DECIDIBILITÀ E INDECIDIBILITÀ

Indecidibilità
(Capitolo IV, sezione 4.2, seconda parte)

Linguaggi indecidibili

Il metodo della diagonalizzazione ci ha permesso di dimostrare che esistono linguaggi non Turing riconoscibili.

La prova non era costruttiva. Vorremmo poter esibire un particolare linguaggio che non sia Turing riconoscibile.

Esibiremo prima un particolare linguaggio, associato a un problema di decisione, che è indecidibile.

Nella prova viene utilizzato il metodo della diagonalizzazione e l'autoreferenzialità.

Autoreferenzialità e paradosso di Russel

- Consideriamo i seguenti insiemi:

$$
\begin{aligned}
A &= \text{l'insieme di tutti gli insiemi finiti} \\
B &= \text{l'insieme di tutti gli insiemi infiniti} \\
C &= \text{l'insieme di tutti gli insiemi} \\
&\quad\ \text{che non sono elementi di se stessi}
\end{aligned}
$$

Domande:
$A \in A$? NO
$B \in B$? SÌ
$A \in C$? SÌ
$B \in C$? NO
$C \in C$? ?

Autoreferenzialità e paradosso di Russel

In un paese vive un solo barbiere, un uomo ben sbarbato, che rade tutti e soli gli uomini del villaggio che non si radono da soli.
Chi sbarba il barbiere?

Un linguaggio indecidibile

Teorema

Il linguaggio

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una macchina di Turing e } M \text{ accetta la parola } w\}$$

è Turing riconoscibile ma non è decidibile.

Macchina di Turing Universale

- Una macchina di Turing universale U riceve in input una **codifica** $\langle M, w \rangle$ di una macchina di Turing M e di una stringa w.
- U simula la computazione di M sull'input w.
- Anticipò alcuni sviluppi fondamentali in informatica:
  - Compilatore Java (o C, C^{++}) in Java (o in C, C^{++})
  - Sviluppo di computer a programma memorizzato

Teorema

Esiste una TM universale.

(Schema del comportamento di U)

$$\langle M, w \rangle \rightarrow \boxed{\text{MT Universale } U} \rightarrow \begin{cases} \text{accetta} & \text{se } M \text{ accetta } w \\ \text{rifiuta} & \text{se } M \text{ rifiuta } w \\ \text{non termina} & \text{se } M \text{ non termina} \end{cases}$$

Macchina di Turing Universale

Ma com'è fatta la macchina di Turing Universale?

Possiamo pensare a una macchina di Turing U a tre nastri.

La macchina U riceve in input la codifica $\langle M, w \rangle$ di M e w.

Prima di "eseguire" M su w, U esegue alcuni passi di inizializzazione:

1. copia sul secondo nastro la codifica di M,
2. copia sul terzo nastro la codifica dello stato iniziale di M,
3. lascia sul primo nastro la codifica di w.

Macchina di Turing Universale

Durante la sua computazione U:

- usa il primo nastro per simulare la computazione di M,
- lascia sul secondo nastro la codifica di M,
- ha sul terzo nastro la codifica dello stato corrente di M.

In parole più informatiche,

- il secondo nastro rappresenta il programma da eseguire,
- il primo nastro rappresenta i dati da elaborare,
- il terzo è una componente del contatore istruzioni (l'altra sarà il simbolo corrente).

U individua l'istruzione corrente sul secondo nastro, usando il contenuto del terzo nastro e il simbolo corrente (codificato) sul primo nastro, quindi decodifica l'istruzione e la esegue.

A_{TM} è Turing riconoscibile

Teorema

Il linguaggio

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una macchina di Turing e } M \text{ accetta la parola } w\}$$

è Turing riconoscibile.

A_{TM} è Turing riconoscibile

Dimostrazione.

La seguente macchina di Turing U riconosce A_{TM}.

$U =$ "Sull'input $\langle M, w \rangle$ dove M è una TM e w è una stringa

1. Simula M sull'input w.
2. Se M accetta w, accetta l'input $\langle M, w \rangle$; se M rifiuta w, rifiuta l'input $\langle M, w \rangle$."

U rifiuta ogni stringa che non sia della forma $\langle M, w \rangle$ dove M è una TM e w è una stringa.

Quindi U accetta una stringa y se e solo se y è della forma $\langle M, w \rangle$ dove M è una TM, w è una stringa e M accetta w.

In altri termini, U accetta una stringa y se e solo se $y = \langle M, w \rangle$ è un elemento di A_{TM}.

Ne segue $L(U) = A_{TM}$.

A_{TM} è Turing riconoscibile

- **Nota.** La macchina di Turing U nella prova del teorema precedente esiste ed è la Macchina di Turing Universale.

- **Nota:** U non termina su $\langle M, w \rangle$ se (e solo se) M non termina su w. Quindi U **non decide** A_{TM}.

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una macchina di Turing e } M \text{ accetta la parola } w\}$$

(Linguaggio associato al problema di decisione dell'accettazione di una macchina di Turing)

Teorema

Il linguaggio A_{TM} non è decidibile.

- **Nota.** Nella dimostrazione che segue, dato un decider S e una stringa w, $S(w)$ denota il risultato (accetta o rifiuta) della computazione di S sull'input w.

Dimostrazione

Supponiamo per assurdo che A_{TM} sia decidibile. Quindi supponiamo che esista un decisore H che riconosca A_{TM}.

Allora H accetta le stringhe di A_{TM} e rifiuta le stringhe che non sono in A_{TM}.

In particolare H accetta $\langle M, w\rangle$ se $\langle M, w\rangle \in A_{TM}$ e rifiuta $\langle M, w\rangle$ se $\langle M, w\rangle \notin A_{TM}$. Quindi

$$H(\langle M, w\rangle) = \begin{cases} \textit{accetta} & \text{se } M \text{ accetta } w \\ \textit{rifiuta} & \text{se } M \text{ non accetta } w \end{cases}$$

H rifiuta le stringhe y che non sono della forma $\langle M, w\rangle$, con M macchina di Turing e w stringa.

Costruiamo una nuova TM D che usa H come sottoprogramma. Questa nuova macchina di Turing D chiama H su $\langle M, \langle M \rangle \rangle$.

H accetta se M accetta $\langle M \rangle$ e rifiuta se M non accetta $\langle M \rangle$.

Una volta che D ha determinato questa informazione, D fa il contrario. Cioè, rifiuta se M accetta ed accetta se M non accetta.

(Schema del comportamento di D)

$$\langle M \rangle \rightarrow \boxed{\boxed{C} \rightarrow \langle M, \langle M \rangle \rangle \rightarrow \boxed{H} \rightarrow \begin{cases} accetta \\ rifiuta \end{cases} \rightarrow} \begin{cases} rifiuta \\ accetta \end{cases}$$

Diamo una descrizione di D.

$D =$ "Sull'input $\langle M \rangle$, dove M è una TM

1. Simula H quando H riceve in input $\langle M, \langle M \rangle \rangle$
2. Fornisce come output l'opposto di H, cioè se H accetta $\langle M, \langle M \rangle \rangle$, *rifiuta* e se H rifiuta $\langle M, \langle M \rangle \rangle$, *accetta*"

Ricapitolando,

$$D(\langle M \rangle) = \begin{cases} \textit{rifiuta} & \text{se } M \text{ accetta } \langle M \rangle, \\ \textit{accetta} & \text{se } M \text{ non accetta } \langle M \rangle \end{cases}$$

$$D(\langle M \rangle) = \begin{cases} \textit{rifiuta} & \text{se } M \text{ accetta } \langle M \rangle, \\ \textit{accetta} & \text{se } M \text{ non accetta } \langle M \rangle \end{cases}$$

Ora se diamo in input a D la sua stessa codifica $\langle D \rangle$ abbiamo

$$D(\langle D \rangle) = \begin{cases} \textit{rifiuta} & \text{se } D \text{ accetta } \langle D \rangle, \\ \textit{accetta} & \text{se } D \text{ non accetta } \langle D \rangle \end{cases}$$

Cioè D accetta $\langle D \rangle$ se e solo se D non accetta $\langle D \rangle$, il che è ovviamente una contraddizione. Quindi D non può esistere. Siccome D può facilmente essere costruita a partire da H, nemmeno H può esistere.

Nella prova precedente è stato utilizzato il metodo della diagonalizzazione. Ma dove si usa tale metodo?

Riesaminiamo il comportamento delle macchine H e D.

Indecidibilità di A_{TM}

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accetta*		*accetta*		
M_2	*accetta*	*accetta*	*accetta*	*accetta*	
M_3					$\cdots$
M_4	*accetta*	*accetta*			
$\vdots$		$\vdots$			

FIGURA 4.19
L'entrata i, j è *accetta* se M_i accetta $\langle M_j \rangle$

Figura tratta da M. Sipser, Introduzione alla teoria della computazione.

Indecidibilità di A_{TM}

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$
M_1	*accetta*	*rifiuta*	*accetta*	*rifiuta*	
M_2	*accetta*	*accetta*	*accetta*	*accetta*	$\cdots$
M_3	*rifiuta*	*rifiuta*	*rifiuta*	*rifiuta*	
M_4	*accetta*	*accetta*	*rifiuta*	*rifiuta*	
$\vdots$			$\vdots$		

FIGURA 4.20
L'entrata i, j è il valore di H su input $\langle M_i, \langle M_j \rangle \rangle$

Figura tratta da M. Sipser, Introduzione alla teoria della computazione.

Indecidibilità di A_{TM}

	$\langle M_1 \rangle$	$\langle M_2 \rangle$	$\langle M_3 \rangle$	$\langle M_4 \rangle$	$\cdots$	$\langle D \rangle$	$\cdots$
M_1	*accetta*	*rifiuta*	*accetta*	*rifiuta*		*accetta*	
M_2	*accetta*	*accetta*	*accetta*	*accetta*	$\cdots$	*accetta*	$\cdots$
M_3	*rifiuta*	*rifiuta*	*rifiuta*	*rifiuta*		*rifiuta*	
M_4	*accetta*	*accetta*	*rifiuta*	*rifiuta*		*accetta*	
$\vdots$			$\vdots$		$\ddots$		
D	*rifiuta*	*rifiuta*	*accetta*	*accetta*		?	
$\vdots$			$\vdots$				$\ddots$

FIGURA 4.21
Se D compare nella figura, si ha una contraddizione in corrispondenza di "?"

Figura tratta da M. Sipser, Introduzione alla teoria della computazione.

- In conclusione, A_{TM} è Turing riconoscibile ma non è decidibile.
- Che differenza c'è tra le due dimostrazioni?
- Utilizzando il metodo della diagonalizzazione, abbiamo provato che esistono linguaggi che non sono Turing riconoscibili. Ma ancora non abbiamo visto un esempio di un tale linguaggio.

Linguaggi co-Turing riconoscibili

Definizione

Diciamo che un linguaggio L è co-Turing riconoscibile se $\overline{L}$ è Turing riconoscibile.

Quindi un linguaggio co-Turing riconoscibile è il complemento di un linguaggio Turing-riconoscibile.

Proprietà di chiusura

Esercizio.
La classe dei linguaggi Turing-riconoscibili è chiusa rispetto al complemento?

Esercizio.
La classe dei linguaggi decidibili è chiusa rispetto al complemento?

Esercizio.
La classe dei linguaggi Turing-riconoscibili è chiusa rispetto all'unione? E rispetto all'intersezione?

Esercizio.
La classe dei linguaggi decidibili è chiusa rispetto all'unione? E rispetto all'intersezione?

Una proprietà dei linguaggi decidibili

Esercizio.
La classe dei linguaggi decidibili è chiusa rispetto al complemento.

Soluzione:

La classe dei linguaggi decidibili è chiusa rispetto al complemento. Sia A un linguaggio decidibile, sia M_A una macchina di Turing che decide A.

Definiamo la macchina di Turing $M_{\overline{A}}$: sull'input w, $M_{\overline{A}}$ simula M_A e accetta w se e solo se M_A rifiuta w.

Poiché M_A si arresta su ogni input anche $M_{\overline{A}}$ si arresta su ogni input.

Inoltre, il linguaggio di $M_{\overline{A}}$ è $\overline{A}$ perché $M_{\overline{A}}$ accetta w se e solo se M_A rifiuta w e quindi se e solo se $w \notin A$.

Quindi $M_{\overline{A}}$ è una macchina di Turing che decide $\overline{A}$ e $\overline{A}$ è decidibile.

Soluzione:

Formalmente, se

$$M_A = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$$

definiamo

$$M_{\overline{A}} = (Q, \Sigma, \Gamma, \delta', q_0, q_{accept}, q_{reject})$$

dove, per ogni $q \in Q \setminus \{q_{accept}, q_{reject}\}$, per ogni $\gamma \in \Gamma$

$$\delta'(q,\gamma) = \begin{cases} \delta(q,\gamma) & \text{se } \delta(q,\gamma) = (q',\gamma',d), \\ & \text{con } q' \notin \{q_{accept}, q_{reject}\}, \\ (q_{accept},\gamma',d) & \text{se } \delta(q,\gamma) = (q_{reject},\gamma',d), \\ (q_{reject},\gamma',d) & \text{se } \delta(q,\gamma) = (q_{accept},\gamma',d) \end{cases}$$

Una proprietà dei linguaggi decidibili

Teorema

Un linguaggio L è decidibile se e solo se L è Turing riconoscibile **e** *co-Turing riconoscibile.*

Dimostrazione

Dobbiamo provare che L è decidibile $\Leftrightarrow$ L e il suo complemento sono entrambi Turing riconoscibili.

($\Rightarrow$) Se L è decidibile allora esiste un decider, cioè una macchina di Turing M con due possibili risultati di una computazione (accettazione, rifiuto), tale che M accetta w se e solo se $w \in L$. In particolare, M riconosce L e quindi L è Turing riconoscibile. Inoltre abbiamo provato che anche il complemento $\overline{L}$ di L è decidibile. Con lo stesso ragionamento concludiamo che $\overline{L}$ è Turing riconoscibile.

Una proprietà dei linguaggi decidibili

($\Leftarrow$) Supponiamo che L e il suo complemento siano entrambi Turing riconoscibili. Sia M_1 una TM che riconosce L e M_2 una TM che riconosce $\overline{L}$. Definiamo una TM M.

$M =$ "Su input w:

1. Esegue sia M_1 che M_2 su input w in parallelo.
2. Se M_1 accetta, *accetta*; se M_2 accetta, *rifiuta*."

Una proprietà dei linguaggi decidibili

M è una macchina di Turing a due nastri.

M, dopo aver copiato w sul secondo nastro, simula M_1 sul primo nastro e M_2 sul secondo nastro.

Quindi M alterna la simulazione di un passo di M_1 con un passo di M_2 e continua finché una delle due accetta.

Una proprietà dei linguaggi decidibili

Vogliamo provare che M decide L.

Dobbiamo provare due cose:

- M è un decisore
- M riconosce L, cioè $L = L(M)$.

M è un decisore. Infatti, per ogni stringa w abbiamo due casi: o w è in L, oppure w è in $\overline{L}$. Pertanto una tra M_1 ed M_2 deve accettare w. Poiché M si ferma ogni volta che M_1 accetta oppure M_2 accetta, allora M si ferma sempre. Quindi M è un decisore.

Ora dobbiamo provare che $L(M) = L$.

1. $w \in L$. Ma $w \in L$ se e solo se M_1 accetta w. Quindi M accetta w.
2. $w \notin L$. Ma $w \notin L$ se e solo se M_2 accetta w. Quindi M rifiuta w.

Siccome M accetta w se e solo se $w \in L$ possiamo concludere che $L(M) = L$.

Nota. Nella prova precedente dovevamo dimostrare che è vero:

$$w \in L \quad \Leftrightarrow \quad M \text{ accetta } w$$

Cioè che è vero:

$$w \in L \quad \Rightarrow \quad M \text{ accetta } w$$

$$M \text{ accetta } w \quad \Rightarrow \quad w \in L$$

Invece abbiamo dimostrato che è vero:

$$w \in L \quad \Rightarrow \quad M \text{ accetta } w$$

$$w \notin L \quad \Rightarrow \quad M \text{ non accetta } w$$

Ma la proposizione

$$w \notin L \quad \Rightarrow \quad M \text{ non accetta } w$$

è logicamente equivalente a

$$M \text{ accetta } w \quad \Rightarrow \quad w \in L$$

Un linguaggio che non è Turing riconoscibile

Teorema

$\overline{A_{TM}}$ non è Turing riconoscibile.

Dimostrazione.

Supponiamo per assurdo che $\overline{A_{TM}}$ sia Turing riconoscibile.
Sappiamo che A_{TM} è Turing riconoscibile.
Quindi A_{TM} sarebbe Turing riconoscibile e co-Turing riconoscibile.
Per il precedente teorema, A_{TM} sarebbe decidibile.
Assurdo, poichè abbiamo dimostrato che A_{TM} è indecidibile.

□

Esercizio.
La classe dei linguaggi Turing-riconoscibili è chiusa rispetto al complemento?