Indecidibilità e Riducibilità

Tutorato 9: Limiti della Computabilità

Automi e Linguaggi Formali

Gabriel Rovesti

Corso di Laurea in Informatica - Università degli Studi di Padova

19 Maggio 2025

Contents

1 Il Metodo della Diagonalizzazione

1.1 Origini e Concetto

Concetto chiave

Il metodo della diagonalizzazione è una tecnica di dimostrazione matematica scoperta da Georg Cantor nel 1873. Viene utilizzato per confrontare le dimensioni di insiemi infiniti e dimostrare l'esistenza di problemi algoritmicamente irrisolvibili.

1.2 Cardinalità e Corrispondenze

Per comprendere il concetto di diagonalizzazione, dobbiamo prima chiarire alcune nozioni fondamentali:

Definizione

Date due insiemi A e B e una funzione $f : A \to B$:

- f è **iniettiva** se $f(a) \neq f(b)$ ogniqualvolta $a \neq b$
- f è **suriettiva** se per ogni $b \in B$ esiste $a \in A$ tale che $f(a) = b$
- f è **biettiva** se è sia iniettiva che suriettiva

Due insiemi A e B hanno la stessa **cardinalità** se esiste una funzione biettiva $f : A \to B$.

Definizione

Un insieme è **numerabile** (o contabile) se è finito oppure ha la stessa cardinalità dell'insieme dei numeri naturali $\mathbb{N}$.

1.3 Esempi Importanti

- $\mathbb{N}$ (numeri naturali) è numerabile per definizione
- $\mathbb{Z}$ (interi) è numerabile
- $\mathbb{Q}$ (razionali) è numerabile
- $\mathbb{R}$ (reali) non è numerabile (dimostrato tramite diagonalizzazione)
- Σ^* (l'insieme di tutte le stringhe su un alfabeto finito) è numerabile
- L'insieme di tutte le macchine di Turing è numerabile
- L'insieme di tutti i linguaggi su Σ^* non è numerabile

Concetto chiave

Il fatto che l'insieme di tutte le macchine di Turing sia numerabile, mentre l'insieme di tutti i linguaggi non lo sia, implica che devono esistere linguaggi che non possono essere riconosciuti da alcuna macchina di Turing.

2 Problemi Indecidibili

2.1 Il Problema dell'Accettazione

Definizione

Il problema dell'accettazione per le macchine di Turing è definito come:

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una TM che accetta la stringa } w\}$$

Teorema

A_{TM} è indecidibile.

Procedimento di risoluzione

Dimostriamo per contraddizione che A_{TM} è indecidibile:

1. Supponiamo che esista un decisore H per A_{TM}
2. H accetta $\langle M, w\rangle$ se M accetta w, altrimenti rifiuta
3. Costruiamo una nuova TM D che usa H come subroutine:
   - D su input $\langle M\rangle$:
     (a) Esegue H su input $\langle M, \langle M\rangle\rangle$
     (b) Se H accetta, D rifiuta; se H rifiuta, D accetta
4. Analizziamo cosa succede quando D riceve in input la propria codifica $\langle D\rangle$:
   - D accetta $\langle D\rangle$ se e solo se D non accetta $\langle D\rangle$
5. Questa è una contraddizione logica, quindi la nostra ipotesi iniziale deve essere falsa
6. Concludiamo che A_{TM} è indecidibile

Concetto chiave

Questa dimostrazione utilizza un argomento diagonale simile a quello usato da Cantor, e mostra come si possa costruire una TM che genera una contraddizione se supponiamo l'esistenza di un decisore per A_{TM}.

Suggerimento

Sebbene A_{TM} sia indecidibile, è comunque Turing-riconoscibile. Possiamo costruire una TM che simula M su input w e accetta se M accetta, ma non possiamo costruire una TM che si fermi sempre e decida se M accetta w.

2.2 La Macchina Universale di Turing

Concetto chiave

La Macchina Universale di Turing, introdotta da Alan Turing nel 1936, è una TM capace di simulare qualsiasi altra TM a partire dalla sua descrizione. Questo concetto è fondamentale nella teoria della computabilità e può essere visto come il primo modello teorico di computer programmabile.

La macchina universale U può essere descritta informalmente come:

- U su input $\langle M, w \rangle$:
  1. Simula M su input w
  2. Se la simulazione raggiunge lo stato di accettazione, accetta
  3. Se la simulazione raggiunge lo stato di rifiuto, rifiuta

U è un riconoscitore per A_{TM}, ma non è un decisore perché non può determinare se M entrerà in loop su w.

3 Linguaggi Non Turing-riconoscibili

3.1 Complemento e Riconoscibilità

Teorema

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

Procedimento di risoluzione

Dimostriamo entrambe le direzioni:

1. ($\Rightarrow$) Se A è decidibile, allora esiste una TM M che decide A. Possiamo usare M per costruire riconoscitori sia per A che per $\overline{A}$.
2. ($\Leftarrow$) Se A è Turing-riconoscibile e $\overline{A}$ è Turing-riconoscibile, allora esistono TM M_1 e M_2 che riconoscono A e $\overline{A}$ rispettivamente. Possiamo costruire un decisore per A che esegue M_1 e M_2 in parallelo e termina quando uno dei due accetta.

Definizione

Un linguaggio è **co-Turing riconoscibile** se il suo complemento è Turing-riconoscibile.

3.2 Linguaggi non Turing-riconoscibili

Teorema

Il complemento di A_{TM}, denotato come $\overline{A_{TM}}$, non è Turing-riconoscibile.

Procedimento di risoluzione

Se $\overline{A_{TM}}$ fosse Turing-riconoscibile, allora A_{TM} sarebbe sia Turing-riconoscibile che co-Turing riconoscibile, quindi sarebbe decidibile. Ma abbiamo già dimostrato che A_{TM} è indecidibile, quindi $\overline{A_{TM}}$ non può essere Turing-riconoscibile.

Concetto chiave

L'esistenza di linguaggi non Turing-riconoscibili conferma l'intuizione che deriva dal confronto tra la cardinalità dell'insieme delle TM (numerabile) e la cardinalità dell'insieme dei linguaggi (non numerabile).

4 Riducibilità

4.1 Concetto di Riduzione

Definizione

Una riduzione è un modo per trasformare un problema in un altro problema, in modo che una soluzione per il secondo problema possa essere utilizzata per risolvere il primo.

Concetto chiave

Se un problema A è riducibile a un problema B, allora B è almeno tanto difficile quanto A. Questo significa che:

- Se B è decidibile, allora anche A è decidibile
- Se A è indecidibile, allora anche B è indecidibile

4.2 Dimostrazioni per Riduzione

Il metodo standard per dimostrare l'indecidibilità di un problema B attraverso la riduzione è:

1. Assumere per contraddizione che B sia decidibile
2. Mostrare come ridurre un problema già noto come indecidibile (es. A_{TM}) a B
3. Concludere che, poiché A è indecidibile, anche B deve essere indecidibile

4.3 Altri Problemi Indecidibili

4.3.1 Il Problema della Fermata

Definizione

Il problema della fermata è definito come:

$$HALT_{TM} = \{\langle M, w\rangle \mid M \text{ è una TM che si ferma su input } w\}$$

Procedimento di risoluzione

Possiamo dimostrare che $HALT_{TM}$ è indecidibile riducendo A_{TM} a $HALT_{TM}$:

1. Supponiamo che $HALT_{TM}$ sia decidibile tramite un decisore R
2. Costruiamo un decisore S per A_{TM} che usa R:
   - S su input $\langle M, w \rangle$:
     (a) Costruisce una TM M' che su input x:
        i. Simula M su w
        ii. Se M accetta w, allora M' accetta x
        iii. Se M rifiuta w, allora M' rifiuta x
     (b) Usa R per decidere se M' si ferma su un input arbitrario
     (c) Se M' si ferma, allora M si ferma su w, quindi S simula M su w e risponde di conseguenza
     (d) Se M' non si ferma, allora M non si ferma su w, quindi S rifiuta
3. Ma questo è impossibile perché A_{TM} è indecidibile

4.3.2 Il Problema del Vuoto

Definizione

Il problema del vuoto è definito come:

$$E_{TM} = \{\langle M \rangle \mid M \text{ è una TM tale che } L(M) = \emptyset\}$$

Procedimento di risoluzione

Dimostriamo che E_{TM} è indecidibile riducendo A_{TM} a E_{TM}:

1. Supponiamo che E_{TM} sia decidibile tramite un decisore R
2. Costruiamo un decisore S per A_{TM} che usa R:
   - S su input $\langle M, w \rangle$:
     (a) Costruisce una TM M_w che:
        i. Su input x, ignora x e simula M su w
        ii. Accetta se e solo se M accetta w
     (b) Usa R per decidere se $L(M_w) = \emptyset$
     (c) Se $L(M_w) = \emptyset$, allora M non accetta w, quindi S rifiuta
     (d) Se $L(M_w) \neq \emptyset$, allora M accetta w, quindi S accetta
3. Ma questo è impossibile perché A_{TM} è indecidibile

4.4 Il Problema dell'Equivalenza

Definizione

Il problema dell'equivalenza è definito come:

$$EQ_{TM} = \{\langle M_1, M_2 \rangle \mid L(M_1) = L(M_2)\}$$

Possiamo dimostrare che EQ_{TM} è indecidibile riducendo E_{TM} a EQ_{TM}.

5 Riducibilità mediante Funzione

5.1 Definizione Formale

Definizione

Una funzione $f : \Sigma^* \to \Sigma^*$ è **calcolabile** se esiste una TM M che, su input w, termina con f(w) sul nastro.

Definizione

Un linguaggio A è **riducibile mediante funzione** al linguaggio B, scritto come $A \leq_m B$, se esiste una funzione calcolabile $f : \Sigma^* \to \Sigma^*$ tale che per ogni stringa w:

$$w \in A \text{ se e solo se } f(w) \in B$$

La funzione f è detta **riduzione** da A a B.

5.2 Proprietà delle Riduzioni

Teorema

Se $A \leq_m B$ e B è decidibile, allora A è decidibile.

Teorema

Se $A \leq_m B$ e A è indecidibile, allora B è indecidibile.

Teorema

Se $A \leq_m B$ e B è Turing-riconoscibile, allora A è Turing-riconoscibile.

Teorema

Se $A \leq_m B$ e A non è Turing-riconoscibile, allora B non è Turing-riconoscibile.

5.3 Esempi di Riduzioni mediante Funzione

5.3.1 $A_{TM} \leq_m HALT_{TM}$

Possiamo costruire una riduzione f che mappa $\langle M, w\rangle$ a $\langle M', w'\rangle$ dove M' è una TM che su input w':

1. Simula M su w
2. Se M accetta w, allora M' si ferma (accettando)
3. Se M rifiuta w, allora M' si ferma (rifiutando)
4. Se M non si ferma su w, allora M' non si ferma

In questo modo, M accetta w se e solo se M' si ferma su w'.

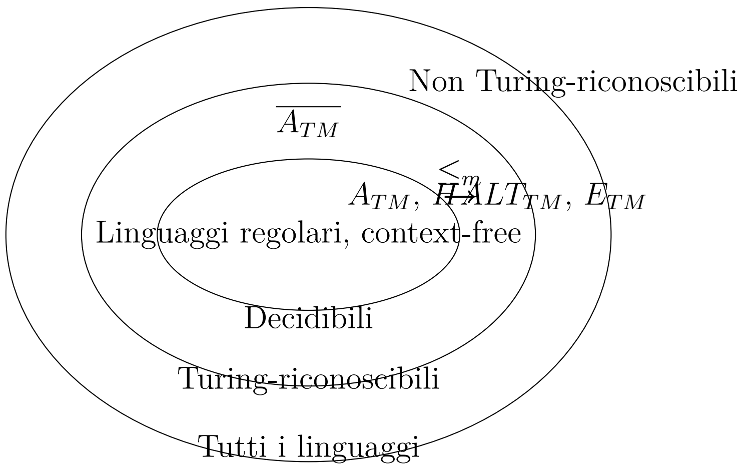
5.3.2 $E_{TM} \leq_m EQ_{TM}$

Possiamo costruire una riduzione f che mappa $\langle M\rangle$ a $\langle M_1, M_2\rangle$ dove:

- M_1 è una TM che rifiuta ogni input (quindi $L(M_1) = \emptyset$)
- M_2 è la TM originale M

In questo modo, $L(M) = \emptyset$ se e solo se $L(M_1) = L(M_2)$.

6 Gerarchia dei Problemi Indecidibili



Concetto chiave

La teoria della calcolabilità ci mostra che esistono limiti fondamentali a ciò che può essere calcolato algoritmicamente. I problemi indecidibili non sono semplici curiosità matematiche, ma hanno implicazioni profonde in informatica, logica e matematica.

Errore comune

Un errore comune è pensare che tutti i problemi computazionali possano essere risolti con algoritmi sufficientemente sofisticati o con computer più potenti. La teoria dell'indecidibilità dimostra invece che esistono problemi che non possono essere risolti da alcun algoritmo, indipendentemente dalla potenza di calcolo disponibile.