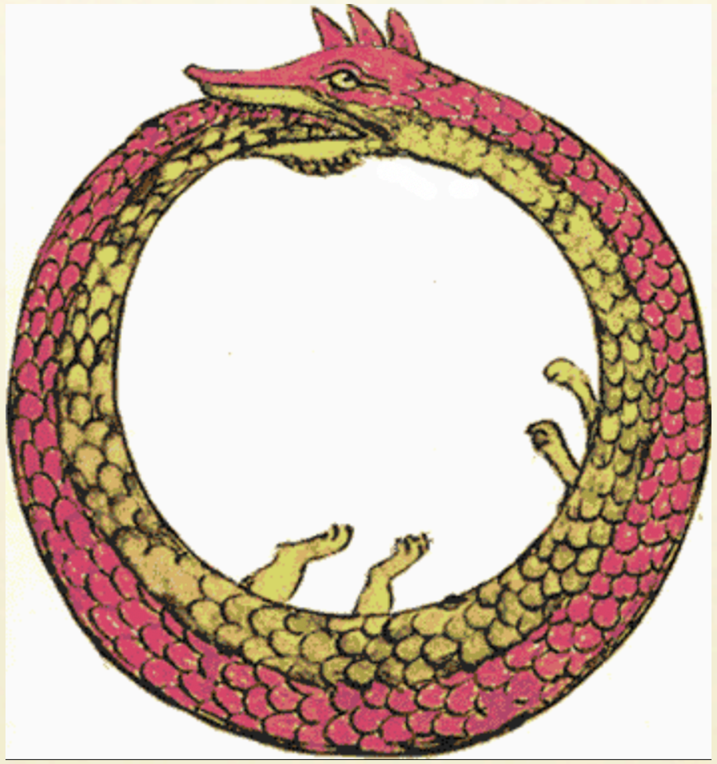
Informatica Teorica



Anno Accademico 2022/2023

Fabio Zanasi
https://www.unibo.it/sitoweb/fabio.zanasi

Sesta lezione

Nelle puntate precedenti

Abbiamo studiato cosa **sono in grado di fare** le macchine di Turing:

Decidere/riconoscere problemi (linguaggi)

Simulare altri modelli di calcolo, tra cui le macchine a registri e i linguaggi di programmazione di alto livello.

Essere ‘universali' e quindi programmabili.

In questa lezione

Cominciamo lo studio di cosa **non possono fare** le TM.

Introduciamo il nostro primo problema **indecidibile**:
il problema della **fermata** (*halting problem*).

Questo risultato ci informa, più in generale, sui **limiti della computazione per algoritmi.**

Ripasso: la tesi di Church-Turing

Se la soluzione di un dato problema può essere calcolata attraverso una procedura algoritmica, allora può essere calcolata da una macchina di Turing.

Conseguenze della tesi di Church-Turing

Visto la scorsa settimana

Calcolabile da una procedura algoritmica. → Calcolabile da una TM.

Ora:

Non calcolabile da una TM. → Non calcolabile da nessuna procedura algoritmica.

Non da un programma C++/Python/…
Non da un computer quantistico
Etc.

Linguaggi e TM: ripasso

Una TM $\mathcal{M}$ **decide** un linguaggio L se:

- Quando $x \in L$, allora $\mathcal{M}$ accetta x (= ferma nello stato Y).
- Quando $x \notin L$, allora $\mathcal{M}$ rigetta x (= ferma nello stato N).

Una $\mathcal{M}$ **riconosce** un linguaggio L se:

- Quando $x \in L$, allora $\mathcal{M}$ termina.
- Quando $x \notin L$, allora $\mathcal{M}$ non termina.

Gradi di (in)calcolabilità

Decidibile da una TM	~	Calcolabile (C'è un algoritmo che risponde "Si" o "No")
Non decidibile da nessuna TM ma riconoscibile da una qualche TM	~	Non calcolabile (Semi-decidibile: nessun algoritmo saprà calcolare tutte le risposte"No")
Non riconoscibile da nessuna TM	~	Non calcolabile (Del tutto non calcolabile: qualsiasi algoritmo fallirà nel dare sia le risposte "Si" che "No.)

Un problema non calcolabile

Dimostriamo l’esistenza di un problema del primo tipo: riconoscibile ma **non decidibile**.

Un problema indecidibile

Prima di tutto, supponiamo che esista una codifica code(-):

TM su alfabeto Σ. $\xrightarrow{\text{code(-)}}$ Stringhe $x \in \Sigma^*$.

La codifica usata per definire la macchina di Turing universale é un esempio di tale procedura.

Il problema della fermata

Definiamo il linguaggio del **problema della fermata**:

$HALT = \{ \langle y, x \rangle \in \Sigma^* \mathrm{x}\ \Sigma^* \mid y = \mathrm{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x. \}$

Teorema Il problema della fermata é riconoscibile
ma non é decidibile.

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \text{x } \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

La parte semplice: *HALT* é riconoscibile.

Esercizio: come costruiamo una TM che riconosca *HALT*?

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \text{x } \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

La parte rimanente: dimostriamo che *HALT* non é decidibile.

La dimostrazione sarà per contraddizione.

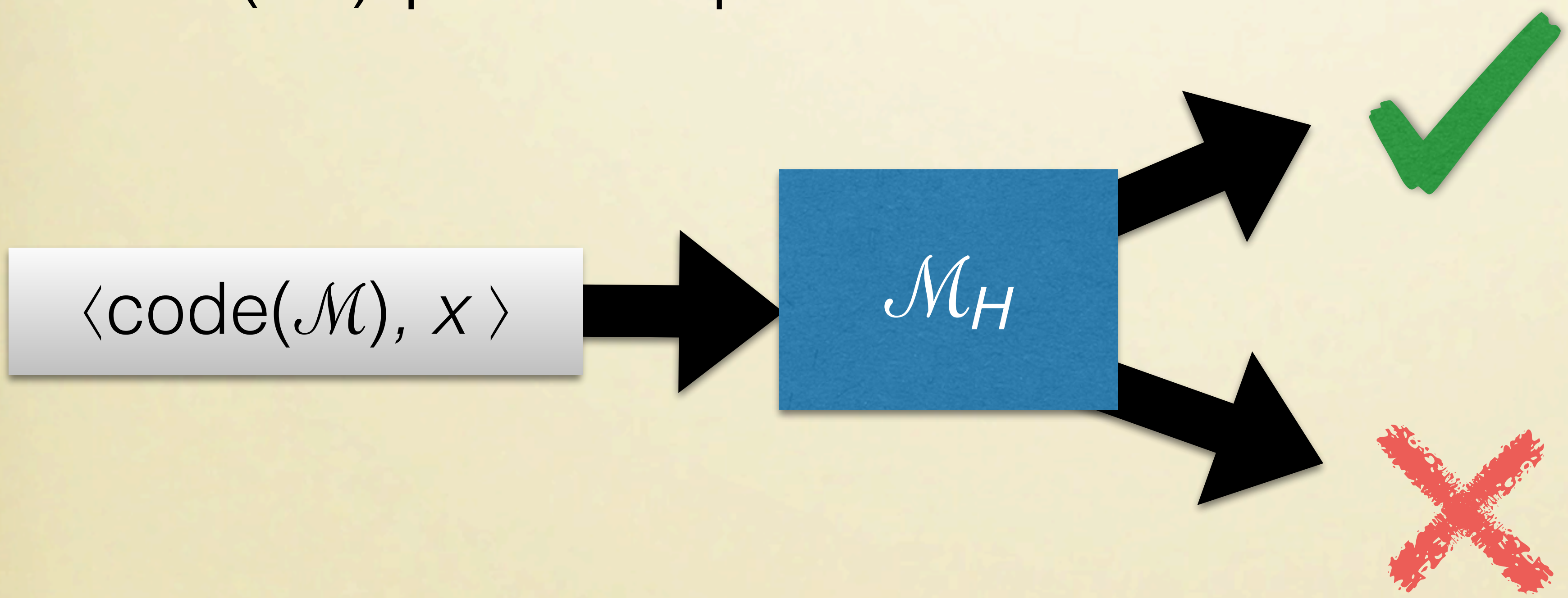
Perciò assumiamo che *HALT sia* decidibile, e chiamiamo $\mathcal{M}_H$ la TM che decide HALT.

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* x\ \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

Perciò $\mathcal{M}_H$ si comporterà come segue. Se y = code($\mathcal{M}$) per un qualche $\mathcal{M}$:



Se $\mathcal{M}$ ferma su x.

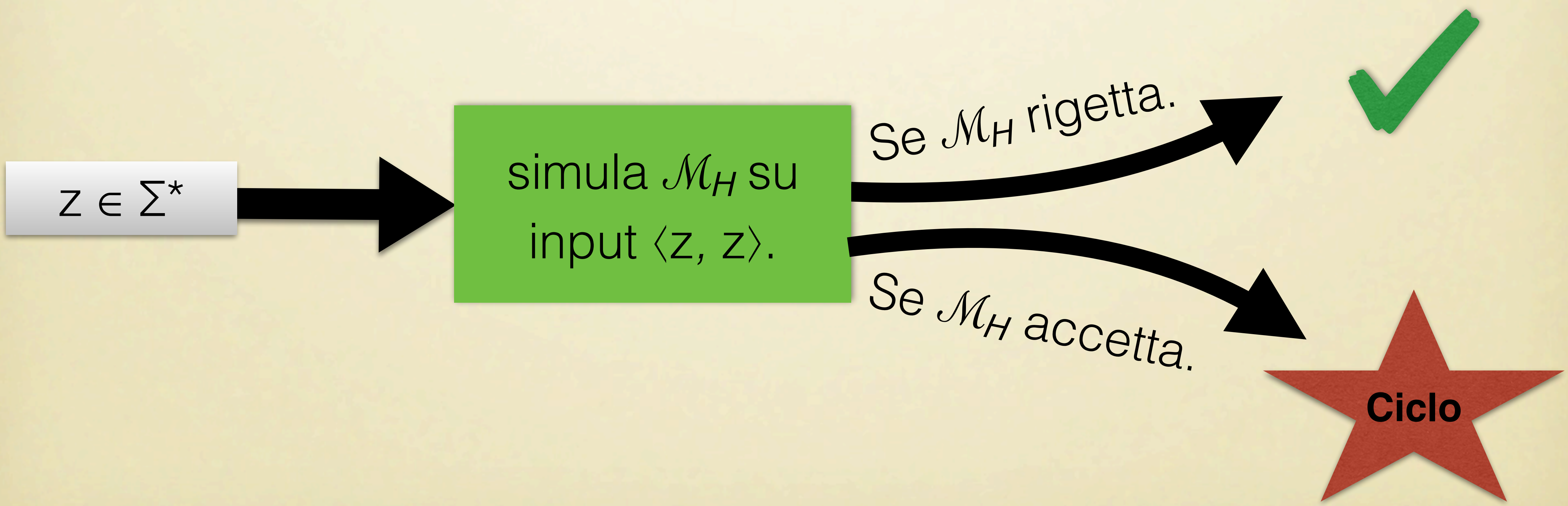
Se $\mathcal{M}$ non ferma su x.

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \times \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

Possiamo definire una nuova TM $\mathcal{M}'$ come segue.

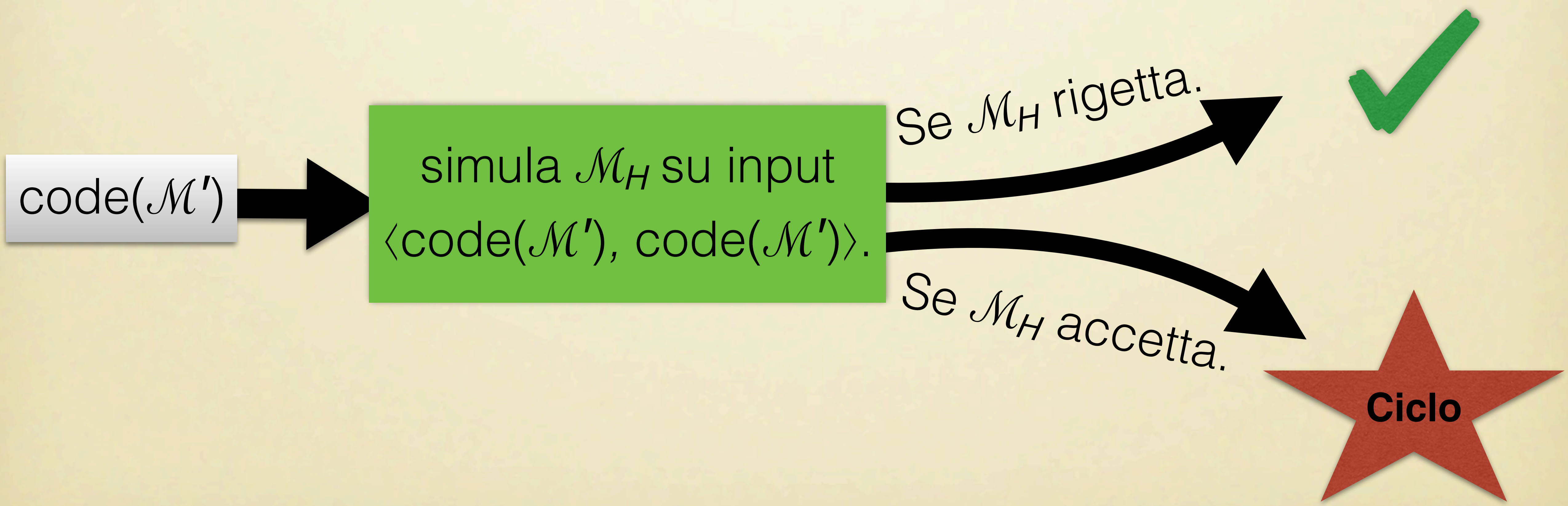


Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \text{x } \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

Proviamo ora ad eseguire $\mathcal{M}'$ su input code($\mathcal{M}'$).



Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \mathrm{x}\ \Sigma^* \mid y = \mathrm{code}(\mathcal{M})$ e $\mathcal{M}$ ferma su $x.\}$

Dimostrazione

$\mathcal{M}_H$ accetta $\langle \mathrm{code}(\mathcal{M}'), \mathrm{code}(\mathcal{M}')\rangle$.

per def di $\mathcal{M}'$ $\Downarrow$

$\mathcal{M}'$ non ferma su $\mathrm{code}(\mathcal{M}')$.

$\Uparrow$ per def di $HALT$

$\langle \mathrm{code}(\mathcal{M}'), \mathrm{code}(\mathcal{M}')\rangle \notin HALT$

$\Downarrow$ per def di $\mathcal{M}_H$

$\mathcal{M}_H$ non accetta $\langle \mathrm{code}(\mathcal{M}'), \mathrm{code}(\mathcal{M}')\rangle$.

Contraddizione. Proviamo l'altra opzione…

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* \times \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

$\mathcal{M}_H$ rigetta $\langle \text{code}(\mathcal{M}'), \text{code}(\mathcal{M}')\rangle$.

$\Downarrow$ per def di $\mathcal{M}'$

$\mathcal{M}'$ ferma su code($\mathcal{M}'$).

$\Uparrow$ per def di *HALT*

$\langle \text{code}(\mathcal{M}'), \text{code}(\mathcal{M}')\rangle \in HALT$

$\Downarrow$ per def di $\mathcal{M}_H$

$\mathcal{M}_H$ accetta $\langle \text{code}(\mathcal{M}'), \text{code}(\mathcal{M}')\rangle$.

Perciò, in entrambi i casi, abbiamo una contraddizione.

Il problema della fermata

$HALT = \{ \langle y, x \rangle \in \Sigma^* x\ \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

Dimostrazione

L'unica assunzione utilizzata nel costruire $\mathcal{M}'$ é che esista una TM $\mathcal{M}_H$ che decide *HALT*.

Perciò $\mathcal{M}_H$ non può esistere: *HALT* é indecidibile.

Gradi di (in)calcolabilità

Decidibile da una TM	∼	Calcolabile (C'è un algoritmo c… o")
Non decidibile da nessuna TM ma riconoscibile da una qualche TM	∼	Non calcolabile (Semi-decidibile: nessun algoritmo saprà calcolare tutte le ris…
Non riconoscibile da nessuna TM	∼	Non calcolabile (Del tutto non calcolabile: qualsiasi algoritmo fallirà nel dare sia le risposte "Si" che "No.)

Il problema della fermata

?

Problemi non riconoscibili

Teorema il complemento $HALT^-$ del problema della fermata non é riconoscibile da nessuna TM.

$HALT = \{ \langle y, x \rangle \in \Sigma^* x\ \Sigma^* \mid y = \text{code}(\mathcal{M}) \text{ e } \mathcal{M} \text{ ferma su } x.\}$

$HALT^- = \{ \langle y, x \rangle \in \Sigma^* x\ \Sigma^* \mid y \neq \text{code}(\mathcal{M}) \text{ per qualsiasi } \mathcal{M} \text{ oppure } y = \text{code}(\mathcal{M}) \text{ and } \mathcal{M} \text{ non ferma su } x.\}$

Il complemento di HALT

Teorema il complemento $HALT^{-}$ del problema della fermata non é riconoscibile da nessuna TM.

C'é una dimostrazione diretta, per contraddizione, ma é più interessante mostrare una dimostrazione più astratta. Deriva dal seguente teorema:

Teorema Se $HALT^{-}$ fosse riconoscibile, allora *HALT* sarebbe decidibile.

Infatti, dal momento che *HALT* é indecidibile, se questo teorema é vero allora $HALT^{-}$ non può essere riconoscibile.

Il complemento di HALT

Teorema Se $HALT^{-}$ fosse riconoscibile, allora *HALT* sarebbe decidibile.

Dimostrazione

Abbiamo già visto che *HALT* é riconoscibile, diciamo da una TM $\mathcal{M}_{HR}$.

Supponiamo per assurdo che anche $HALT^{-}$ sia riconoscibile, e chiamiamo $\mathcal{M}_{H-}$ la TM che lo riconosce.

Possiamo ora costruire una TM $\mathcal{M}_{H}$ che decide HALT come segue.

Il complemento di HALT

Dimostrazione Definizione di $\mathcal{M}_H$:

su input $\langle y, x \rangle$, simula $\mathcal{M}_{HR}$ e $\mathcal{M}_{H^-}$ *in parallelo* su input $\langle y, x \rangle$.

Se $\mathcal{M}_{HR}$ ferma, accetta. Se $\mathcal{M}_{H^-}$ ferma, rigetta.

$\langle y, x \rangle \in HALT$ $\Rightarrow$ $\mathcal{M}_{HR}$ ferma. $\Rightarrow$ $\mathcal{M}_H$ ferma e accetta

$\langle y, x \rangle \notin HALT$

$\Downarrow$

$\langle y, x \rangle \in HALT^-$ $\Rightarrow$ $\mathcal{M}_{H^-}$ ferma. $\Rightarrow$ $\mathcal{M}_H$ ferma e rigetta

Perciò $\mathcal{M}_H$ decide *HALT*.

Il complemento di HALT

Perciò abbiamo dimostrato:

Teorema Se $HALT^-$ fosse riconoscibile, allora *HALT* sarebbe decidibile.

In conclusione, dal momento che *HALT* é indecidibile, allora $HALT^-$ non può essere riconoscibile.

Gradi di (in)calcolabilità

Decidibile da una TM ~ Calcolabile

(C'è un algoritmo che … o")

Il problema della fermata

Non decidibile da nessuna TM ma riconoscibile da una qualche TM ~ Non calcolabile

(Semi-decidibile: n… rà

calcolare tu…

Il complemento del problema della fermata

Non riconoscibile da nessuna TM ~ Non calcolabile

(Del tutto non calcolabile: qualsiasi algoritmo fallirà nel dare sia le risposte "Si" che "No.)

Il risultato su $HALT^-$ suggerisce due ulteriori osservazioni.

Osservazione #1
Complemento di un linguaggio riconoscibile

La dimostrazione data non sfrutta in alcun modo il fatto che HALT sia definito nel modo in cui é definito: potremmo sostituire HALT con un qualsiasi problema riconoscibile, e funzionerebbe lo stesso. Abbiamo dunque:

Teorema Se L e $\bar{L}$ sono riconoscibili, allora L é decidibile.

Dimostrazione La stessa data per $L = HALT$.

Osservazione #1
Complemento di un linguaggio riconoscibile

Usando

Teorema Se L e $\bar{L}$ sono riconoscibili, allora L é decidibile.

Teorema *HALT* é riconoscibile ma non decidibile.

Otteniamo:

Corollario I linguaggi riconoscibili *non* sono chiusi sotto complemento.

Dimostrazione *HALT* é riconoscibile ma il suo complemento non é riconoscibile.

Osservazione #2
Ridurre un problema ad un altro

La nostra dimostrazione del fatto che *HALT* $^{-}$ non sia riconoscibile ha la seguente struttura:

Se potessimo riconoscere *L*, *allora* potremmo decidere *L'*.
Poiché L' non é decidibile, *allora* non possiamo riconoscere L.

La parte veramente 'nuova' della dimostrazione é la prima: come *ridurre* L a L'. Nella prossima lezione vedremo come questa intuizione possa essere formalizzata in una tecnica di dimostrazione, che ci permette di ridurre problemi tra di loro al fine di dimostrarne la non calcolabilità.