Informatica Teorica 2022/2023 - Esercitazione 2

15 Marzo 2023

melissa.antonelli2@unibo.it

1 Esercizi

Nozioni Richieste. Mapping-Reduction.
Siano L e L' linguaggi su un dato alfabeto Σ, diciamo che L *mapping-riduce* a L', $L \leq L'$, se esiste una TM che computa la funzione $f : \Sigma^* \to \Sigma^*$ tale che:

$$x \in L \quad sse \quad f(x) \in L'.$$

Problema 1. Mostra che $\leq$ é una relazione transitiva.

Soluzione 1. Assumi $L \leq L'$ e $L' \leq L''$. Allora, per definizione di *mapping-reduction*, esistono due funzioni f e g tali che:

$$x \in L \quad sse \quad f(x) \in L'$$
$$y \in L' \quad sse \quad g(y) \in L''.$$

Consideriamo la composizione $h(x) = g(f(x))$. Costruiamo una TM che computa h come segue:

1. Simula una macchina che computa f su input x e chiama l'output y (tale macchina esiste perché f é computabile per ipotesi e Df. di *mapping-reduction*)
2. Simula una TM che computa g su y (anche questa macchina esiste perché f é computabile per ipotesi e Df. di *mapping-reduction*).

L'output é $h(x) = g(f(x))$. Dunque h é una funzione computabile. Inoltre,

$$x \in L \quad sse \quad h(x) \in L''$$

e quindi $L \leq L''$ tramite funzione di riduzione h.

Soluzione 1 (in classe). Devo dimostrare che $\leq$ é transitiva, cioé per ogni L'', L', L, se $L'' \leq L'$ e $L' \leq L$, allora $L'' \leq L$.
(H1) Esiste una TM tale che M computa f tale che per ogni $x \in \Sigma^*$, $x \in L''$ sse $f(x) \in L'$.
(H2) Esiste una TM tale che M' computa g tale che per ogni $x \in L'$ sse $g(x) \in L$.
Considero dunque $h(x) = g \circ f = g(f(x))$. Osservo che $x \in L'$ sse $f(x) \in L'$ (H1) sse $g(f(x)) \in L$ (H2), e cioé $g(f(x)) \in L$.

Nozioni Richieste. Mapping-Reduction; Linguaggio Decidibile.
Una TM M *decide* un linguaggio L quando: se $x \in L$, allora M accetta x; se $x \notin L$, allora M rigetta x. Un linguaggio L si dice *decidibile* quando c'é una TM in grado di decidere L.

Problema 2. Dimostra che se $L \leq L'$ e L' é decidibile, allora L é decidibile.

Soluzione 2. Per ipotesi L' é decidibile dunque, per Df. di linguaggio decidibile, esiste una TM, diciamo M', che decide L'. Per ipotesi $L \leq L'$ e Df. di *mapping-reduction* esiste una funzione computabile f tale che

$$x \in L \quad \text{sse} \quad f(x) \in L'.$$

Consideriamo una TM M costruita come segue. Per ogni input x (nell'alfabeto di L e L'):

1. M computa $f(x)$ (f é computabile per Df. di *mapping-reduction*).[1]
2. M esegue M' su input $f(x)$ e restituisce in output ció che M' restituisce in output.

Per Df. di mapping-reduction, se $x \in L$, allora $f(x) \in L'$. Inoltre, se $x \in L$ M' deve accettare $f(x)$. Conseguentemente M accetta x. Analogamente, se $x \notin L$, allora $f(x) \notin L'$ e, se $f(x) \notin L'$ M' deve rigettare $f(x)$. Conseguentemente M rigetta x. Ma allora, M decide L e, per Df. di linguaggio decidibile, L é provato decidibile.

Soluzione 2 (in classe). Sappiamo che (i) $L \leq L'$ e (ii) L' é decidibile. Devo dimostrare che L é decidibile.
(1) Esiste una TM M che computa f tale per cui per ogni $x \in L$ sse $f(x) \in L'$ (per (i) e Df. di m-reduction).
(2) Esiste $M_{L'}$ tale che per ogni $x \in \Sigma^*$, se $x \in L'$ allora $M_{L'}$ accetta x; se $x \notin L'$ allora $M_{L'}$ rigetta x.
Costruiamo M_L tale che, su input x:

1. M_L esegue x (dove M é la TM citata in (1)), ottenendo $f(x)$
2. M_L esegue $M_{L'}$ su $f(x)$.

Consideriamo i casi possibili (a partire dalla costruzione di M_L): $x \in L$ e $x \notin L'$.

[1] Detto altrimenti, poiché f é computabile, esiste una M_f che computa f. Dunque M esegue M_f con input x, ottenendo in output $f(x)$ (l'output di M_f).

Nozioni Richieste. Mapping-Reduction; Linguaggio Riconoscibile.
Una TM M *riconosce* un linguaggio L quando: se $x \in L$, allora M termina; se $x \notin L$, allora M non termina. Un linguaggio L si dice *riconosce* quando c'é una TM in grado di decidere L.

Problema 2 bis. Dimostra che se $L \leq L'$ e L' é riconoscibile, allora anche L é riconoscibile.

Soluzione 2 bis. Per ipotesi L' é riconoscibile, dunque, per Df. di linguaggio riconoscibile, esiste una TM che lo riconosce, diciamo M'. Inoltre, poiché per ipotesi $L \leq L'$, per Df. di *m-reduction*, esiste una funzione computabile f tale che

$$x \in L \quad \text{sse} \quad f(x) \in L'.$$

Definiamo dunque una TM M che riconosce L nel modo seguente. Per ogni input x (nell'alfabeto desiderato):

1. M computa $f(x)$ (computabile per Df. di m-reduction).[2]
2. M esegue M' e: se M' termina, M termina; altrimenti, M entra in loop.

Osserva che, anche in questo caso, per Df. di m-reduction, se $x \in L$, allora $f(x) \in L'$, dunque M' termina (Df. linguaggio riconoscibile). Quindi, M termina su x. Altrimenti M entra in loop. Dunque, poiché M riconosce L, M é riconoscibile

[2]Come ne caso precedente, ció corrisponde a considerare la TM M_f che computa f (che esiste per Df. di m-reduction e funzione computabile) ed eseguire M_f con x in input, ottenendo ovviamente in output l'output di M_f, ovvero $f(x)$.

Problema 3. Considera il seguente linguaggio

$$U = \{y \in \{0,1\}^* \mid y = \mathsf{code}(M) \ \&\ M \text{ ferma su } 111\}.$$

Dimostra che U é indecidibile sfruttando l'indecidibilitá di $HALT$.
Suggerimento. Ricorda che, per il Corollario 1 (Lezione 7), se $L \leq L'$ e L é indecidibile, allora L' é indecidibile.

Soluzione 3. Questo é il linguaggio delle stringhe tali che la stringa é codifica di una TM e questa TM si ferma su 111. U comprende tutti i codici delle macchine che hanno questo comportamento. Notiamo preliminarmente che questa proprietá non é triviale: il linguaggio delle parole di lunghezza dispari si ferma su tale input, quello delle pari no.

Dimostrazione diretta. Sappiamo che $HALT$ é indecidibile, dunque, per il Corollario 1, per dimostrare l'indecidibilitá di U é sufficiente ridurre $HALT$ a U, ovvero dimostrare $HALT < U$. Per Df. di *mapping-reduction* questo corrisponde a mostrare che esiste una funzione f computabile tale che:

$$\langle y, x\rangle \in HALT \quad \text{sse} \quad f(\langle y, x\rangle) \in U.$$

Costruiamo la funzione di riduzione desiderata come segue:

1. $y \neq \mathsf{code}(M)$, ($y \in \{0,1\}^*$ non codifica una TM) allora sia $f(\langle y, x\rangle) = y$.[3]
2. $y = \mathsf{code}(M)$, ovvero y é il codice di TM M. Definiamo TM $M_{M,x}$ (ovvero una TM costruita a partire dai parametri M, x) tale che:
   - su input 111, cancella il nastro, $M_{M,x}$ scrive x e simula M su x.
   - su ogni altro input, $M_{M,x}$ entra in loop.

   Definiamo dunque $f(\langle y, x\rangle) = \mathsf{code}(M_{M,x})$.[4]

Dimostriamo ora che f é computabile e che

$$\langle y, x\rangle \in HALT \quad \text{sse} \quad f(\langle y, x\rangle) \in U,$$

ovvero $HALT \leq U$. Poiché, come dimostrato, $HALT$ é indecidibile, anche U deve essere indecidibile. f é computabile: se $y \neq \mathsf{code}(M)$, standard (cf. lezioni); se $y = \mathsf{code}(M)$ perché codifica di TM che o simula un'altra TM (rimanendo computabile) o entra in loop.
Il caso $y \neq \mathsf{code}(M)$ é triviale (se $y \neq \mathsf{code}(M)$, allora $f(\langle y, x\rangle) \notin U$, indipendentemente da $M_{M,x}$).
Consideriamo il caso $y = \mathsf{code}(M)$:

$$\langle y, x\rangle \in HALT$$
$$\Updownarrow$$
$$y = \mathsf{code}(M) \ \&\ M \textit{ ferma su } x$$
$$\Updownarrow$$

[3] Per ipotesi di questo caso y non é codice di TM, quindi $y \notin U$.
[4] Dunque la TM si comporta in modo diverso in base all'input: se l'input é 111, allora si comporta come si comporterebbe M su x; se l'input non é 111, allora la macchina cicla.

$$M_{M,x} \textit{ ferma su } \texttt{111} \ (\& \ f(\langle y, x\rangle) = \mathsf{code}(M_{M,x}))$$
$$\Updownarrow$$
$$f(\langle y, x\rangle) \in U$$

ovvero,[5]

$$\langle y, x\rangle \in HALT \quad \text{sse} \quad f(\langle y, x\rangle) \in U.$$

[5] $M_{M,x}$ ferma su $\texttt{111}$ sse M ferma su x. Le due TM hanno lo stesso comportamento quando $M_{M,x}$ riceve y come input (quando non riceve y non importa cosa faccia $M_{M,x}$)

$$y = \mathsf{code}(M) \ \& \ M \textit{ ferma su } x \quad \Leftrightarrow \quad M_{M,x} \textit{ ferma su } \texttt{111}.$$

Nozioni Richieste. Proprietá di Linguaggio; Proprietá Triviale.
Il *linguaggio* della TM M é definito come

$$L_M = \{x \in \{0,1\}^* \mid M \text{ accetta } x\}.$$

Una *proprietá di TM-linguaggio* é una funzione da insieme di TM a $\{0,1\}$, tale che $L_M = L_{M'}$ implica $P(M) = P(M')$. Let TM che soddisfano P sono indicate come:

$$\{y \in \Sigma^* \mid y = \mathsf{code}(M) \ \&\ P(M) = 1\}.$$

Una proprietá di TM-linguaggio é *non triviale* se esiste una TM M tale che $P(M) = 1$ e una TM M' tale che $P(M') = 0$

Problema 4. Quali delle seguenti sono proprietá di linguaggio triviali? Quali no?

a. $\{y \mid y = \mathsf{code}(M)\ \&\ \epsilon \in L_M\}$
b. $\{y \mid y = \mathsf{code}(M)\ \&\ M$ ha almeno uno stato$\}$
c. $\{y \mid y = \mathsf{code}(M)\ \&\ L_M$ contiene tutte le stringhe di lunghezza pari$\}$
d. $\{y \mid y = \mathsf{code}(M)\ \&\ M$ ha 3 stati$\}$.

Soluzione 4.

a. É una proprietá non triviale. (Per es. é proprietá del linguaggio delle stringhe di lunghezza pari ma non delle stringhe di lunghezza dispari.)
b. É una proprietá triviale.
c. É una proprietá non-triviale. (Analogo a caso a.)
d. Non é una proprietá di un linguaggio, ma una proprietá della TM. (Macchine diverse possono definire lo stesso linguaggio indipendentemente dal fatto di avere questa proprietá.)

Nozioni Richieste. Teorema di Rice.
Se P é una proprietá non-triviale, allora "il problema M ha la proprietá P?" é indecidibile.

Problema 5. Enuncia il teorema di Rice. Puoi applicarlo per dimostrare l'indecidibilitá di

$$INF = \{\mathsf{code}(M) \mid M \text{ TM tale che } L(M) \text{ linguaggio infinito}\}?$$

Soluzione 5. INF é un linguaggio di descrizioni di TM. Soddisfa le condizioni richieste dal teorema di Rice:

1. Non é triviale: alcune TM hanno linguaggi infiniti altre no
2. Dipende solo dal linguaggio: se due TM riconoscono ll stesso linguaggio o entrambe hanno descrizioni in INF o nessuna delle due ne ha.

Quindi, per il teorema di Rice, INF é indecidibile.

2 Esercizi Supplementari

Problema 6. Descrivi (in linguaggio naturale) una TM M che decide il linguaggio delle stringhe con uguale numero di `0` e `1`. (Alfabeto $\{\texttt{0}, \texttt{1}\}$)

Soluzione 6. Per ogni stringa in input x, la TM M:

1. Scorre il nastro e segna il primo `0` che non sia stato segnato:
   - se non trova `0` non-segnato, va al passo 4.
   - altrimenti, muove la testina indietro.
2. Scorre il nastro e segna il primo `1` non-segnato; se non trova un `1` non-segnato, M rigetta.
3. Muove la testina indietro nel nastro e va al passo 1.
4. Muove la testina indietro nel nastro e scorre il nastro per vedere se resta qualche `1` non-segnato:
   - se non ve ne sono, M accetta x;
   - altrimenti, M rigetta x.

Problema 7. Supponi che L sia riconoscibile e il suo complemento L^- non sia riconoscibile. Considera il linguaggio

$$L' = \{\texttt{0}x \mid x \in L\} \cup \{\texttt{1}x \mid x \notin L\}.$$

(a.) Il linguaggio L' é decidibile? riconoscibile? non-riconoscibile? (b.) L'^- é decidibile? riconoscibile? non-riconoscibile? Dimostra per contraddizione.

Soluzione 7. (a.) L' non é riconoscibile. La prova é per contraddizione. Assumi L' sia riconoscibile. Allora possiamo costruire una TM M che riconosca L^- nel modo seguente. Per ogni input x:

1. M cambia il suo input in $\texttt{1}x$
2. simula l'ipotetica TM per L' su tale input:
   - se tale TM accetta, allora $x \in L^-$, quindi M dovrebbe accettare.
   - se tale TM per L' non accetta mai, allora nemmeno M accetta.

Quindi, M accetta esattamente L^- e ció contraddice l'assunzione che L^- non sia riconoscibile. Concludiamo che L' é non riconoscibile (quindi neppure decidibile).

(b.) Anche L'^- non é riconoscibile. Ancora, la prova é per contraddizione. Assumiamo L'^- sia riconoscibile. Allora, possiamo ideare una TM M che riconosce L' come segue. Dato l'input x:

- se l'input é vuoto, M rigetta
- se l'input é $\texttt{0}x$, eseguiamo l'ipotetica TM per L'^- su $\texttt{1}x$. Se tale TM accetta, allora $\texttt{1}x \in L'^-$, quindi $x \in L$. Conseguentemente, $\texttt{0}x \in L'$ e accetta.
- se l'input é $\texttt{1}x$, eseguiamo l'ipotetica TM per L'^- su $\texttt{0}x$ e accetta se tale macchina accetta.

Abbiamo dunque costruito una TM che riconosce L', contraddicendo la prova precedente che L' non sia riconoscibile. Dunque, L'^{-} deve é non-riconoscibile (quindi nemmeno decidibile).

Problema 8. Considera il problema decisionale

$$Data\ TM\ M,\ M\ ferma\ su\ input\ x\ in\ 100\ passi?$$

e il linguaggio corrispondente

$$L = \{\mathsf{code}(x)\mathsf{code}(M) : TM\ M\ ferma\ su\ x\ in\ 100\ passi\}.$$

L é decidibile? riconoscibile? non-riconoscibile?

Soluzione 8. L é decidibile. Su input $\mathsf{code}(x)\mathsf{code}(M)$ usiamo una UTM per simulare M su input x, modificata in modo che:

1. conta il numero di passi simulati sul nastro
2. la macchina:
   - accetta se la simulazione termina prima che il 100-esimo passo sia raggiunto
   - si ferma e rigetta, se l'esecuzione raggiunge il 100-esimo passo senza fermarsi.

Questa macchina chiaramente riconosce il linguaggio e ferma sempre, quindi L é decidibile.

Problema 9. Quali delle seguenti sono proprietá di linguaggi? Quali sono triviali?

a. $\{y \mid y = \mathsf{code}(M)\ \&\ L_M = L^{-}$ per qualche L finito$\}$
b. $\{y \mid y = \mathsf{code}(M)\ \&$ ferma su qualche input$\}$
c. $\{y \mid y = \mathsf{code}(M)\ \&\ M$ non torna allo stato iniziale$\}$
d. $\{y \mid y = \mathsf{code}(M)\ \&\ ab \in L_M\}$
e. $\{y \mid y = \mathsf{code}(M)\ \&\ L_M$ riconoscibile$\}$

Soluzione 9. (a.) É non-triviale, (b.) É non-triviale, (c.) É non triviale ma non é una proprietá di linguaggio (possiamo avere TM tale che non restituisce mai (o restituisce sempre) lo stato iniziale pur accettando il linguaggio espresso dalla proprietá. (d.) É non-triviale. (e.) É triviale.