Sol. Turno 1

Esercizio 1.1

Descrivere il concetto di configurazione in un modello di calcolo e la sua utilità nel descrivere una computazione. Si porti ad esempio il modello della macchina di Turing, dicendo cos'è una configurazione per una TM, come si cambia una configurazione in seguito all'esecuzione di una mossa e quindi come si definisce il linguaggio delle parole accettate.

Sol

Una configurazione è la generalizzazione del concetto di stato, dà conto non solo dello stato in cui si trova la macchina, ma anche dello stato in cui si trova la sua memoria, o almeno della porzione di questa che è utile ai fini di definire i successivi passi di calcolo. Infatti possiamo così descrivere un passo di calcolo facendo vedere come cambia la configurazione.
Nel caso di una TM, la configurazione deve dar conto dello stato in cui si trova la macchina, del contenuto del nastro e della posizione della testina di lettura/scrittura. Un modo sintetico di rappresentare questo è di considerare stringhe in $\Gamma^*Q\Gamma^*$, usando q per individuare la posizione della testina di lettura/scrittura e il simbolo in lettura, infatti una stringa del tipo $\alpha qa\beta$ dice che $\alpha a\beta$ è il contenuto del nastro, q è lo stato e la posizione della testina è tale che α è la porzione di nastro a sinistra della testina che legge una cella in cui c'è a.
Possiamo allora descrivere un passo di calcolo:
se $\delta(q,a) = (p,b,R)$ allora $\alpha qa\beta \Rightarrow \alpha bq\beta$
Qui la testina di lettura/scrittura si è spostata a destra, dopo aver scritto b al posto di a nella cella in lettura
se $\delta(q,a) = (p,b,L)$ allora $\alpha cqa\beta \Rightarrow \alpha qcb\beta$
Qui la testina di lettura/scrittura si è spostata a sinistra, dopo aver scritto b al posto di a nella cella in lettura
Considerando poi la chiusura riflessiva e transitiva di => possiamo descrivere il risultato di più passi di computazione per cui
L(T) = {x | x è sull'alfabeto di input di T e $q_0x \Rightarrow^* \alpha q_a\beta$, dove q_0 e q_a sono rispettivamente lo stato iniziale e lo stato di accettazione di T}.

Esercizio 1.2

Sol

Si costruisca una funzione di riduzione da A_{TM} al problema CO={<T> | T è una TM con alfabeto di input {0,1} e w è in L(T) => co(w) è in L(T), dove co(w) è la parola ottenuta da w scambiando ogni 1 con uno 0 e viceversa}
Qui è importante notare che il linguaggio vuoto è un'istanza sì.

Sia R: input <M,w>
output <T>
T: input x

se x = 100 allora accetta
esegui M su w
 se M accetta w allora
  se x = 011 allora accetta altrimenti rifiuta
 se M rifiuta w allora rifiuta

se w è in L(M), e quindi <M,w> è un'istanza sì del problema dell'accettazione, allora L(T) = {100,011} che è un linguaggio che soddisfa la condizione di appartenenza a CO, quindi <T> è in CO.
Se invece w non è in L(M) allora L(T) = {100} e quindi <T> non è in CO.
La riduzione è quindi corretta e CO è indecidibile.

Sol. Turno 2

Esercizio 2.1

Data una TM T deterministica e che si ferma sempre si sa che scambiando lo stato di accettazione con quello di rifiuto si ottiene una TM T' tale che L(T') = ¬L(T). Si spieghi perché questo è vero e perché invece non si ottiene questo risultato se si applica la costruzione a una TM non deterministica che si ferma sempre. Si definisca il linguaggio deciso e quello rifiutato da una TM T che si ferma sempre, nel caso deterministico e non. Questo vuol dire che la classe dei linguaggi riconosciuti da un NTM non è chiusa rispetto al complemento? Si motivi la risposta.

Sol.

Se x è accettata da una TM T deterministica, sarà rifiutata da T' e viceversa perché ogni parola in input è accettata o è rifiutata, quindi il linguaggio deciso da T' è esattamente il complemento di quello deciso da T.
L(T) = {x | x è sull'alfabeto di input di T e $q_0x \Rightarrow^* \alpha q_a \beta$, dove q_0 e q_a sono rispettivamente lo stato iniziale e lo stato di accettazione di T}.
R(T) = {x | x è sull'alfabeto di input di T e $q_0x \Rightarrow^* \alpha q_r \beta$, dove q_0 e q_r sono rispettivamente lo stato iniziale e lo stato di rifiuto di T}.
Evidentemente L(T') = R(T) = ¬L(T).
Nel caso di una NTM invece, per accettare una parola basta che un cammino di computazione termini in una configurazione di accettazione. mentre per rifiutarla occorre che tutti i cammini di computazione portino allo stato di rifiuto o a una configurazione nella quale non ci sono mosse eseguibili.
Quindi una parola potrebbe essere accettata anche se un altro cammino di computazione la rifiuta, dunque invertire gli stati può far sì che una parola sia accettata sia da T che da T' e dunque non è vero che T' decide il complemento di L(T).
Poiché ogni NTM può essere simulata da una TM, la classe dei linguaggi riconosciuti da una NTM è uguale alla classe dei linguaggi riconosciuti da una TM e dunque la chiusura rispetto al complemento può essere considerata per il caso deterministico. Nel caso consideriamo la classe dei

linguaggi decisi da una macchina di Turing abbiamo visto che la costruzione garantisce la chiusura. Nel caso generale invece abbiamo visto che il linguaggio A_{TM} è Turing riconoscibile, ma il suo complemento no, quindi la classe de linguaggi Turing riconoscibili non è chiusa rispetto al complemento.

Esercizio 2.2

Si costruisca una funzione di riduzione da A_{TM} al problema RIP={<T> | T è un TM e se w è in L(T) allora anche ww è in L(T)}
Si può assumere che l'alfabeto di T sia {0,1}.

Sol.

Si tratta di costruire una TM R che calcola la funzione di riduzione. La funzione deve associare a istanze di A_{TM} istanze di RIP in modo tale che <M,w> è in A_{TM} sse <T> è in RIP.
Notiamo che {0,1}* e il linguaggio vuoto sono istanze sì di RIP, ma anche il linguaggio $L= \{0^{2^n} \mid n \geq 0\}$, perché per ogni $z= 0^{2^n}$ in L anche $zz= 0^{2^{n+1}}$ è in L e quindi L è un linguaggio che soddisfa la proprietà. Decidiamo di usare L per le istanze sì.

Sia R: input <M,w>
output <T>

T: input x
se x = 0 allora accetta
esegui M su w
se M accetta w allora
se $x = 0^{2^n}$ per qualche $n \geq 1$ allora accetta altrimenti rifiuta
se M rifiuta w allora rifiuta

se w è in L(M), e quindi <M,w> è un'istanza sì del problema dell'accettazione, allora $L(T) = \{0^{2^n} \mid n \geq 0\}$ che è un linguaggio che soddisfa la condizione di appartenenza a RIP, quindi <T> è in RIP.
Se invece w non è in L(M) allora L(T) = {0} e quindi <T> non è in RIP.
La riduzione è quindi corretta e RIP è indecidibile.
Qui la scelta più semplice è di considerare {0,1}* come istanza sì da associare all'istanza sì di A_{TM}, per ottenerla basta eliminare il controllo su x e accettare ogni parola nel caso M accetti w e sempre {0} altrimenti.

Turno 3

Esercizio 3.1

Si consideri la sequenza di classi di linguaggi: Regolari, Context-free, Decidibili, Turing riconoscibili e si dimostri che sono ognuna contenuta strettamente nella successiva. Infine si esibisca un linguaggio non decidibile, uno non Turing riconoscibile e uno non Turing riconoscibile il cui complemento anche è non Turing riconoscibile.

Prova.

La classe dei linguaggi regolari è contenuta nella classe dei Context-free perchè ogni PDA che non usa la pila è un NFA e quindi accetta un linguaggio regolare. Ogni PDA può essere simulato da una TM che usa un secondo nastro come la pila, quindi ogni linguaggio context-content-free è Turing riconoscibile. Per dimostrare che è decidibile si usa la decidibilità del problema dell'appartenenza per le CFG. Si può costruire una TM T che si ferma sempre per decidere L(G) per una CFG G. La TM T prende in input una stringa x, sull'alfabeto dei terminali della grammatica e utilizza l'algoritmo dell'appartenenza per le CFG per stabilire se x è o no in L(G).
Inoltre si è dimostrato che $\{0^n1^n \mid n\geq 0\}$ è CFL ma non è regolare e che $\{ww \mid w$ è in $\{0,1\}^*\}$ non è CFL ma decidibile. Poiché A_{TM} è Turing riconoscibile ma non decidibile e banalmente ogni linguaggio decidibile è anche Turing riconoscibile si è conclusa la sequenza di contenimenti stretti. Si è dimostrato infine che $\neg A_{TM}$ non è Turing riconoscibile e che EQ_{TM} e $\neg EQ_{TM}$ non sono Turing riconoscibili.

Esercizio 3.2

Si costruisca una funzione di riduzione da A_{TM} al problema SUB=$\{$<T> $\mid$ T è una TM con alfabeto di input $\{0,1\}$ e $L(T)\subseteq\{0^n1^n \mid n\geq 0\}$ e $L(T)\neq\emptyset\}$

Si tratta di costruire una TM R che calcola la funzione di riduzione. La funzione deve associare a istanze di A_{TM} istanze di SUB, in modo tale che <M,w> è in A_{TM} sse <T> è in SUB. Visto che il linguaggio vuoto è un'istanza no, possiamo pensare a una macchina di Turing che accetti proprio $\{0^n1^n \mid n\geq 0\}$ se M accetta w e il vuoto altrimenti.

Sia R: input <M,w>
output <T>
T: input x
esegui M su w
se M accetta w allora
se x = 0^n1^n per qualche n allora accetta altrimenti rifiuta
se M rifiuta w allora rifiuta

se w è in L(M), e quindi <M,w> è un’istanza sì del problema dell’accettazione, allora L(T) = $\{0^n1^n \mid n\geq 0\}$ che è un linguaggio che soddisfa la condizione di appartenenza a SUB, quindi <T> è in SUB. Se invece w non è in L(M) allora L(T) = $\varnothing$ e quindi <T> non è in SUB.

La riduzione è quindi corretta e RIP è indecidibile.