Esercitazione 3 - Linguaggi e Calcolabilità

12-04-2019

Antonio Cruciani
antonio.cruciani@alumni.uniroma2.eu

Esercizi a lezione

Esercizio 1:
Sia L il linguaggio definito come segue:

$$L_{HB} = \{\langle M \rangle : \text{M è una macchina di turing e } M(\epsilon) \text{ Termina}\}$$

Si discuta la decidibilità di tale linguaggio.
Esercizio 2:
Sia $L_1 \subseteq \Sigma^*$ un linguaggio decidibile deciso da una macchina di Turing T_1, e sia $L_2 \subseteq \Sigma^*$ un linguaggio accettabile, accettado dalla macchina di Turing T_2. Dimostrare se il seguente linguaggio:

$$L = \{(x, k) : x \in \Sigma^* \land k \in \mathbb{N} \land T_1(x) \textit{ accetta in } r \geq k \textit{ passi } \land T_2(x) \textit{ rigetta in } s \leq k \textit{ passi } \}$$

è decidibile

Esercizio 3:
Si consideri il seguente linguaggio:

$$L_{NH} = \{(\langle M \rangle, x) : \langle M \rangle \textit{ non è la codifica di una macchina di Turing } \lor M(x) \textit{ non termina}\}$$

Si dimostri l'accettabilità o la non accettabilità di tale linguaggio.

Esercizi per casa

Esercizio 1:
Sia $L_1 \subseteq \Sigma^*$ un linguaggio decidibile deciso da una macchina di Turing T_1, e sia $L_2 \subseteq \Sigma^*$ un linguaggio accettabile ma non decidibile, accettado dalla macchina di Turing T_2. Dimostrare se il seguente linguaggio:

$$L = \{(x,k) : x \in \Sigma^* \wedge k \in \mathbb{N} \wedge T_1(x) \textit{ accetta in } r \geq k \textit{ passi } \wedge T_2(x) \textit{ rigetta in } s \geq k \textit{ passi } \}$$

è decidibile
Esercizio 2:
Sia $f : \{0,1\}^* \rightarrow \mathbb{N}$ una funzione totale e calcolabile, si consideri il seguente linguaggio:
(Versione modificata dell'Halting Problem)

$$L_H(f) = \{(\langle M \rangle, x) \ : \ M(x) \text{ Termina in } f(x) \text{ passi}\}$$

Si dimostri se $L_H(f)$ è decidibile o non decidibile.
Esercizio 3*:
Sia L_{NE} il linguaggio dalle macchine di Turing che accettano un linguaggio non vuoto, ovvero:

$$L_{NE} = \{\langle M \rangle : M \text{ è un macchina di Turing } \wedge \mathcal{L}(M) \neq \emptyset\}$$

Si discuta l'accettabilità e la decidibilità di tale linguaggio.
Esercizio 4:**
Sia

$$L_{EQ} = \{\langle M_1, M_2 \rangle : M_1 \textit{ e } M_2 \text{ sono due macchine di Turing } \wedge \mathcal{L}(M_1) = \mathcal{L}(M_2)\}$$

Si discuta la decidibilità di L_{EQ}
Esercizio 5*:
In una macchina di Turing si definisce *stato inutile* uno stato al quale non si fa mai accesso per qualsiasi parola in input. Si consideri il problema di determinare se uno stato in una macchina di Turing è inutile. Si definisca questo problema come linguaggio e se ne dimostri la decidibilità o la non decidibilità.
Legenda:
Sia M una macchina di Turing tale che accetta $x \in \Sigma^*$,
scriviamo $\mathcal{L}(M) = \{x \in \Sigma^* : M \text{ Accetta } x\}$ e ci riferiamo a $\mathcal{L}(M)$ come al linguaggio accettato da M.
Gli asterischi dopo il numero degli esercizi indicano la loro difficoltà, più sono gli * e più richiede impegno l'esercizio.

Soluzioni esercizi a lezione

Esercizio 1:
Mostriamo che L_{HB} non è decidibile in quanto $L_{ATM} \leq_m L_{HB}$:
Sia $x \in \Sigma^*$ e si assuma che $x = \langle M, w \rangle$ dove M è una macchina di Turing. (Se x non è di questa forma allora possiamo definire $f(x)$ affinché sia qualcosa non in L_{HB}. In generale assumeremo che l'input sarà **ben formato**).
Sia $f(x) = \langle M' \rangle$ dove M' lavora come segue sul nastro vuoto (non ci interessa cosa faccia M' quando il nastro non è vuoto).
Spieghiamo il funzionamento di M': per prima cosa M' scrive w sul nastro (questo può farlo poiché possiamo assumere che la parola w sia memorizzata negli stati interni di M' in quanto w è sempre una parola finita) e poi simula la computazione $M(w)$:

a) Se $M(w)$ termina e accetta allora **Accetta**

b) Se $M(w)$ termina e rigetta allora **Non termina**

Osserviamo esplicitamente che $f(x)$ è una funzione calcolabile e soprattutto, osserviamo che :

$$x \in L_{ATM} \iff f(x) \in L_{HB}$$

poiché

$$M \text{ } \textit{Accetta } w \iff M' \text{ } \textit{Termina sul nastro vuoto}$$

Quindi possiamo concludere che L_{HB} non è decidibile.

Esercizio 2:
Osserviamo che il linguaggio L è decidibile.
Dimostriamolo.
Osserviamo esplicitamente che L_1 è un linguaggio decidibile, quindi esiste una macchina di Turing T_1 che decide tale linguaggio.L_2 è un linguaggio accettabile ma non decidibile, quindi esiste una macchina di Turing che accetta tale linguaggio.
Osserviamo che L è definito come segue:

$$L = \{(x, k) : x \in \Sigma^* \wedge k \in \mathbb{N} \wedge T_1(x) \textit{ accetta in } r \geq k \textit{ passi } \wedge T_2(x) \textit{ rigetta in } s \leq k \textit{ passi } \}$$

Definiamo la macchina di Turing T che decide tale linguaggio.
Senza perdita di generalità definiamo T come segue: T è composta da 4 nastri:

N_1) Nastro d'input che contiene (x, k)

N_2) Nastro dove trascriverà k in unario

N_3) Nastro dove simulerà la computazione $T_1(x)$

N_4) Nastro dove simulerà la computazione $T_2(x)$

Illustriamo il funzionamento di tale macchina:

1. input (x, k)
2. Controlla che $x \in \Sigma^* \wedge k \in \mathbb{N}$, se $x \notin \Sigma^* \vee k \notin \mathbb{N}$ **Rigetta**, altrimenti prosegui con il passo 3)
3. leggi k presente in N_1 e trascrivilo in unario in N_2
4. Posiziona la tesista di N_2 sul primo $\square$ a sinistra su N_2
5. Simula la computazione $T_1(x)$ come segue: Ad ogni passo della simulazione di $T_1(x)$ sposta a destra la testina su N_2 di una posizione. Ora,
   - Se $T_1(x)$ accetta e la testina su N_2 legge blank allora **prosegui con il passo 5** riposizionando la testina sul primo $\square$ a sinistra su N_2.
   - Se $T_1(x)$ accetta e la testina su N_2 legge 1 allora, sposta a destra di una posizione la testina su N_2:
     - se legge 1 allora **Rigetta**
     - se legge $\square$ allora **Prosegui con il passo 5**
   - Se $T_1(x)$ rigetta allora **Rigetta**
6. Simula $T_2(x)$ come alla fase 5 e se :
   - Se $T_2(x)$ accetta e su N_2 legge 1 allora **Rigetta**
   - Se $T_2(x)$ non ha né accettato né rifiutato e sul nastro N_2 legge blank, allora **Rigetta**
   - Se $T_2(x)$ rigetta e sul nastro N_2 legge un 1 allora **Accetta**

Osserviamo esplicitamente che la macchina T appena descritta decide L. Abbiamo quindi dimostrato che L è decidibile.

Esercizio 3:
Osserviamo che il linguaggio non è accettabile in quanto è il complemento del linguaggio L_{HALT}.
Assumiamo per assurdo che L_{NH} sia accettabile, allora possiamo costruire una macchina di Turing che decide il linguaggio L_{HALT}. Ovvero, sia T_1 la macchina di Turing composta di tre nastri: sul primo nastro sarà presente l'input, sul secondo nastro verrà simulata la macchina T_{Halt} che accetta $\forall(\langle M\rangle, x) \in L_{Halt}$ e sul terzo nastro verrà simulata T_{NH} che accetta $\forall(\langle M\rangle, x) \in L_{NH}$. T_1 lavorerà come segue:

1) Controlla se $\langle M\rangle$ è la codifica di una macchina di Turing, se non lo è **Rigetta**, altrimenti prosegui con il passo 2).

2) Alterna un passo di simulazione di $T_{Halt}(\langle M\rangle, x)$ e $T_{NH}(\langle M\rangle, x)$, se T_{Halt} accetta allora **Accetta** se T_{NH} accetta allora **Rigetta**.

Abbiamo quindi definito una macchina di Turing in grado di decidere L_{Halt}, ma tale linguaggio non è decidibile, è solo accettabile (ovvero non è co-Turing-recognizable) quindi tale macchina T_{NH} non può esistere e quindi L_{NH} non è un linguaggio accettabile.

Soluzioni esercizi per casa

Esercizio 1:
Il lunguaggio L non è decidibile.
Dimostriamolo. Osserviamo esplicitamente che L_1 è un linguaggio decidibile e che L_2 è un linguaggio accettabile ma non decidibile.
Procediamo con la dimostrazione della non decidibilità di L:
Assumiamo per assurdo che L sia decidibile, allora esiste una macchina di Turing T_L che decide L, ovvero. Tale macchina, senza perdita di generalità, sarà a 4 nastri, definita come segue:

N_1) Nastro d'input che contiene (x, k)

N_2) Nastro dove trascriverà k in unario

N_3) Nastro dove simulerà la computazione $T_1(x)$

N_4) Nastro dove simulerà la computazione $T_2(x)$

Illustriamo il funzionamento di tale macchina:
ASSUNZIONE: Ogni coppia (x, k) sarà ben formata, ovvero $\forall(x, k)$ avremo sempre che $x \in \Sigma^* \wedge k \in \mathbb{N}$. Quest'assunzione serve per facilitare l'analisi del linguaggio e può essere, chiaramente, rilassata.

1. input (x, k)
2. leggi k presente in N_1 e trascrivilo in unario in N_2
3. Posiziona la tesista di N_2 sul primo $\square$ prima del primo 1 a sinistra su N_2
4. Simula la computazione $T_1(x)$ come segue: Ad ogni passo della simulazione di $T_1(x)$ sposta a destra la testina su N_2 di una posizione. Ora,
   - Se $T_1(x)$ accetta e la testina su N_2 legge $\square$ allora **prosegui con il passo 5** riposizionando la testina sul primo $\square$ a sinistra su N_2.
   - Se $T_1(x)$ accetta e la testina su N_2 legge 1 allora, sposta a destra di una posizione la testina su N_2:
     - se legge 1 allora **Rigetta**
     - se legge $\square$ allora **Prosegui con il passo 5**
   - Se $T_1(x)$ rigetta allora **Rigetta**

5. Simula $T_2(x)$ come alla fase 4 e se :
   - Se $T_2(x)$ accetta e su N_2 legge 1 allora **Rigetta**
   - Se $T_2(x)$ rigetta e su N_2 legge 1 allora, sposta a destra di una posizione la testina su N_2:
     - se legge 1 allora **Rigetta**
     - se legge $\square$ allora **Accetta**
   - Se $T_2(x)$ rigetta e sul nastro N_2 legge un $\square$ allora **Accetta**
   - Se $T_2(x)$ accetta e sul nastro N_2 legge un $\square$ allora **Rigetta**

Tale macchina riesce a decidere L, abbiamo però che L_2 è un linguaggio accettabile ma non decidibile, quindi abbiamo che al passo 5 la computazione $T_2(x)$ per $x \in L_2^c$, senza perdita di generalità, non termina e di conseguenza non termina nemmeno T_L. Quindi abbiamo che L non è decidibile, ma per ipotesi iniziale abbiamo che L è decidibile, abbiamo quindi ottenuto un assurdo. Possiamo concludere che L è un linguaggio non decidibile.

Esercizio 2:
Mostriamo che $L_H(f)$ è un linguaggio decidibile.
Abbiamo $f : \{0,1\}^* \to \mathbb{N}$ la quale è una totale e calcolabile.
Allora poiché è una funzione totale e calcolabile essa viene calcolata da un trasduttore che $\forall x \in \{0,1\}^*$ scrive $f(x)$ (in questo caso un numero naturale) sul nastro di output. Senza perdita di generalità assumiamo che tale numero naturale sia in unario.
Detto questo, mostriamo che $L_H(f)$ è decidibile, sia T_1 la TM che decide tale linguaggio (il quale è una modifica dell' halting problem), quindi, T_1 senza perdita di generalità, sarà una macchina a 3 nastri, la quale funzionerà come segue: Per prima cosa simula sul nastro 1) il trasduttore che calcola $f(x)$ e scrivi l'output della computazione di tale trasduttore sul nastro 3), poi una volta terminata tale computazione (e tale computazione termina sempre poiché f è totale e calcolabile), riavvolgi la testina del nastro 3) tutta a sinistra e simula sul nastro 2) $M(x)$ come segue: ad ogni passo di $M(x)$ muovi a destra la testina sul nastro d'output. Se $M(x)$ è terminata e sulla cella sulla quale si trova la testina del nastro 3) leggi 1 allora T_1 accetta.
Se $M(x)$ non è ancora terminata e sul nastro 3) leggi un $\square$ allora T_1 rigetta, in quanto $M(x)$ non è terminata in $f(x)$ passi.
Osserviamo esplicitamente che T_1 decide $L_H(f) \Rightarrow L_H(f)$ è decidibile.

Esercizio 3*:
L_{NE} è un linguaggio accettabile ma non decidibile.
Dimostriamo l'accettabilità di L_{NE} costruendo una macchina di Turing M_0 tale che $L_{NE} = \mathcal{L}(M_0)$. M_0 prende in input $x = \langle M \rangle$ ed M_0 si comporta come segue:

- Simula M su tutti gli input di lunghezza ≤ 1 per un passo
- Simula M su tutti gli input di lunghezza ≤ 2 per due passi
- Simula M su tutti gli input di lunghezza ≤ 3 per tre passi
- Simula M su tutti gli input di lunghezza ≤ 4 per quattro passi
- etc..

Se e quando M_0 "scopre" M accetta un qualche input allora M_0 **Accetta** e termina.
Chiaramente $\mathcal{L}(M_0) = L_{NE}$, allora L_{NE} è accettabile.
Procediamo con il dimostrare che tale linguaggio non è decidibile, per fare questo, mostriamo una riduzione $L_{HB} \leq_m L_{NE}$.
Sia x un input di L_{HB}, assumiamo che x sia ben formato, ovvero $x = \langle M \rangle$.
Definiamo $f(x) = \langle M' \rangle$ dove M' lavora come segue:

- Su ogni input
  1. M' cancella il suo inupt
  2. M' simula $M(\epsilon)$
- Se e quando M termina allora M' **Accetta** e termina.

Osserviamo esplicitamente

$$\langle M \rangle \in L_{HB} \iff \langle M' \rangle \in L_{NE} \Rightarrow L_{HB} \leq_m L_{NE}$$

E questo ci permette di dire che L_{NE} non è decidibile.

Esercizio 4:**

Dimostriamo la non decidibilità di questo linguaggio mostrando che se fosse decidibile allora potremmo decidere L_{NE} il quale è non decidibile.
Si supponga che esista una macchina di Turing T_{EQ} che decide L_{EQ}. Allora possiamo costruire una macchina di Turing T' in grado di decidere L_{NE}.
Definiamo tale T':

- Controlla se l'input è della forma $\langle T_1 \rangle$ dove T_1 è una macchina di Turing. Se non lo è **rigetta**, altrimenti esegui i seguenti passi:

1) Costruisci la parola $\langle T_1, T_\emptyset \rangle$, dove $T_\emptyset$ è una macchina di Turing che rigetta tutti gli input (ovvero abbiamo che $\mathcal{L}(T_\emptyset) = \emptyset$) .

2) Simula $T_{EQ}(\langle T_1, T_\emptyset \rangle)$.
   - Se tale computazione è accettante allora **Rigetta**
   - Se tale computazione è rigettante allora **Accetta**.

Osserviamo esplicitamente che :

- Se $\langle T_1 \rangle \notin L_{NE} \Rightarrow \mathcal{L}(T_1) = \emptyset = \mathcal{L}(T_\emptyset)$ e $T_{EQ}(\langle T_1, T_\emptyset \rangle)$ ACCETTA $\Rightarrow$ T' **Rigetta**

- Se $\langle T_1 \rangle \in L_{NE} \Rightarrow \mathcal{L}(T_1) \neq \emptyset = \mathcal{L}(T_\emptyset)$ e $T_{EQ}(\langle T_1, T_\emptyset \rangle)$ RIGETTA $\Rightarrow$ T' **Accetta**

Quindi T' decide L_{NE} il quale non è decidibile, quindi tale macchina non può esistere e di conseguenza nemmeno T_{EQ} esiste e quindi possiamo concludere che L_{EQ} non è decidibile.

Esercizio 5*:
Definiamo il linguaggio:
$L_{ITM} = \{\langle M, q\rangle$: $\textsf{M è una macchina di Turing} \wedge \textsf{q è uno stato inutile in M}\}$
Dimostriamo la non decidibilità di tale linguaggio.
Si supponga per assurdo che L_{ITM} sia decidibile, allora esiste T_{ITM} che lo decide. Si noti che per ogni macchina di Turing T_i con stato accettante q_a, q_a è inutile se e solo se $\mathcal{L}(T_i) = \emptyset$.
Quindi, poiché T_{ITM} decide L_{ITM}, abbiamo un mezzo per controllare se q_a è uno stato inutile e decidere L_{NE} (che sappiamo essere non decidibile).

Definiamo il funzionamento di una tale macchina T' che utilizza T_{ITM} per decidere L_{NE}:

- Su input $\langle M\rangle$ dove M è una macchina di Turing

1) Simula $T_{ITM}(\langle M, q_a\rangle)$ dove q_a è lo stato accettante di M.
   - Se $T_{ITM}(\langle M, q_a\rangle)$ accetta allora **Rigetta**
   - Se $T_{ITM}(\langle M, q_a\rangle)$ rigetta allora **Accetta**

Si osservi che:
Se la computazione $T_{ITM}(\langle M, q_a\rangle)$ è accettante allora significa che q_a è uno stato inutile, ovvero significa che il linguaggio accettato dalla macchina di Turing M è vuoto ($\mathcal{L}(M) = \emptyset$).
Se, invece, la computazione $T_{ITM}(\langle M, q_a\rangle)$ è rigettante allora significa che q_a non è uno stato inutile e quindi che il linguaggio accettato da M non è vuoto ($\mathcal{L}(M) \neq \emptyset$).
Alla luce di queste due osservazioni, possiamo concludere osservando esplicitamente che tale macchina di Turing T' non può esistere in quanto decide L_{NE} e di conseguenza T_{ITM} non può esistere.
Quindi L_{ITM} non è decidibile.