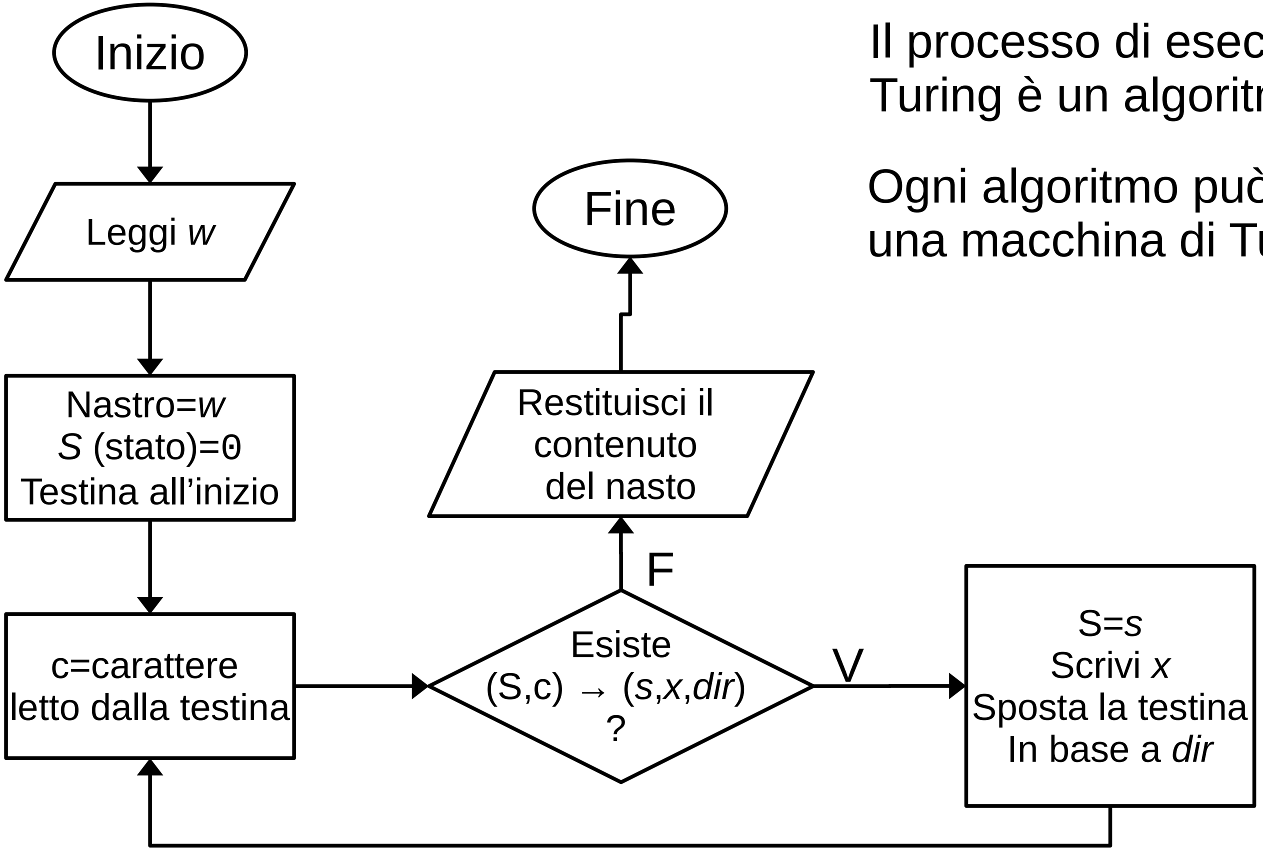
configurazione finale

La Macchina Universale - intuizione

Il processo di esecuzione di una macchina di Turing è un algoritmo.



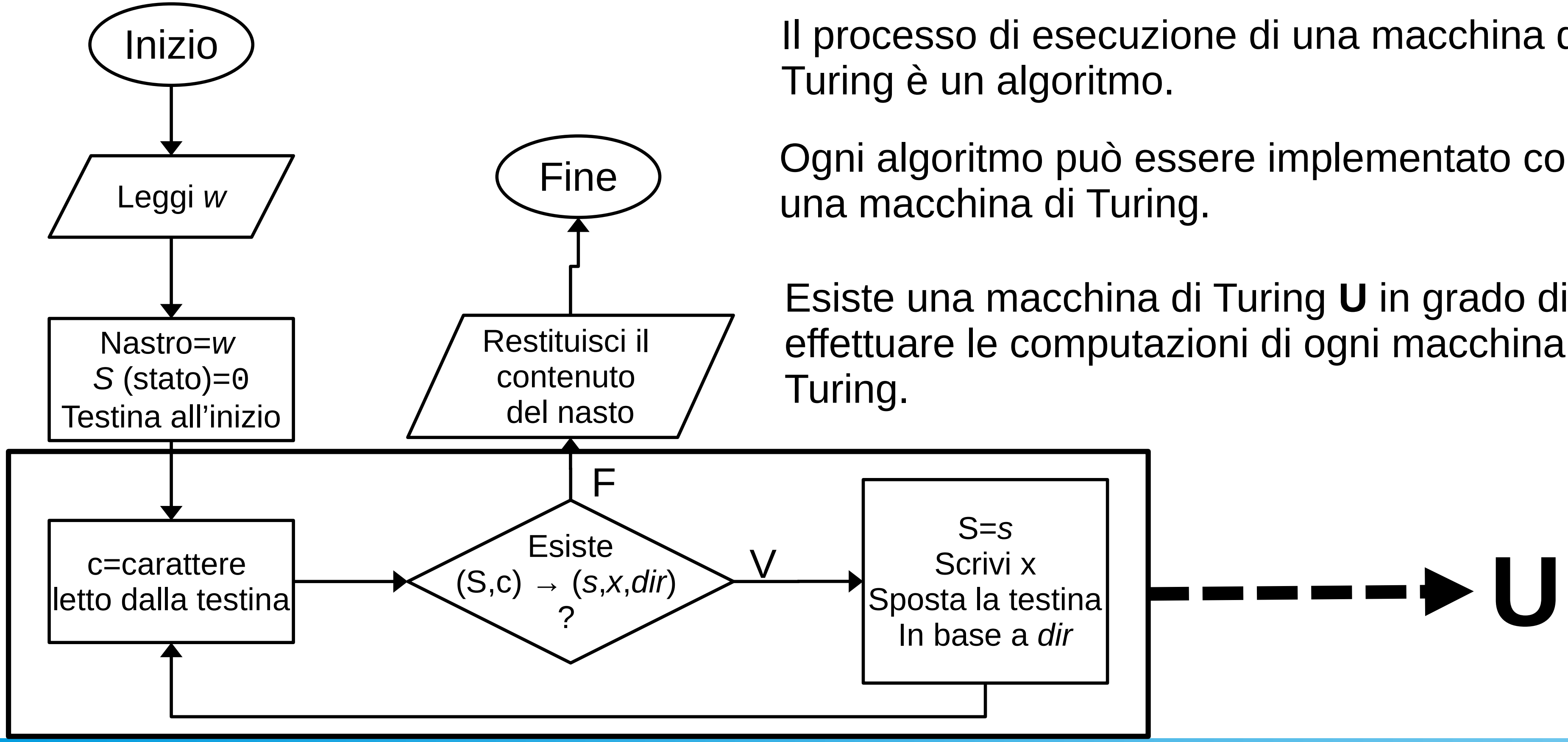
La Macchina Universale - intuizione



Il processo di esecuzione di una macchina di Turing è un algoritmo.

Ogni algoritmo può essere implementato con una macchina di Turing.

La Macchina Universale - intuizione



Il processo di esecuzione di una macchina di Turing è un algoritmo.

Ogni algoritmo può essere implementato con una macchina di Turing.

Esiste una macchina di Turing **U** in grado di effettuare le computazioni di ogni macchina di Turing.

La Macchina Universale - funzionalità

I dettagli di **U** saranno omessi in questa presentazione, ne verranno descritte solo le funzionalità come *Black Box*.

Dati di ingresso:

- una macchina di Turing (regole di transizione) *T*,
- una configurazione iniziale *c* per *T.*

Esempio (con abuso di notazione)

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

Risultato: il risultato della computazione di *T* a partire da *c.*

Esempio

Y	Y	Y	0

La Macchina Universale - esempio

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

La Macchina Universale - esempio

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	0	Y	X

La Macchina Universale - esempio

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	0	Y	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	Y	0	X

La Macchina Universale - esempio

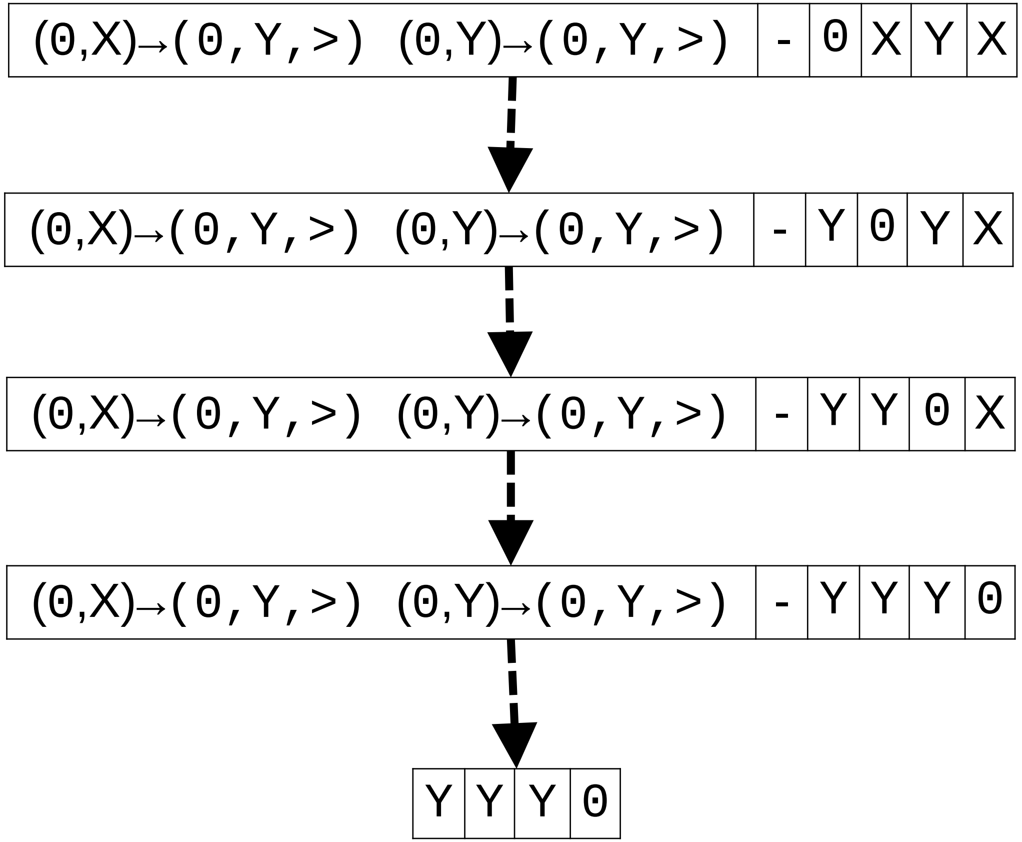
(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	0	Y	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	Y	0	X

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	Y	Y	Y	0

La Macchina Universale - esempio

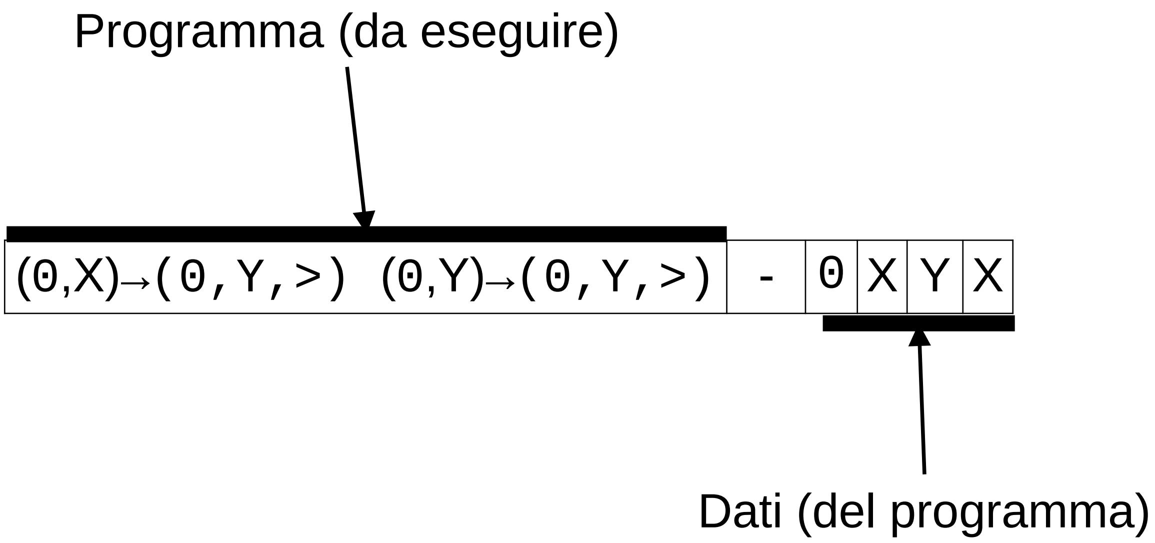


Programma e dati

Programma (da eseguire)

(0,X)→(0,Y,>) (0,Y)→(0,Y,>)	-	0	X	Y	X

Programma e dati



Computabilità

Teorema dell'Halt: non può esistere una macchina di Turing HALT che possa determinare, per ogni macchina di Turing *T* e per ogni configurazione iniziale *c*, se la computazione di *T* a partire da *c* termini oppure no.

Computabilità

Teorema dell'Halt: non può esistere una macchina di Turing HALT che possa determinare, per ogni macchina di Turing *T* e per ogni configurazione iniziale *c*, se la computazione di *T* a partire da *c* termini oppure no.

Corollario: esistono problemi **non computabili**, che quindi non ammettono una soluzione automatizzabile.

Il problema dell’HALT

Per assurdo,
supponiamo che esista una macchina di turing HALT con parametri di ingresso

- una macchina di Turing *T* e
- una configurazione iniziale *c*

Il problema dell'HALT

Per assurdo,
supponiamo che esista una macchina di turing HALT con parametri di ingresso

- una macchina di Turing *T* e
- una configurazione iniziale *c*

e restituisca

- V se la computazione di *T* a partire da *c* è finita (quindi la computazione termina),

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina

Il problema dell'HALT

Per assurdo,
supponiamo che esista una macchina di turing HALT con parametri di ingresso

- una macchina di Turing *T* e
- una configurazione iniziale *c*

e restituisca

- V se la computazione di *T* a partire da *c* è finita (quindi la computazione termina),
- F altrimenti.

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT(*T*,*c*)=F se e solo se *T*(*c*) è infinita.

Il problema dell’HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull’input *c*
HALT*(T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV una macchina di Turing che prenda come parametro una macchina di Turing *T* e una configurazione iniziale *c* e tale che

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(T,c)=V se e solo se T(c) termina sull'input c
HALT(T,c)=F se e solo se $T(c)$ è infinita.

Sia INV una macchina di Turing che prenda come parametro una macchina di Turing T e una configurazione iniziale c e tale che

- restituisca V se e solo se HALT(T,c)=F,

INV(T,c)=V se e solo se HALT(T,c)=F

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(T,c)=V se e solo se T(c) termina sull'input c
HALT(T,c)=F se e solo se $T(c)$ è infinita.

Sia INV una macchina di Turing che prenda come parametro una macchina di Turing T e tale che

- restituisca V se e solo se HALT(T,T)=F,
- entri in un ciclo infinito se HALT(T,T)=V.

INV(T)=V se e solo se HALT(T,T)=F
INV(T) non termina se e solo se HALT(T,T)=V

Il problema dell’HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull’input *c*
HALT*(T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV*(T*) non termina se e solo se HALT(*T*,*T*)=V

INV(INV) termina?

Caso 1: Si

Caso 2: No

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(T,c)=V se e solo se T(c) termina sull'input c
HALT(T,c)=F se e solo se T(c) è infinita.

Sia INV tale che

INV(T)=V se e solo se HALT(T,T)=F
INV(T) non termina se e solo se HALT(T,T)=V

INV(INV) termina?

Caso 1: Si

Caso 2: No

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(T,c)=V se e solo se T(c) termina sull'input c
HALT(T,c)=F se e solo se $T(c)$ è infinita.

Sia INV tale che

INV(T)=V se e solo se HALT(T,T)=F
INV(T) non termina se e solo se HALT(T,T)=V

INV(INV) termina?

Caso 1: Si → INV(INV)=V

Caso 2: No

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT(*T*,*c*)=F se e solo se *T(c)* è infinita (non termina).

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV(*T*) non termina se e solo se HALT(*T*,*T*)=V

INV(INV) termina?

Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F

Caso 2:

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT(*T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV(*T*) non termina se e solo se HALT(*T*,*T*)=V

Contraddizione!

INV(INV) termina?

Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina

Caso 2: No → HALT(INV,INV)=V → INV(INV) termina

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(T,c)=V se e solo se T(c) termina sull'input c
HALT(T,c)=F se e solo se $T(c)$ è infinita.

Sia INV tale che

INV(T)=V se e solo se HALT(T,T)=F
INV(T) non termina se e solo se HALT(T,T)=V

INV(INV) termina?

~~Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina~~

Caso 2: No

Il problema dell’HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

> **HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull’input *c***
> HALT(*T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

> INV(*T*)=V se e solo se HALT(*T*,*T*)=F
> INV(*T*) non termina se e solo se HALT(*T*,*T*)=V

INV(INV) termina?

~~Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina~~

Caso 2: No → HALT(INV,INV)=V

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT(*T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV(*T*) non termina se e solo se HALT(*T*,*T*)=V

INV(INV) termina?

~~Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina~~

Caso 2: No → HALT(INV,INV)=V → INV(INV) termina

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT(*T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV(*T*) non termina se e solo se HALT(*T*,*T*)=V

Contraddizione!

INV(INV) termina?

~~Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina~~

Caso 2: No → HALT(INV,INV)=V → INV(INV) termina

Il problema dell'HALT - dimostrazione

Per assurdo,
supponiamo che esista una macchina di turing HALT tale che

HALT(*T*,*c*)=V se e solo se *T*(*c*) termina sull'input *c*
HALT*(T*,*c*)=F se e solo se *T(c)* è infinita.

Sia INV tale che

INV(*T*)=V se e solo se HALT(*T*,*T*)=F
INV*(T*) non termina se e solo se HALT(*T*,*T*)=V

Assurdo

INV(INV) termina?

~~Caso 1: Si → INV(INV)=V → HALT(INV,INV)=F → INV(INV) non termina~~

~~Caso 2: No → HALT(INV,INV)=V → INV(INV) termina~~