Soluzioni Esercizi Automi

Gabriel Rovesti

June 2024

1 Riducibilità

Esercizio 1

Dimostrazione dell'Indecidibilità di ALLTM

Il problema ALLTM è definito come:

$$ALLTM = \{\langle M \rangle \mid M \text{ è una TM tale che } L(M) = \Sigma^*\}$$

Dove $L(M)$ rappresenta il linguaggio riconosciuto dalla macchina di Turing M, e Σ^* è l'insieme di tutte le possibili stringhe sul alfabeto Σ.

Per dimostrare che ALLTM è indecidibile, usiamo una riduzione dal problema dell'arresto (HALT), noto per essere indecidibile. La riduzione procede nel seguente modo:

1. Data un'istanza $\langle M, w \rangle$ del problema HALT, costruiamo una nuova TM M' come segue:
   - M' simula M sull'input w.
   - Se M si ferma su w, M' accetta qualsiasi input.
   - Se M non si ferma su w, M' rifiuta o non si ferma.
2. Se M si ferma su w, allora $L(M') = \Sigma^*$, altrimenti $L(M') \neq \Sigma^*$.

Questa costruzione mostra che decidere se $L(M') = \Sigma^*$ è equivalente a decidere se M si ferma su w. Poiché HALT è indecidibile, segue che anche ALLTM è indecidibile.

Esercizio 2

Dimostrazione dell'Indecidibilità di X

Il problema X è definito come:

$$X = \{\langle M, w \rangle \mid M \text{ è una TM a nastro singolo che non modifica la porzione di nastro che contiene l'input } w\}$$

Per dimostrare che X è indecidibile, consideriamo la riduzione dal problema dell'arresto (HALT). La riduzione procede come segue:

1. Dato $\langle M, w\rangle$, costruiamo una TM M' che opera nel seguente modo:
   - M' copia w in una parte separata del nastro.
   - M' esegue M sull'input w senza modificare la copia originale di w.
   - Se M termina, M' termina e accetta, altrimenti continua a eseguire indefinitamente.
2. Se M si ferma su w, M' non modifica la porzione di nastro che contiene la copia di w e accetta. Altrimenti, M' non termina mai, rispettando la condizione di non modifica.

Poiché decidere se M si ferma su w è indecidibile, ne consegue che anche decidere se $\langle M', w\rangle \in X$ è indecidibile.

Esercizio 3

Se $A \leq_m B$ e B è un linguaggio regolare, A è necessariamente un linguaggio regolare?

La risposta è no. Una riduzione mediante mappatura ($\leq_m$) da un linguaggio A a un linguaggio B regolare implica che esiste una funzione computabile f tale che per ogni stringa x, $x \in A$ se e solo se $f(x) \in B$. Questa condizione non garantisce che A debba essere regolare, perché la funzione f potrebbe trasformare le stringhe in modo che le proprietà non regolari di A siano "nascoste" entro la struttura regolare di B.

Per esemplificare, consideriamo il linguaggio A dei palindromi su $\{0, 1\}$, che non è regolare. Possiamo definire una funzione f che mappa ogni stringa in A (e non in A) su una stringa vuota ϵ, che appartiene a un linguaggio B che consiste solo della stringa vuota. In questo caso, B è chiaramente regolare, ma A non lo è. La riduzione da A a B tramite f è valida, ma non trasforma A in un linguaggio regolare.

Questo esempio dimostra che anche se B è regolare, A può non esserlo, anche se esiste una riduzione mediante mappatura da A a B.

Esercizio 4

Dimostrazione che A_TM non è riducibile mediante funzione a E_TM

Il problema di accettazione per le macchine di Turing (A_TM) e il problema di vuotezza per le macchine di Turing (E_TM) sono definiti rispettivamente come:

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

$$E_TM = \{\langle M\rangle \mid L(M) = \emptyset\}$$

Per dimostrare che non esiste una riduzione funzionale da A_TM a E_TM, dobbiamo mostrare che non è possibile costruire una funzione computabile f tale che, data una coppia $\langle M, w\rangle$, $f(\langle M, w\rangle) = \langle M'\rangle$ con $L(M') = \emptyset$ se e solo se M accetta w.

Supponiamo per assurdo che tale funzione f esista. Allora per ogni $\langle M, w\rangle$ tale che M accetta w, f dovrebbe trasformare questa coppia in una macchina M' il cui linguaggio è vuoto, e viceversa. Questo significa che f dovrebbe essere in grado di determinare se M accetta w e costruire M' in modo che $L(M')$ sia vuoto solo se M accetta w. Tuttavia, questo va contro la definizione di A_TM, che è un problema indecidibile, e quindi non può essere ridotto a E_TM, che è un problema decidibile. Pertanto, una tale funzione f non può esistere, dimostrando che A_TM non è riducibile a E_TM.

Esercizio 5

Dimostrazione che se A è Turing-riconoscibile e $A \leq_m \overline{A}$, allora A è decidibile

Se un linguaggio A è Turing-riconoscibile e esiste una riduzione mediante mappatura da A al suo complemento $\overline{A}$, possiamo dimostrare che A è decidibile. Supponiamo che f sia la funzione di riduzione tale che per ogni stringa x, $x \in A$ se e solo se $f(x) \in \overline{A}$.

Dato che A è Turing-riconoscibile, esiste una macchina di Turing M che accetta tutte le stringhe in A. Per verificare se una stringa x appartiene ad A, utilizziamo il seguente algoritmo:

1. Calcolare $f(x)$ usando la funzione di riduzione f.

2. Simulare M su $f(x)$. Poiché $f(x) \in \overline{A}$ se $x \in A$, stiamo verificando se $f(x)$ è rifiutata da M.

3. Se M rifiuta $f(x)$, allora x appartiene ad A. Se M accetta $f(x)$, allora x non appartiene ad A.

Questo algoritmo è decidibile perché f è computabile e M termina per tutte le stringhe in $\overline{A}$ (dato che rifiuta le stringhe in A e accetta quelle in $\overline{A}$). Poiché M è una macchina di Turing che riconosce $\overline{A}$, e $f(x)$ mappa stringhe da A a $\overline{A}$ e viceversa, M può essere usata per decidere l'appartenenza di x ad A o a $\overline{A}$.

Quindi, dato che possiamo decidere l'appartenenza di ogni stringa x ad A utilizzando M e f, il linguaggio A è decidibile.