Automi e Linguaggi Formali – 10/7/2023
Secondo appello – Seconda Parte: Macchine di Turing, decidibilità ed indecidibilità

1. **(12 punti)** Una *R2-L3 Turing Machine* è una macchina di Turing deterministica a nastro semi-infinito che può effettuare solo due mosse: spostarsi a destra di due celle (R2), oppure spostarsi a sinistra di tre celle (L3). Se in uno spostamento a sinistra la macchina tenta di spostare la testina a sinistra dell'inizio del nastro, allora lo spostamento termina con la testina nella prima cella del nastro.

   (a) Dai una definizione formale della funzione di transizione di una R2-L3 Turing Machine.

   (b) Dimostra che le R2-L3 Turing Machine riconoscono la classe dei linguaggi Turing-riconoscibili. Usa una descrizione a livello implementativo per definire le macchine di Turing.

Soluzione.

(a) $\delta : Q \times \Gamma \mapsto Q \times \Gamma \times \{L3, R2\}$

(b) Per dimostrare che le R2-L3 Turing Machine riconoscono la classe dei linguaggi Turing-riconoscibili dobbiamo dimostrare due cose: che ogni linguaggio Turing-riconoscibile è riconosciuto da una R2-L3 Turing Machine, e che ogni linguaggio riconosciuto da una R2-L3 Turing Machine è Turing-riconoscibile.

Per la prima dimostrazione, mostriamo come convertire una macchina di Turing deterministica a nastro semi-infinito M in una R2-L3 Turing Machine S equivalente.

S = "Su input w:

1. Per simulare una mossa del tipo $\delta(q, a) = (r, b, L)$, S scrive b sul nastro e muove la testina di due celle a destra, e subito dopo di tre celle a sinistra. Se lo spostamento a sinistra porta la testina oltre l'inizio del nastro, allora vuol dire che la simulazione era partita dalla prima cella del nastro. In questo caso la simulazione riprende con la testina nella prima cella del nastro, come per le TM standard. Negli altri casi, la simulazione continua con la testina in corrispondenza della cella immediatamente a sinistra di quella di partenza.
2. Per simulare una mossa del tipo $\delta(q, a) = (r, b, R)$, S scrive b sul nastro e muove la testina di due celle a destra, di nuovo di due celle a destra, e subito dopo di tre celle a sinistra. La simulazione continua con la testina in corrispondenza della cella immediatamente a destra di quella di partenza.
3. Se in qualsiasi momento la simulazione raggiunge lo stato di accettazione di M, allora S termina con accettazione. Se in qualsiasi momento la simulazione raggiunge lo stato di rifiuto di M, allora S termina con rifiuto. Negli altri casi continua la simulazione dal punto 2."

Per dimostrare che ogni linguaggio riconosciuto da una R2-L3 Turing Machine è Turing-riconoscibile, mostriamo come convertire una R2-L3 Turing Machine S in una TM deterministica a nastro semi-infinito M equivalente.

M = "Su input w:

1. Per simulare una mossa del tipo $\delta(q, a) = (r, b, L3)$, M scrive b sul nastro e muove la testina di tre celle a sinistra. Se lo spostamento a sinistra porta la testina oltre l'inizio del nastro, allora lo sposta meno a sinistra si ferma con la testina nella prima cella del nastro, come per le R2-L3 Turing Machine.
2. Per simulare una mossa del tipo $\delta(q, a) = (r, b, R2)$, M scrive b sul nastro e muove la testina di due celle a destra.
3. Se in qualsiasi momento la simulazione raggiunge lo stato di accettazione di S, allora M termina con accettazione. Se in qualsiasi momento la simulazione raggiunge lo stato di rifiuto di S, allora M termina con rifiuto. Negli altri casi continua la simulazione dal punto 2."

2. (12 punti) Dati due DFA, considera il problema di determinare se esiste una stringa accettata da entrambi.

(a) Formula questo problema come un linguaggio $AGREE_{\text{DFA}}$.

(b) Dimostra che $AGREE_{\text{DFA}}$ è decidibile.

Soluzione.

(a) $AGREE_{\text{DFA}} = \{\langle A, B\rangle \mid A, B \text{ sono DFA, ed esiste una parola } w \text{ tale che } w \in L(A) \text{ e } w \in L(B)\}$

(b) La seguente macchina N usa la Turing machine M che decide E_{DFA} per decidere $AGREE_{\text{DFA}}$:

$N =$ "su input $\langle A, B\rangle$, dove A, B sono DFA:
1. Costruisci il DFA C che accetta l'intersezione dei linguaggi di A e B
2. Esegui M su input $\langle C\rangle$. Se M accetta, rifiuta, se M rifiuta, accetta."

Mostriamo che N è un decisore dimostrando che termina sempre e che ritorna il risultato corretto. Sappiamo che esiste un algoritmo per costruire l'intersezione di due DFA. Il primo step di N si implementa eseguendo questo algoritmo, e termina sempre perché la costruzione dell'intersezione termina. Il secondo step termina sempre perché sappiamo che E_{DFA} è un linguaggio decidibile. Quindi N termina sempre la computazione.
Vediamo ora che N dà la risposta corretta:

- Se $\langle A, B\rangle \in AGREE_{\text{DFA}}$ allora esiste una parola che viene accettata sia da A che da B, e quindi il linguaggio $L(A) \cap L(B)$ non può essere vuoto. Quindi $\langle C\rangle \notin E_{\text{DFA}}$, e l'esecuzione di M terminerà con rifiuto. N ritorna l'opposto di M, quindi accetta.
- Viceversa, se $\langle A, B\rangle \notin AGREE_{\text{DFA}}$ allora non esiste una parola che sia accettata sia da A che da B, e quindi il linguaggio $L(A) \cap L(B)$ è vuoto. Quindi $\langle C\rangle \in E_{\text{DFA}}$, e l'esecuzione di M terminerà con accettazione. N ritorna l'opposto di M, quindi rifiuta.

3. (12 punti) Data una Turing Machine M, considera il problema di determinare se esiste un input tale che M scrive "$xyzzy$" su cinque celle adiacenti del nastro. Puoi assumere che l'alfabeto di input di M non contenga i simboli x, y, z.

(a) Formula questo problema come un linguaggio $MAGIC_{\text{TM}}$.

(b) Dimostra che il linguaggio $MAGIC_{\text{TM}}$ è indecidibile.

Soluzione.

(a) $MAGIC_{\text{TM}} = \{\langle M\rangle \mid M \text{ è una TM che scrive } xyzzy \text{ sul nastro per qualche input } w\}$

(b) La seguente macchina F calcola una riduzione $A_{\text{TM}} \leq_m MAGIC_{\text{TM}}$:

$F =$ "su input $\langle M, w\rangle$, dove M è una TM e w una stringa:
1. Verifica che i simboli x, y, z non compaiano in w, né nell'alfabeto di input o nell'alfabeto del nastro di M. Se vi compaiono, sostituiscili con tre nuovi simboli X, Y, Z nella parola w e nella codifica di M.
2. Costruisci la seguente macchina M':

   $M' =$ "Su input x:
   1. Simula l'esecuzione di M su input w, senza usare i simboli x, y, z.
   2. Se M accetta, scrivi $xyzzy$ sul nastro, altrimenti rifiuta senza modificare il nastro.

3. Ritorna $\langle M'\rangle$."

Mostriamo che F calcola una funzione di riduzione da A_{TM} a $MAGIC_{\text{TM}}$, cioè una funzione tale che

$$\langle M, w\rangle \in A_{\text{TM}} \text{ se e solo se } \langle M'\rangle \in MAGIC_{\text{TM}}.$$

- Se $\langle M, w\rangle \in A_{\text{TM}}$ allora la macchina M accetta w. In questo caso la macchina M' scrive $xyzzy$ sul nastro per tutti gli input. Di conseguenza $\langle M'\rangle \in MAGIC_{\text{TM}}$.
- Viceversa, se $\langle M, w\rangle \notin A_{TM}$ allora la macchina M rifiuta o va in loop su w. Per tutti gli input, la macchina M' simula l'esecuzione di M su w senza usare i simboli x, y, z (perché sono stati tolti dalla definizione di M e di w se vi comparivano), e rifiuta o va in loop senza scrivere mai $xyzzy$ sul nastro. Di conseguenza $\langle M'\rangle \notin MAGIC_{\text{TM}}$.