Teoremi Informatica Teorica

Zbirciog Ionut Georgian

May 16, 2024

Indice

1 Teoremi Dispensa 2

1.1 Teorema a pag. 5

Per ogni macchina di Turing non deterministica NT esiste una macchina di Turing detreministica T tale che, per ogni possibile input x di NT , l'esito della computazione $NT(x)$ coincide con l'esito della computazione di $T(x)$.

Dimostrazione: Eseguiamo una simulazione della macchina non deterministica NT mediante una macchina deterministica T. La simulazione consiste in una visita in ampiezza[1] dell'albero delle computazioni di NT basata sulla tecnica *coda di rondine con ripetizioni*. Partiamo dallo stato globale $SG(T, x, 0)$ e simuliamo tutte le computazione di lunghezza 1. Se tutte le computazioni terminano in q_R allora T rigetta, se almeno una computazione termina in q_A allora T accetta, altrimenti ricominciamo da capo eseguendo tutte le computazioni di lunghezza 2 e così via.

[1]Perché non in profondità? Non possiamo fare una visità in profondità perché non sappiamo la lunghezza di ciascuna computazione, in quanto potrebbero anche non finire.

2 Teoremi Dispensa 3

2.1 Teorema a pag. 3

Un linguaggio $L \subseteq \Sigma^\star$ è decidibile se e soltanto se L e L^c sono accettabili.

Dimostrazione:

($\Rightarrow$ Se L è decidibile allora esiste una macchina di Turing T deterministica tale che $\forall x \in \Sigma^\star$, $T(x) = q_A \Leftrightarrow x \in L \wedge T(x) = q_R \Leftrightarrow x \in L^c$. Osserviamo dunque che T accetta L.

Da T, deriviamo ora $T^{'}$ aggiungendo le seguenti quintuple:

$$\langle q_A, x, x, q_R^{'}, stop\rangle \wedge \langle q_R, x, x, q_A^{'}, stop\rangle \ \forall x \in \Sigma \cup \square$$

L'esecuzione di $T^{'}$ è simile a quella di T, solo che gli stati di accettazione e rigetto sono stati invertiti, in questo modo se T accetta x allora $T^{'}$ rigetta x, mentre se T rigetta x, $T^{'}$ accetta x, dunque T' accetta L^c.

$\Leftarrow$) Se L e L^c sono accettabili allora esistono due macchine di Turing T_1 e T_2 tali che, $\forall x \in \Sigma^\star\, T_1(x) = q_A \Leftrightarrow x \in L \wedge T_2(x) = q_A \Leftrightarrow x \in L^c$. Non esendo specificato l'esito della computazione nel caso in cui $x \notin L$ e $x \notin L^c$ definiamo la macchina T che, simulando T_1 e T_2 decide L nel seguento modo[2]:

1. Esegui una singola istruzione di T_1 sul nastro 1: se $T_1(x) = q_A$ allora $T(x) = q_A$, altrimenti esegui il passo (2).
2. Esegui una singola istruzione di T_2 sul nastro 2: se $T_2(x) = q_A$ allora $T(x) = q_R$, altrimenti esegui il passo (1).

Se $x \in L$, allora prima o poi, al passo (1), T_1 entrerà nello stato di accettazione, portando T ad accettare. Se $x \in L^c$, allora prima o poi, al passo (1), T_1 entrerà nello stato di accettazione, portando T a rigettare.

2.2 Teorema a pag. 4

Un linguaggio L è decidibile se e soltanto se la funzione χ_L è calcolabile.

Dimostrazione:

($\Rightarrow$ Se L è decidibile allora esiste una macchina di Turing T deterministica di tipo **riconoscitore** tale che $\forall x \in \Sigma^\star$, $T(x) = q_A \Leftrightarrow x \in L \wedge T(x) = q_R \Leftrightarrow x \in L^c$. A partire da T definiamo una macchina di Turing $T^{'}$ di tipo trasduttore a 2 natri, con input $x \in \Sigma^\star$ che opera nel seguente modo:

1. Sul primo nastro simula $T(x)$.
2. Se $T(x)$ termina nello stato q_A allora $T'(x)$ scrive sul nastro di output il valore 1, altrimenti scrive il valore 0 e poi termina.

Osserviamo che poiché L è decidibile il passo (1) termina sempre per ogni input x. Se $x \in L$ allora $T(x) = q_A$ e $T^{'}(x)$ scrive 1 sul nastro di output. Se $x \notin L$ allora $T(x) = q_R$ e $T^{'}(x)$ scrive 0 sul nastro di output. Questo dimostra che χ_L è calcolabile.

$\Leftarrow$) Se χ_L è calcolabile e per costruzione anche totale allora esiste una macchina di Turing T di tipo **trasduttore**, che per ogni $x \in \Sigma^\star$, calcola $\chi_L(x)$. A partire da T definiamo $T^{'}$ di tipo riconoscitore a 2 natri, con input $x \in \Sigma^\star$ che opera nel seguente modo:

1. Sul primo nastro simula $T(x)$ scrivendo il risultato sul secondo nastro.
2. Se sul secondo nastro c'é scritto 1 allora $T^{'}(x) = q_A$, altrimenti nello stato q_R.

Osserviamo che poiché χ_L è calcolabile il passo (1) termina sempre per ogni input x. Se $\chi_L(x) = 1$ allora (1) termina scrivendo 1 sul secondo nastro e $T^{'}(x) = q_A$. Se $\chi_L(x) = 0$ allora (1) termina scrivendo 0 sul secondo nastro e $T^{'}(x) = q_R$. Questo dimostra che L è decidibile.

2.3 Teorema a pag. 5

Se la funzione $f : \Sigma^\star \to \Sigma_1^\star$ è totale e calcolabile allora il linguaggio $L_f \subseteq \Sigma^\star \times \Sigma_1^\star$ è decidibile.

[2] Osserviamo che non possiamo simulare T_1 e T_2 "blackbox", in quanto non sappiamo se la loro computazione termina o meno.

Dimostrazione: Poiché f è calcolabile e totale allora esiste una macchina di Turing trasduttore che calcola $f(x) \forall x \in \Sigma^\star$. A partire da T definiamo una macchina di Turing $T^{'}$ riconoscitore a due nastri con input $\langle x, y \rangle$ dove $x \in \Sigma^\star$ e $y \in \Sigma_1^\star$, che opera nel seguente modo:

1. Sul nastro 1 è scritto l'input $\langle x, y \rangle$.
2. Sul nastro 2 simula $T(x)$, scrivendovi il risultato z.
3. Se $z = y$ allora $T^{'}(x) = q_A$ altrimenti va in q_R.

Osserviamo che, poiché f è totale e calcolabile il passo (2) termina per ogni input $x \in \Sigma\star$. Se $f(x) = z = y$ allora $T^{'}(x)$ termina in q_A. Se $f(x) = z \neq y$ allora $T^{'}(x)$ termina in q_R. Questo dimostra che L_f è decidibile.

2.4 Teorema a pag. 5

Sia $f : \Sigma^\star \to \Sigma_1^\star$ una funzione. Se il linguaggio $L_f \subseteq \Sigma^\star \times \Sigma_1^\star$ è decidibile allora f è calcolabile[3].

Dimostrazione: Poiché $L_f \subseteq \Sigma^\star \times \Sigma_1^\star$ è decidibile, esiste una macchina di Turing riconoscitore T, tale che $\forall x \in \Sigma^\star$ e $\forall y \in \Sigma_1^\star$, $T(x) = q_A$ se $y = f(x)$ e $T(x) = q_R$ se $y \neq f(x)$. A partire da T definiamo una macchina di Turing trasduttore $T^{'}$ con input $x \in \Sigma^\star$ che opera nel seguente modo:

1. Scrive $i = 0$ sul nastro 1.
2. Enumera tutte le stringhe $y \in \Sigma_1^\star$ di lunghezza pari al valore scritto sul primo nastro, simulando per ciascuna stringa $T(x, y)$.
   (a) Sia y la prima stringa di lunghezza i non ancora enumerata, allora scrive y sul secondo nastro.
   (b) Sul terzo nastro, esegue la computazione $T(x, y)$.
   (c) Se $T(x, y) = q_A$ allora scrive y sul nastro di output eventualmente incrementando i se y era l'ultima stringa, torna al passo (2).

Poiché L_f è decidibile il passo (b) termina per ogni input (x, y). Se x appartiene al dominio di f, allora $\exists y \in \Sigma_1^\star$ tale che $y = f(x)$, e quindi $(x, y) \in L_f$. Allora prima o poi la strigna y verrà scritta sul secondo nastro e $T(x, y) = q_A$. Questo dimostra che f è calcolabile.

2.5 Teorema a pag. 7

Per ogni programma scritto in accordo con il linguaggio di programmazione **PascalMinimo**, esiste un macchina di Turing T di tipo trasduttore che scrive sul nastro di output lo stesso valore fornito in output dal programma.

Dimostrazione omessa

2.6 Teorema a pag. 9

Per ogni macchina di Turing deterministica T di tipo riconoscitore ad un nastro esiste un programma P scritto in accordo alle regole del linguaggio **PascalMinimo** tale che, per ogni stringa x, se $T(x)$ termina nello stato fiale $q_F \in \{q_A, q_R\}$ allora P con input x restituisce q_F in output.

Dimostrazione omessa

[3] Osserviamo che non possiamo invertire del tutto il teorema precendente, dalla decidibilità di L_f possiamo dedurre solo la calcolabilità di f

3 Teoremi Dispensa 5

3.1 Teorema a pag. 2

L'insieme T delle macchine di Turing definite sull'alfabeto $\{0,1\}$ e dotate di un singolo nastro (più l'eventuale nastro di output) è numerabile

Dimostrazione: Per dimostrare tale teorema, dobbiamo trovare una biezione tra l'insieme T e l'insieme $\mathbb{N}$. Tale biezione non è altro che una etichettatura degli elementi dell'insieme con etichette appartenenti ad $\mathbb{N}$, ossia, una numerazione degli elementi dell'insieme. Sia T una macchina di Turing e β_T la sua codifica.

Dunque, rappresentiamo T con la parola $\beta_T \in \Sigma^\star$, con $\Sigma = \{0,1,\oplus,\otimes,-,f,s,d\}$ come segue:

$$\beta_T = b(q_0) \;-\; b(q_1) \otimes b(q_{11}) \;-\; b_{11} \;-\; b_{12} \;-\; b(q_{12}) \;-\; m_1 \oplus \cdots \oplus b(q_{h1}) \;-\; b_{h1} \;-\; b_{h2} \;-\; b(q_{h2}) \;-\; m_h$$

Ora, effettuando le seguenti sostituzione in β_T, otteniamo una stringa in $\mathbb{N}$

- "s" con "5"
- "f" con "6"
- "d" con "7"
- "-" con "4"
- "$\otimes$" con "3"
- "$\oplus$" con "2"

Inoltre, dato che la stringa può iniziare con un "0", allora premettiamo il carattere "8" alla stringa ottenuta.

La parola in $\{0,1,2,3,4,5,6,7,8\}^\star$ così ottenuta, può, ovviamente, essere considerata come un numero espresso in notazione decimale, ovvero il numero $v(T) \in \mathbb{N}$ associato univocamente a T.

3.2 Teorema a pag. 4 (Halting Problem)

Definiamo il seguente linguaggio L_H in questo modo:

$$L_H = \{(i,x) : i \text{ è la codifica di una } TM \;\wedge\; T_i(x) \text{ termina}\}$$

Il linguaggio L_H è accettabile.

Dimostrazione: Dobbiamo dimostrare che esiste una macchina di Turing T tale che, per ogni input $(i,x) \in \mathbb{N} \times \mathbb{N}$, $T(i,x) = q_A$ se e soltanto se $(i,x) \in L_H$.
Definiamo U' una macchina di Turing universale modificata con input (i,x). Tale macchina opera nel seguente modo:

1. Verifica se i è la codifica di una macchina di Turing. Se non lo è allora $U'(i,x) = q_R$.
2. Simula $U(i,x)$, se termina in q_A o in q_R allora $U'(x) = q_A$.

U' non sa decidere L_H^c, perciò lo accetta solo.

3.3 Teorema a pag. 4 (Halting Problem)

Il linguaggio L_H non è decidibile

Dimostrazione: Supponiamo che L_H sia decidibile. Allora, deve esistere una macchina di Turing T tale che, $T(i,x) = q_A \Leftrightarrow (i,x) \in L_H$ e $T(i,x) = q_R \Leftrightarrow (i,x) \notin L_H$.

+ Da T **deriviamo** T' che terminando su ogni input, accetta tutte e sole le coppie $(i,x) \in \mathbb{N} \times \mathbb{N} \setminus L_H$, ossia L_H^c. $T'(i,x) = q_R \Leftrightarrow (i,x) \in L_H$ e $T(i,x) = q_A \Leftrightarrow (i,x) \notin L_H$. Quindi $T'(i,x)$ decide L_H^c.
+ Da T' **deriviamo** T'' in questo modo:
  $T''(i,x) =$ *non termina se* $T'(i,x) = q_R$ e $T''(i,x) = q_A$*se* $T'(i,x) = q_A$.
  Quindi $T''(i,x) =$ *non termina se* $(i,x) =\in L_H$ e $T''(i,x) = q_A$ *se* $(i,x) \notin L_H$.

+ Da $T^{''}$ **deriviamo** T^* in questo modo:
  $T^*(i) = T^{''} =$ *non termina se* $(i,i) \in L_H$ e $T^*(i) = T^{''}(i,i) = q_A$ *se* $(i,i) \notin L_H$.

Se T esiste $\Rightarrow T^*$ esiste, allora $\exists k \in \mathbb{N}$ tale che $T^* = T_k$. Se $T_k(k) = T^*(k)$ accettasse, allora $T^{'}(k,k)$ dovrebbe accettare ach'essa. Ma se $T^{'}(k,k)$ accetta, allora $(k,k) \notin L_H$, ossia, $T_k(k)$ non termina. Allora $T^*(k)$ non può accettare e, dunque, necessariamente non termina. Ma, se $T^*(k)$ non termina, allora $T^{'}(k,k)$ rigetta e, quindi, $(k,k) \in L_H$. Dunque, per definizione, $T_k(k)$ termina. Quindi, in entrambi le ipotesi, $T_k(k)$ termina o non termina, portando ad una contraddizione. Allora T^* non può esistere, ma allora neanche $T^{''}$ può esistere, e neanche $T^{'}$ e di conseguenza T. Quindi se T non esiste, L_H non è decidibile.

3.4 Teorema a pag. 6

Se $L_1 e L_2$ sono due linguaggi accettabili, allora $L_1 \cup L_2$ è un linguaggio accettabile.
Se $L_1 e L_2$ sono due linguaggi decidibili, allora $L_1 \cup L_2$ è un linguaggio decidibile.

Dimostrazione:

3.5 Teorema a pag. 6

Se $L_1 e L_2$ sono due linguaggi accettabili, allora $L_1 \cap L_2$ è un linguaggio accettabile.
Se $L_1 e L_2$ sono due linguaggi decidibili, allora $L_1 \cap L_2$ è un linguaggio decidibile.

Dimostrazione:

4 Teoremi Dispensa 5

4.1 Teorema a pag. 3

Sia T una macchina di Turing deterministica, definita su un alfabeto $\Sigma \setminus \Box$ e un insieme di stati Q, e sia $x \in \Sigma^\star$ tale che $T(x)$ termina, allora:

$$dspace(T,x) \leq dtime(T,x) \leq dspace(T,x)|Q|(|\Sigma|+1)^{dspace(T,x)}$$