decidibilità secondo Turing

obiettivo

- dimostrare che alcuni problemi sono algoritmicamente risolvibili ed altri no
  - del primo fatto abbiamo ampia evidenza dalla esperienza di informatici

problemi indecidibili

- un linguaggio è Turing-indecidibile (o semplicemente indecidibile) quando non esiste una MT che lo decida
- analoghe definizioni per la risolvibilità di problemi e per la calcolabilità di funzioni

il problema della fermata

- vogliamo realizzare una funzionalità di base per un debugger
- vogliamo scrivere del software che, dato un programma e dei dati di input, stabilisca se il programma termina su quei dati

un primo problema indecidibile

- molto simile al problema della fermata
- problema A_{TM} (linguaggio A_{TM})
  - A_{TM} = {<M,w> | M e' una MT che accetta la stringa w}
- in primo luogo osserviamo che A_{TM} è Turing-riconoscibile
  - sottoponiamo M e w alla MT universale U
  - se prima o poi U arriva ad uno stato accettante, accettiamo la stringa

il problema A_{TM} è indecidibile

- teorema: il linguaggio A_{TM} = {<M,w> | M è una MT che accetta la stringa w} è indecidibile
- dimostrazione: per assurdo, supponiamo che esista una macchina che decide A_{TM} ed otteniamo una contraddizione

il problema A_{TM} è indecidibile dimostrazione

- sia H una MT che decide A_{TM}
  H(<M,w>) = accetta se M accetta w
  = rifiuta se M non accetta w
- costruiamo un'altra MT H' che usa H come subroutine
- H' usa H per stabilire cosa fa M quando riceve come input M stessa
- H' valuta H(<M,M>)

il problema A_{TM} è indecidibile dimostrazione

- se H esiste, allora esiste anche H', infatti H' deve solo copiare M (usa una macchina C che fa la copia) e far partire H
- H'(M) = accetta se M accetta M
  = rifiuta se M non accetta M
- costruiamo ora un'altra macchina D che usa H' come subroutine
- D fa partire H' e restituisce l'opposto del risultato: se H' restituisce accetta D restituisce rifiuta e viceversa
- se H' esiste, allora esiste anche D (usa una macchina E che calcola l'opposto)

il problema A_{TM} è indecidibile dimostrazione

riassumendo:

- H(<M,w>) = accetta se M accetta w
  = rifiuta se M non accetta w
- H'(M) = accetta se M accetta M
  = rifiuta se M non accetta M
- D(M) = accetta se M non accetta M
  = rifiuta se M accetta M

il problema A_{TM} è indecidibile dimostrazione

- cosa succede se a D diamo in input D?

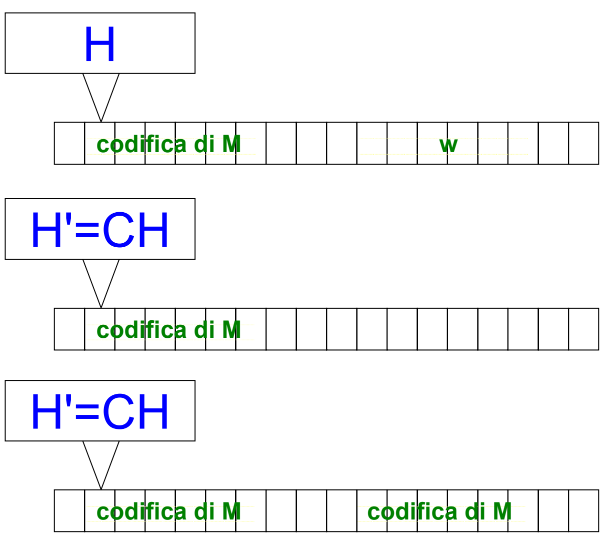
il problema A_{TM} è indecidibile dimostrazione

- H(<M,w>) = accetta se M accetta w
  = rifiuta se M non accetta w
- H'(M) = accetta se M accetta M
  = rifiuta se M non accetta M
- D(M) = accetta se M non accetta M
  = rifiuta se M accetta M
- D(D) = accetta se D non accetta D
  = rifiuta se D accetta D

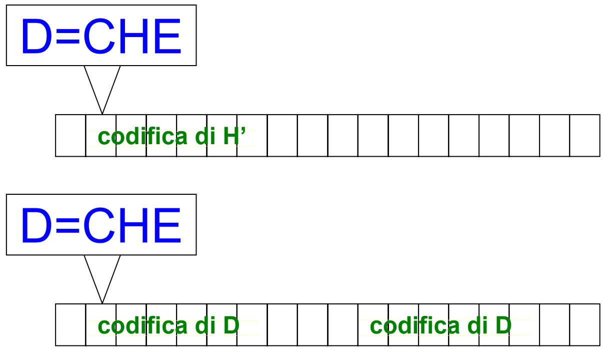
il problema A_{TM} è indecidibile dimostrazione

- qualunque cosa faccia D in realtà fa l'opposto: abbiamo un assurdo

rivediamolo alla moviola



rivediamolo alla moviola



- D(D) = accetta se D non accetta D
  = rifiuta se D accetta D

rapporto tra l'indecidibilità di A_{TM} e la diagonalizzazione

- è possibile osservare una relazione tra la tecnica di dimostrazione appena usata e il metodo di diagonalizzazione di Cantor

richiamo sulla diagonalizzazione

- dimostrazione del fatto che l'intervallo aperto di reali (0,1) non è numerabile
- supponiamo per assurdo che una enumerazione di (0,1) esista, denotiamo con Φ_i l'iesimo elemento di (0,1)
- consideriamo $r \in (0,1)$ che ha come i-esima cifra della mantissa (i=1, 2, …) un valore diverso da 0, da 9, e dal valore della i-esima cifra di Φ_i

richiamo sulla diagonalizzazione

cifre delle mantisse di Φ_i :

	1	**2**	**3**	**4**	**5**	**6**	**7**	**…**
Φ_1	5	1	0	4	3	9	6	…
Φ_2	2	4	1	0	0	0	0	…
Φ_3	7	9	8	5	3	7	7	…
Φ_4	0	0	4	6	0	3	1	…

r	6	5	1	7	…	…	…	…

r, detto *elemento diagonale*, non fa parte della enumerazione, in quanto differisce da ogni elemento della enumerazione in almeno una cifra, e ciò è assurdo

rapporto tra l'indecidibilità di A_{TM} e la diagonalizzazione

- elenchiamo tutte le MT sulle righe e sulle colonne di una tabella
- la posizione i,j è *accetta* se M_i accetta M_j ed è uno spazio se rigetta o cicla

	M_1	**M_2**	**M_3**	**…**
M_1	*accetta*		*accetta*	…
M_2	*accetta*	*accetta*	*accetta*	…
M_3	*accetta*			…
….	….			…

rapporto tra l'indecidibilità di A_{TM} e la diagonalizzazione

- nella tabella che segue è mostrato il comportamento di H quando ha in input gli elementi della tabella precedente

	M_1	**M_2**	**M_3**	**…**
M_1	*accetta*	*rifiuta*	*accetta*	…
M_2	*accetta*	*accetta*	*accetta*	…
M_3	*accetta*	*rifiuta*	*rifiuta*	…
….	….			…

rapporto tra l’indecidibilità di A_{TM} e la diagonalizzazione

- nella tabella possiamo aggiungere D, che essendo una MT, prima o poi appare nell’elenco

	M_1	M_2	M_3	…	D	…
M_1	*accetta*	*rifiuta*	*accetta*	…	*accetta*	
M_2	*accetta*	*accetta*	*accetta*	…	*rifiuta*	
M_3	*accetta*	*rifiuta*	*rifiuta*	…	*accetta*	
….	….			…		
D	*accetta*	*accetta*	*rifiuta*	…		
….	….			…		

rapporto tra l’indecidibilità di A_{TM} e la diagonalizzazione

- ma D calcola esattamente l’opposto di quanto appaia sulla diagonale
- l’elemento in posizione D,D deve essere l’opposto di se stesso

	M_1	M_2	M_3	…	D	…
M_1	*accetta*	*rifiuta*	*accetta*	…	*accetta*	
M_2	*accetta*	*accetta*	*accetta*	…	*rifiuta*	
M_3	*accetta*	*rifiuta*	*rifiuta*	…	*accetta*	
….	….			…		
D	*accetta*	*accetta*	*rifiuta*	…	**?**	
….	….			…		

calcolabilità secondo Turing

calcolabilità secondo Turing in vari contesti

decisione di predicati: un predicato su Σ^* è una funzione $p:(\Sigma^*)^n \rightarrow \{vero,falso\}$

p è Turing-decidibile se esiste una MT che calcola p, se non esiste nessuna MT allora p è Turing-indecidibile;

il predicato A_{TM} e' Turing-indecidibile

p è semi-decidibile se pur essendo indecidibile è Turing-riconoscibile

il predicato A_{TM} è semi-decidibile

un linguaggio che non è Turing-riconoscibile

- pur essendo indecidibile, il linguaggio A_{TM} è riconoscibile (è semi-decidibile)
- esistono linguaggi che non sono neppure riconoscibili
- diciamo che un linguaggio è co-riconoscibile se il suo complemento è riconoscibile

un linguaggio che non è Turing-riconoscibile

- teorema: un linguaggio è decidibile se e solo se è sia Turing-riconoscibile sia co-Turing-riconoscibile
- dimostrazione:
  - consideriamo il linguaggio A e supponiamo sia decidibile
  - se A è decidibile lo è anche il suo complemento
  - e se un linguaggio è decidibile è anche T-riconoscibile
  - ciò completa la prima parte della dimostrazione

un linguaggio che non è Turing-riconoscibile

- dimostrazione:
  - per ciò che riguarda la direzione opposta
  - se sia A che $\underline{A}$ sono Turing-riconoscibili, siano M ed N le MT che riconoscono A e $\underline{A}$
  - costruiamo una MT che decide A eseguendo M ed N in parallelo sullo stesso input
  - Se M accetta la macchina accetta, se N accetta la macchina rifiuta

un linguaggio che non è Turing-riconoscibile

- teorema: $\underline{A}_{TM}$ non è Turing-riconoscibile
- dimostrazione:
  - – se lo fosse, A_{TM} sarebbe decidibile