Elementi di Teoria della Computazione

Classe: Resto_2 - Prof.ssa Marcella Anselmo

Tutorato

01/06/2022 ore 14:00-16:00

Quarta Esercitazione

a cura della dott.ssa Manuela Flores

Esercizio: Relazioni tra L_1 ed L2

Siano L_1 ed L_2 due linguaggi su un alfabeto Σ. Per ognuna delle seguenti affermazioni dire se essa è vera o falsa. È necessario giustificare formalmente la risposta data. Risposte non giustificate non saranno valutate.

(a) *(5 punti)* Se L_1 ed L_2 sono entrambi linguaggi NP-completi, allora $L_1 \leq_m L_2$ ed $L_2 \leq_m L_1$.

(b) *(5 punti)* Se $L_1 \leq_P L_2$ ed $L_2 \leq_P L_1$, allora L_1 ed L_2 sono entrambi linguaggi NP-completi.

(c) *(5 punti)* Se $L_1 \leq_P L_2$ ed L_2 è regolare, allora L_1 è regolare.

Lezione 32 pag. 17

NP - completezza

Vogliamo definire quando un linguaggio B è uno dei linguaggi «più difficili» della classe NP.

Abbiamo visto un modo per definire quando B è «più difficile» di A, ovvero quando A è di difficoltà «minore o uguale» a B:

$$A \leq_p B$$

Quindi B è uno dei linguaggi «più difficili» della classe NP.....

Definizione

Un linguaggio B è *NP-completo* se soddisfa le seguenti due condizioni:

1. B appartiene a NP
2. Per ogni linguaggio A in NP, $A \leq_p B$ (ovvero B è NP-hard)

Lezione 32 pag. 19

NP – completezza: teoremi fondamentali

Ma esistono linguaggi NP-completi?

Ma esistono linguaggi indecidibili?

Come abbiamo dimostrato che A_{TM}, $HALT_{TM}$, E_{TM}, $REGULAR_{TM}$, EQ_{TM} e tanti altri linguaggi sono indecidibili?

Abbiamo provato che:

1. A_{TM} è indecidibile, dalla definizione, per assurdo
2. Se B è indecidibile e $B \leq_m C$ allora anche C è indecidibile

Analogamente dimostreremo che:

1. SAT è NP-completo (Teorema di Cook-Levin)
2. Se B è NP-completo e $B \leq_p C$, con $C \in NP$, allora anche C è NP-completo.

Nota: $\leq_m$ e $\leq_p$

Lezione 32 pag. 21

NP – completezza: teoremi fondamentali

Teorema

Se B è NP-completo e $B \leq_p C$, con $C \in NP$, allora C è NP-completo.

Dimostrazione

Per ipotesi:

1. $C \in NP$
2. Per ogni $A \in NP$, $A \leq_p B$ (B è NP-completo)
3. $B \leq_p C$

Allora, utilizzando la proprietà transitiva di $\leq_p$:

1. $C \in NP$
5. (=2+3) Per ogni $A \in NP$, $A \leq_p C$

Cioè C è NP-completo.

Lezione 28 pag. 6

Risultati

Teorema
$A \leq_m B$ *se e solo se* $\overline{A} \leq_m \overline{B}$.

Teorema
Se $A \leq_m B$ *e* B *è decidibile, allora* A *è decidibile.*

Teorema
Se $A \leq_m B$ *e* B *è Turing riconoscibile, allora* A *è Turing riconoscibile.*

Corollario
Se $A \leq_m B$ *e* A *è indecidibile, allora* B *è indecidibile.*

Corollario
Se $A \leq_m B$ *e* A *non è Turing riconoscibile, allora* B *non è Turing riconoscibile.*

Esercizio: A non si riduce ad E (es. 5.5)

su elearning.informatica.unisa.it

Definire i linguaggi A_{TM} ed E_{TM} .
Mostrare che A_{TM} non è riducibile mediante funziona a E_{TM} .
Suggerimento: ragionare per assurdo e utilizzare risultati noti.

Lezione 25 pag. 36

Un problema indecidibile

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una MdT e } M \text{ accetta } w\}$$

A_{TM} è il linguaggio associato al problema decisionale dell'accettazione di una macchina di Turing.

Teorema

Il linguaggio A_{TM} non è decidibile.

Lezione 27 pag. 28

Problema del vuoto

$$A_{TM} = \{\langle M, w\rangle \mid M \text{ è una TM e } w \in L(M)\}$$

$$E_{TM} = \{\langle M\rangle \mid M \text{ è una } TM \text{ e } L(M) = \emptyset\}$$

Proviamo che

$$A_{TM} \leq_m \overline{E_{TM}}$$

Consideriamo $f : \Sigma^* \to \Sigma^*$ tale che $f(\langle M, w\rangle) = \langle M_1\rangle$ dove M_1 su un input x:

1. Se $x \neq w$ allora M_1 si ferma e rifiuta x.
2. Se $x = w$ allora M_1 simula M su w e accetta x se M accetta w.

Quindi $L(M_1) = \begin{cases} \{w\} & \text{se } \langle M, w\rangle \in A_{TM} \\ \emptyset & \text{altrimenti} \end{cases}$

Lezione 27 pag. 19

Teoremi

Teorema
$A \leq_m B$ *se e solo se* $\overline{A} \leq_m \overline{B}$.

Dimostrazione

Per ipotesi $A \leq_m B$, quindi esiste una riduzione di A a B.

Poiché f è una riduzione, f è calcolabile e inoltre

$$\forall w \in \Sigma^* \quad w \in A \Leftrightarrow f(w) \in B$$

Proviamo che f è anche una riduzione da $\overline{A}$ a $\overline{B}$.

Lezione 26 pag. 12

Insiemi numerabili: Σ^* e MdT

Esempio. L'insieme Σ^* di tutte le stringhe sull'alfabeto Σ è numerabile.

Possiamo elencare le stringhe secondo l'ordine radix, cioè per lunghezza e, a parità di lunghezza, in ordine lessicografico.
Per esempio: $\Sigma = \{0, 1\}$, $w_0 = \epsilon$, $w_1 = 0$, $w_2 = 1$, $w_3 = 00$, ...

Esempio. L'insieme

$$\{\langle M \rangle \mid M \text{ è una MdT sull'alfabeto } \Sigma\}$$

è numerabile.

Abbiamo visto che è possibile codificare le MdT tramite stringhe su un alfabeto (anche binario).
E l'insieme di tutte le stringhe su un alfabeto è numerabile.

Lezione 28 pag. 6

Risultati

Teorema
$A \leq_m B$ se e solo se $\overline{A} \leq_m \overline{B}$.

Teorema
Se $A \leq_m B$ e B è decidibile, allora A è decidibile.

Teorema
Se $A \leq_m B$ e B è Turing riconoscibile, allora A è Turing riconoscibile.

Corollario
Se $A \leq_m B$ e A è indecidibile, allora B è indecidibile.

Corollario
Se $A \leq_m B$ e A non è Turing riconoscibile, allora B non è Turing riconoscibile.

Lezione 27 pag. 30

Indecidibilità del problema del vuoto

Teorema
$\overline{E_{TM}}$ *è indecidibile.*

Infatti $A_{TM} \leq_m \overline{E_{TM}}$ e A_{TM} indecidibile $\Rightarrow$ $\overline{E_{TM}}$ indecidibile.

Corollario E_{TM} è indecidibile.

Infatti la classe dei linguaggi decidibili è chiusa per complemento.

Nota. Non si conosce una riduzione da A_{TM} a E_{TM}.

Esercizio: A non si riduce ad E (es. 5.5)

su elearning.informatica.unisa.it

Definire i linguaggi A_{TM} ed E_{TM} .
Mostrare che A_{TM} non è riducibile mediante funziona a E_{TM} .
Suggerimento: ragionare per assurdo e utilizzare risultati noti.

5.5 Supponiamo per assurdo che $A_{\text{TM}} \leq_{\text{m}} E_{\text{TM}}$ tramite la riduzione f. Dalla definizione di riducibilità mediante funzione segue che $\overline{A_{\text{TM}}} \leq_{\text{m}} \overline{E_{\text{TM}}}$ tramite la stessa funzione di riduzione f. Tuttavia $\overline{E_{\text{TM}}}$ è Turing-riconoscibile (vedere la soluzione dell'Esercizio 4.5) e $\overline{A_{\text{TM}}}$ non è Turing-riconoscibile, contraddicendo il Teorema 5.28.

Teorema

Se $A \leq_m B$ e B è Turing riconoscibile, allora A è Turing riconoscibile.

Esercizio: A non si riduce ad E (es. 5.5)

su elearning.informatica.unisa.it

Definire i linguaggi A_{TM} ed E_{TM} .
Mostrare che A_{TM} non è riducibile mediante funziona a E_{TM} .
Suggerimento: ragionare per assurdo e utilizzare risultati noti.

4.5 Sia $s_1, s_2, \ldots$ la lista di tutte le stringhe in Σ^*. La seguente TM riconosce $\overline{E_{TM}}$.

"Su input $\langle M \rangle$, dove M è una TM:

1. Ripete per $i = 1, 2, 3, \ldots$.
2. Esegue M per i passi su ogni input, $s_1, s_2, \ldots, s_i$.
3. Se M ne ha accettato almeno uno, *accetta*. Altrimenti, continua."

SAT-MON appartiene ad NP

4. Una formula booleana ϕ è monotona se ϕ è una variabile booleana oppure ϕ si ottiene da due formule booleane monotone ϕ_1, ϕ_2, applicando l'operazione AND oppure OR, cioè $\phi = (\phi_1 \vee \phi_2)$ oppure $\phi = (\phi_1 \wedge \phi_2)$. Una formula booleana monotona ϕ è soddisfacibile se esiste un insieme di valori 0 o 1 per le variabili di ϕ che renda la formula uguale a 1.

   (1) Definire il problema della soddisfacibilità di una formula booleana monotona. Definire il linguaggio $SAT\text{-}MON$ associato a tale problema.

   (2) Definire la classe NP. Stabilire se $SAT\text{-}MON$ appartiene alla classe NP, giustificando la risposta.

Lezione 30 pag. 15

La classe *NP*

Definizione

NP è la classe dei linguaggi verificabili in tempo polinomiale.

- Esempi.
- Per *HAMPATH* un certificato per una stringa $\langle G, s, t \rangle \in$ *HAMPATH* è un cammino Hamiltoniano da s a t.
- Per *COMPOSITES* un certificato per una stringa $\langle x \rangle \in$ *COMPOSITES* è uno dei divisori di x.

Nota: *NP* non è l'abbreviazione di tempo Non Polinomiale.
Il nome deriva dalla seguente caratterizzazione.

Lezione 30 pag. 21

Esempi di linguaggi in *NP*

Teorema
SUBSET-SUM $\in$ *NP*

Dimostrazione.
Un algoritmo V che verifica *SUBSET-SUM* in tempo polinomiale:
$V =$ "Sull'input $\langle\langle S, t\rangle, c\rangle$:

1. Verifica se c è un insieme di numeri la cui somma è t, altrimenti rifiuta.
2. Verifica se S contiene tutti i numeri in c, accetta in caso affermativo; altrimenti rifiuta."

$\exists c : \langle\langle S, t\rangle, c\rangle \in L(V) \Leftrightarrow \langle S, t\rangle \in$ *SUBSET-SUM* $\square$

DOMINATING-SET appartiene ad NP

Un sottoinsieme D di vertici di un grafo non orientato $G = (V, E)$ è un insieme dominante per G se ogni vertice in $V \setminus D$ è adiacente a un vertice in D (cioè i due vertici sono connessi mediante un arco in E). Si consideri il seguente problema di decisione:

> Dati un grafo non orientato $G = (V, E)$ e un intero positivo k, esiste un insieme dominante D di cardinalità k?

(a) Definire il linguaggio $DOMINATING\text{-}SET$ associato a tale problema e dimostrare che $DOMINATING\text{-}SET$ è in NP.

Lezione 30 pag. 19

Esempi di linguaggi in *NP*

Teorema
$CLIQUE \in NP$

Dimostrazione.
Un algoritmo V che verifica $CLIQUE$ in tempo polinomiale:
$V =$ "Sull'input $\langle\langle G, k\rangle, c\rangle$:

1. Verifica se c è un insieme di k nodi di G, altrimenti rifiuta.
2. Verifica se per ogni coppia di nodi in c, esiste un arco in G che li connette, accetta in caso affermativo; altrimenti rifiuta."

$\exists c : \langle\langle G, k\rangle, c\rangle \in L(V) \Leftrightarrow \langle G, k\rangle \in CLIQUE$ □

Prova alternativa: utilizzare le macchine di Turing non deterministiche.

Prossimo tutorato

Prima del preappello di giugno:
data da definire, la troverete pianificata su questo canale del Team…
…buono studio
e in bocca al lupo
per la prova intercorso
di mercoledì prossimo ☺