# Testi di esame precedenti a.a. e soluzioni

## 1 Problemi

**Problema 3.1**: Sia $\Sigma$ un alfabeto (finito) e siano $L_1 \subseteq \Sigma^*$ e $L_2 \subseteq \Sigma^*$ due linguaggi decidibili. Dimostrare che $L_1 \cup L_2$ è decidibile.

**Problema 3.2**: Sia $\Sigma$ un alfabeto (finito) e siano $L_1 \subseteq \Sigma^*$ e $L_2 \subseteq \Sigma^*$ due linguaggi accettabili. Dimostrare che $L_1 \cap L_2$ è accettabile.

**Problema 3.3**: Siano $L_1 \subseteq \Sigma^*$ un linguaggio accettabile e $L_2 \subseteq \Sigma^*$ un linguaggio decidibile. Cosa si può dire circa la accettabilità/decidibilità del linguaggio $L = L_1 \cap L_2$? Motivare la risposta con una opportuna dimostrazione.

**Problema 3.4**: Siano $L_1 \subseteq \Sigma^*$ e $L_2 \subseteq \Sigma^*$ due linguaggi decidibili e sia

$$L = \{x \in L_1 : \exists k \in \mathbb{N} \, [\, |x| = 2k \,] \,\} \cup \{x \in L_2 : \exists k \in \mathbb{N} \, [\, |x| = 2k+1 \,] \,\}.$$

Dimostrare che il linguaggio $L$ è decidibile.

**Problema 3.5**: Siano $L_1 \subseteq \Sigma^*$ un linguaggio decidibile e $L_2 \subseteq \Sigma^*$ un linguaggio accettabile. Cosa si può dedurre circa le proprietà di accettabilità/decidibilità di $L = L_1 - L_2^c$?
(Si ricordi che $L_2^c$ è il linguaggio complemento di $L_2$).

**Problema 3.6**: Siano $L_1 \subseteq \mathbb{N} \times \mathbb{N}$ un liguaggio decidibile e $L_H \subseteq \mathbb{N} \times \mathbb{N}$ il linguaggio che definisce l'Halting Problem. Dimostrare se il linguaggio

$$L_2 = L_1 - L_H^c$$

è accettabile (ove, ricordiamo, $L_H^c$ è il linguaggio complemento di $L_H$).

**Problema 3.7**: Sia $\Sigma = \{0,1\}$ e sia $L \subseteq \Sigma^*$ l'insieme delle parole della forma $1^n 0 1^k$ tali che $n \in \mathbb{N}$ è un multiplo di $k \in \mathbb{N}$. Dimostrare che $L$ è decidibile.

**Problema 3.8**: Sia $\Sigma$ un alfabeto finito e siano $L_1, L_2 \subseteq \Sigma^*$ due linguaggi tali che $L_1$ è decidibile e $L_2$ è accettabile. Cosa si può dire circa le proprietà di accettabilità e decidibilità del linguaggio $L = L_1 \cap L_2^c$? Dimostrare la propria affermazione.

## 2 Soluzioni

**Soluzione del problema 3.1**

Indichiamo con $T_1$ la macchina di Turing che decide $L_1$ e con $T_2$ la macchina di Turing che decide $L_2$.
Utilizzando $T_1$ e $T_2$, definiamo ora una nuova macchina di Turing $T$ che decide $L = L_1 \cup L_2$. Indichiamo con $Q_1$ e $Q_2$, rispettivamente, l'insieme degli stati di $T_1$ e l'insieme degli stati di $T_2$ ed assumiamo (senza perdita di generalità) La macchina $T$è definita sull'alfabeto $\Sigma$ e sull'insieme degli stati $Q = Q_1 \cup Q_2 \cup \{q_{A_T}, q_{R_T}\}$, dove $q_{A_T} \notin Q_1 \cup Q_2$ è lo stato di accettazione di $T$ e $q_{R_T} \notin Q_1 \cup Q_2$ è lo stato di rigetto di $T$. $T$ utilizza due nastri, su ciascuno dei quali, all'inizio della computazione, è scritto l'input $x \in \Sigma^*$ ed opera come di seguito descritto.

1) $T$ inizia la sua computazione simulando sul nastro 1 la computazione $T_1(x)$: se tale computazione termina nello stato di accettazione di $T_1$ allora $T$ entra nello stato di accettazione $q_{A_T}$ e termina. Se, invece, $T_1(x)$ non termina nello stato di accettazione allora, poiché $L_1$ è un linguaggio decidibile ed è deciso da $T_1$, $T_1(x)$ termina nello stato di rigetto; in questo secondo caso, viene eseguito il passo 2).

2) $T$ prosegue la sua computazione simulando sul nastro 2 la computazione $T_2(x)$: se tale computazione termina nello stato di accettazione di $T_2$ allora $T$ entra nello stato di accettazione $q_{A_T}$ e termina. Se, invece, $T_2(x)$ non termina nello stato di accettazione allora, poiché $L_2$ è un linguaggio decidibile ed è deciso da $T_2$, $T_2(x)$ termina nello stato di rigetto; in questo secondo caso, $T$ entra nello stato di rigetto $q_{R_T}$ e termina.

Quindi, la macchina $T$ termina per ogni input $x$ e, inoltre, termina nello stato di accettazione se e soltanto se $x \in L_1$ oppure $x \in L_2$, ossia, se e soltanto se $x \in L_1 \cup L_2$. Questo prova che $L$ è decidibile.

**Soluzione del problema 3.2**

Indichiamo con $T_1$ la macchina di Turing che accetta $L_1$ e con $T_2$ la macchina di Turing che accetta $L_2$.
Utilizzando $T_1$ e $T_2$, definiamo ora una nuova macchina di Turing $T$ che accetta $L = L_1 \cap L_2$. Indichiamo con $Q_1$ e $Q_2$, rispettivamente, l'insieme degli stati di $T_1$ e l'insieme degli stati di $T_2$ ed assumiamo (senza perdita di generalità) $Q_1 \cap Q_2 = \emptyset$. Analogamente, siano $q_{01}$ e $q_{02}$, rispettivamente, gli stati iniziali di $T_1$ e $T_2$ e $q_{A1}$ e $q_{A2}$, rispettivamente, gli stati finali di accettazione di $T_1$, e $T_2$ e $q_{R1}$ e $q_{R2}$, rispettivamente, gli stati finali di rigetto di $T_1$ e $T_2$.
La macchina $T$è definita sull'alfabeto $\Sigma$ e sull'insieme degli stati $Q = Q_1 \cup Q_2 \cup \{q_{A_T}, q_{R_T}\}$, dove $q_{A_T} \notin Q_1 \cup Q_2$ è lo stato di accettazione di $T$ e $q_{R_T} \notin Q_1 \cup Q_2$ è lo stato di rigetto di $T$; infine, $q_{01}$ è lo stato iniziale di$T$. $T$ utilizza due nastri, su ciascuno dei quali, all'inizio della computazione, è scritto l'input $x \in \Sigma^*$ ed opera come di seguito descritto.

1) $T$ inizia la sua computazione simulando sul nastro 1 la computazione $T_1(x)$: se tale computazione termina nello stato $q_{R1}$ allora $T$ entra nello stato di rigetto $q_{R_T}$ e termina. Se, invece, $T_1(x)$ termina nello stato $q_{A1}$ allora $T$ entra nello stato $q_{02}$ ed esegue il passo 2).

2) $T$ prosegue la sua computazione simulando sul nastro 2 la computazione $T_2(x)$: se tale computazione termina nello stato di accettazione di $T_2$ allora $T$ entra nello stato di accettazione $q_{A_T}$ e termina. Se, invece, $T_2(x)$ termina nello stato di rigetto, $T$ entra nello stato di rigetto $q_{R_T}$ e termina.

Quindi, la computazione $T(x)$ termina nello stato di accettazione se e soltanto se $x \in L_1 \cap L_2$, dunque $T$ accetta $L_1 \cap L_2$. Osserviamo esplicitamente che nulla si può dire circa l'esito della computazione $T(x)$ per $x \notin L_1 \cap L_2$.

**Soluzione del problema 3.3**

Poiché $L_1$ è accettabile, esiste una macchina di Turing $T_1$ tale che, per ogni $x \in \Sigma^*$, $T_1(x)$ accetta se e soltanto se $x \in L_1$; inoltre, poiché $L_2$ è decidibile, esiste una macchina di Turing $T_2$ tale che $L_2 = L(T_2)$.
Mostriamo ora che $L$ è un linguaggio accettabile. La macchina $T$ che accetta $L$ opera come di seguito descritto. Con input $x \in \Sigma^*$, $T$ esegue, inizialmente, la computazione $T_2(x)$ (che termina sempre): se tale computazione termina nello stato di rigetto, allora $x \notin L_2$ e, dunque, $x \notin L$. Se, invece, $T_2(x)$ termina nello stato di accettazione, allora $x \in L$ se e soltanto se $x \in L_1$: allora, la computazione $T(x)$ prosegue eseguendo la computazione $T_1(x)$ e, se essa termina

nello stato di accettazione, allora anche $T(x)$ termina nello stato di accettazione. Quindi, $T$ accetta $L$. Osserviamo esplicitamente che la computazione $T_1(x)$ potrebbe non terminare e, quindi, l'esistenza della macchina $T$ non permette di affermare che $L$ è decidibile.

In effetti, è anche possibile mostrare un esempio in cui il linguaggio intersezione fra un linguaggio accettabile ed uno decidibile è non decidibile. Infatti, siano $L_H$ il linguaggio che definisce l'halting problem ed $L_2 = \Sigma^*$: ricordiamo che $L_H$ è accettabile ma non decidibile e, inoltre, osserviamo che $L_2$ è banalmente decidibile (è sufficiente utilizzare una macchina di Turing che accetta ogni input). Allora, $L_1 \cap L_2 = L_H$.

**Soluzione del problema 3.4**

Poiché $L_1$ è decidibile, esiste una macchina di Turing $T_1$ tale che$L_1 = L(T_1)$; inoltre, poiché anche $L_2$ è decidibile, esiste una macchina di Turing $T_2$ tale che $L_2 = L(T_2)$. Assumiamo, senza perdita di generalità, che $T_1$ e $T_2$ siano macchine dotate di un solo nastro e che l'insieme degli stati di $T_1$ e di $T_2$ siano disgiunti. Indichiamo con $q_{0_1}$ e $q_{0_2}$, rispettivamente, lo stato iniziale di $T_1$ e lo stato iniziale di $T_2$. Analogamente, indichiamo con $q_{A_1}, q_{A_2}, q_{R_1}, q_{R_2}$ gli stati finali (di accettazione e di rigetto) di $T_1$ e di $T_2$.
Mostriamo ora che $L$ è un linguaggio decidibile. La macchina $T$ che decide $L$ utilizza un solo nastro e quattro nuovi stati: lo stato iniziale $q_0$, uno stato intermedio $q_1$, lo stato finale di accettazione $q_A$ e lo stato finale di rigetto $q_R$. Con input $x \in \Sigma^*$, $T$ esegue, inizialmente, una computazione che le permette di decidere se $x$ ha lunghezza pari o dispari:

- se nello stato $q_0$ legge un simbolo diverso da $\square$ riscrive tale simbolo, entra nello stato $q_1$ e muove la testina a destra di una posizione;
- se nello stato $q_1$ legge un simbolo diverso da $\square$ riscrive tale simbolo, entra nello stato $q_0$ e muove la testina a destra di una posizione.

Quando la macchina incontra il simbolo $\square$ ha terminato la scansione dell'input: se si trova nello stato $q_0$ allora l'input ha lunghezza pari e, per decidere l'appartenenza a $L$, è necessario decidere l'appartenenza ad $L_1$ mentre se si trova nello stato $q_1$ allora l'input ha lunghezza dispari e, per decidere l'appartenenza a $L$, è necessario decidere l'appartenenza ad $L_2$. Dunque, quando la macchina incontra il simbolo $\square$

- se è nello stato $q_0$ muove la testina a sinistra fino a riportarla sul primo simbolo di $x$ ed entra nello stato $q_{0_1}$,
- se è nello stato $q_1$ muove la testina a sinistra fino a riportarla sul primo simbolo di $x$ ed entra nello stato $q_{0_2}$.

Infine, se $T$ entra nello stato $q_{A_1}$ oppure nello stato $q_{A_2}$ allora esegue un'ultima istruzione che la porta nello stato $q_A$, altrimenti se $T$ entra nello stato $q_{R_1}$ oppure nello stato $q_{R_2}$ allora esegue un'ultima istruzione che la porta nello stato $q_R$.

**Soluzione del problema 3.5**

Si osservi che $L = L_1 - L_2^c = L_1 \cap L_2$.

Sia $T_1$ la macchina di Turing che decide $L_1$ e sia $T_2$ la macchina di Turing che accetta $L_2$. Definiamo una macchina di Turing $T$ che, con input $x \in \Sigma^*$, opera come segue:

1) simula la computazione $T_1(x)$: se rigetta (e, dunque, $x \notin L_1$) allora rigetta, altrimenti (se $x \in L_1$) esegue il passo 2);
2) simula la computazione $T_2(x)$: se accetta (e, dunque, $x \in L_2$) allora accetta, se rigetta (e, dunque, $x \notin L_2$) allora rigetta.

Poiché $L_1$ è un linguaggio decidibile, il passo 1) termina per ogni $x \in \Sigma^*$. Poiché $L_2$ è un linguaggio accettabile, il passo 2) temina per ogni $x \in L_2$ (e nulla si può dire se $x \notin L_2$). Quindi, se $x \in L_1 \cap L_2 = L$ la computazione $T(x)$ termina nello stato di accettazione (e nulla si può dire se $x \notin L$), ossia, $L$ è un linguaggio accettabile.

### Soluzione del problema 3.6

Si osservi, innanzi tutto, che $L_2 = L_1 \cap L_H$. Ricordiamo, inoltre, che $L_H$ è un linguaggio accettabile.
Siano $T_1$ e $T_H$, rispettivamente, la macchina di Turing che accetta $L_1$ e la macchina di Turing che accetta $L_H$. Descriviamo, ora, una macchina $T_2$ che utilizza 3 nastri e simula le due macchine $T_1$ e $T_H$ operando in due fasi, come di seguito descritto. Con input $x$ scritto sul primo nastro,

**fase 1)** simula $T_1(x)$ sul secondo nastro: se la computazione $T_1(x)$ rigetta allora anche la macchina $T_2$ rigetta, mentre se la computazione $T_1(x)$ accetta allora $T_2$ esegue la fase 2;

**fase 2)** simula $T_H(x)$ sul terzo nastro: se la computazione $T_H(x)$ rigetta allora anche la macchina $T_2$ rigetta, mentre se la computazione $T_H(x)$ accetta allora anche $T_H$ accetta.

Dimostriamo che la macchina $T_2$ appena descritta accetta $L_2$. Sia $x \in L_2$: allora, per quanto osservato, $x \in L_1$ e $x \in L_H$. Allora, poiché $T_1$ accetta $L_1$ e $x \in L_1$, la prima fase termina quando $T_1(x)$ raggiunge lo stato di accettazione e, quindi, per definizione della macchina $T_2$, ha inizio la fase 2. Poiché $T_H$ accetta $L_H$ e $x \in L_H$, la seconda fase termina quando $T_H(x)$ raggiunge lo stato di accettazione e, quindi, per definizione della macchina $T_2$, anche $T_2(x)$ accetta.

Osserviamo, infine, che, poiché $L_1$ è decidibile, la fase 1 di qualunque computazione della macchina $T_2$ termina per ogni $x \in \mathbb{N} \times \mathbb{N}$. D'altra parte, poiché $L_H$ è un linguaggio non decidibile, esiste $y \in \mathbb{N} \times \mathbb{N}$ tale che $T_H(y)$ non termina: pertanto, se esiste $z \in L_1$ tale che $T_H(z)$ non termina, allora $T_2(z)$ non termina. Pertanto, nulla è possibile concludere circa la decidibilità di $L_2$.

### Soluzione del problema 3.7

Dimostriamo la decidibilità di $L$ descrivendo una macchina di Turing $T$, a due nastri e a testine indipendenti, che lo decide.
L'alfabeto di lavoro di $T$ è lo stesso $\Sigma = \{0,1\}$ ed il suo insieme degli stati è $Q = \{q_0, q_1, q_2, q_A, q_R\}$, ove $q_0$ è lo stato iniziale, $q_A$ lo stato di accettazione e $q_R$ lo stato di rigetto.
All'inizio della computazione, l'input è scritto sul nastro $N_1$ mentre il nastro $N_2$ è vuoto.
Descriviamo, ora, le quintuple di $T$ illustrando, contestualmente, una computazione della macchina.

1) la macchina copia sul nastro $N_2$ il contenuto del nastro $N_1$, cancellandolo da $N_1$, fino a quando non incontra il carattere '0' (ossia, se l'input appartiene ad $L$, copia $1^n$ sul nastro $N_2$):
   $\langle q_0, (1,\square), (\square,1), q_0, (d,d)\rangle$, $\langle q_0, (0,\square), (\square,\square), q_1, (d,s)\rangle$.
   A questo punto, la testina del nastro $N_1$ è posizionata sul carattere a destra del primo carattere '0' che ha incontrato, e la testina del nastro $N_2$ è posizionata sul carattere '1' più a destra che vi ha scritto. Inoltre, se l'input appartiene ad $L$, sul nastro $N_1$ è scritto $1^k$, ossia, una parola non contenente il carattere '0' (poiché l'input è una parola di $\Sigma^*$ e quindi non può contenere il carattere '$\square$').

2) La macchina verifica che la parola sul nastro $N_1$ non contenga '0' e che $n$ sia un multiplo di $k$ mediante il seguente procedimento:
   *fino a quando non legge '$\square$' sul nastro $N_1$*
   *prova a cancellare dal nastro $N_2$ tanti '1' quanti ne incontra sul nastro $N_1$:*
   *se non vi riesce (perché non sono rimasti abbastanza '1' sul nastro $N_2$), allora rigetta.*
   Tale procedimento è implementato dalle quintuple seguenti:
   $\langle q_1, (1,1), (1,\square), q_1, (d,s)\rangle$ (cancella un '1' da $N_2$ se legge '1' sia su $N_1$ che su $N_2$)
   $\langle q_1, (\square,1), (\square,1), q_2, (s,f)\rangle$ (si predispone a riposizionare la testina sul nastro $N_1$)
   $\langle q_2, (1,1), (1,1), q_2, (s,f)\rangle$, $\langle q_2, (\square,1), (\square,1), q_1, (d,f)\rangle$ (ha riposizionato la testina sul nastro $N_1$ sul carattere '1' più a sinistra e si predispone ad eseguire una nuova sequenza di cancellazioni sul nastro $N_2$)
   $\langle q_1, (0,x), (0,x), q_R, (f,f)\rangle$ (la parola su $N_1$ contiene uno '0' e, quindi, l'input non appartiene al linguaggio)
   $\langle q_1, (1,\square), (1,\square), q_R, (f,f)\rangle$ ($n$ non è multiplo di $k$)
   $\langle q_1, (\square,\square), (\square,\square), q_A, (f,f)\rangle$ ($n$ è multiplo di $k$).

**Soluzione del problema 3.8**

Poiché $L_1$ è decidibile, esiste una macchina di Turing $T_1$ (di tipo riconoscitore) tale che, per ogni $x \in \Sigma^*$, $T_1(x)$ termina ed inoltre

$$o_{T_1}(x) = \begin{cases} q_A^1 & \text{se } x \in L_1. \\ q_R^1 & \text{se } x \notin L_1, \end{cases}$$

dove $q_A^1$ e $q_R^1$ sono, rispettivamente, gli stati di accettazione e di rigetto di $T_1$.
Analogamente, poiché $L_2$ è accettabile, esiste una macchina di Turing $T_2$ (di tipo riconoscitore) tale che, per ogni $x \in L_2$, $T_2(x)$ termina ed inoltre $o_{T_2}(x) = q_A^2$ dove $q_A^2$ è lo stato di accettazione di $T_2$; osserviamo che nulla si può affermare circa le computazioni $T_2(y)$ con $y \notin L_2$.

Osserviamo, ora, che per poter affermare "$x \in L$" è necessario mostrare che $x \in L_1$ **e** $x \notin L_2$: poiché per affermare $x \notin L_2$ è necessario disporre di una macchina di Turing che accetti $L_2^c$. Dunque, la sola accettabilità di $L_2$ non è sufficiente ad assicurare la accettabilità di $L$.

Osserviamo, infine, che invece è accettabile il linguaggio $L^c = (L_1 \cap L_2^c)^c = L_1^c \cup L_2$: infatti, tale linguaggio è accettato dalla macchina che, con input $x \in \Sigma^*$, esegue i passi seguenti

a) simula la computazione $T_1(x)$: se $o_{T_1}(x) = q_A^1$, allora accetta, altrimenti esegue il passo b);

b) simula la computazione $T_2(x)$: se $o_{T_1}(x) = q_A^2$, allora accetta.