2. *Dimostra che il seguente linguaggio è indecidibile:*

$$A_{1010} = \{\langle M \rangle \mid M \text{ è una TM tale che } 1010 \in L(M)\}.$$

Dimostriamo che A_{1010} è un linguaggio indecidibile mostrando che A_{TM} è riducibile ad A_{1010}. La funzione di riduzione f è calcolata dalla seguente macchina di Turing:

$F =$ "su input $\langle M, w \rangle$, dove M è una TM e w una stringa:

1. Costruisci la seguente macchina M_w:
   $M_w =$ "su input x:
   1. Se $x \neq 1010$, rifiuta.
   2. Se $x = 1010$, esegue M su input w.
   3. Se M accetta, *accetta.*
   4. Se M rifiuta, *rifiuta.*"
2. Restituisci $\langle M_w \rangle$."

Dimostriamo che f è una funzione di riduzione da A_{TM} ad A_{1010}.

- Se $\langle M, w \rangle \in A_{TM}$ allora la TM M accetta w. Di conseguenza la macchina M_w costruita dalla funzione accetta la parola 1010. Quindi $f(\langle M, w \rangle) = \langle M_w \rangle \in A_{1010}$.
- Viceversa, se $\langle M, w \rangle \notin A_{TM}$ allora la computazione di M su w non termina o termina con rifiuto. Di conseguenza la macchina M_w rifiuta 1010 e $f(\langle M, w \rangle) = \langle M_w \rangle \notin A_{1010}$.

Per concludere, siccome abbiamo dimostrato che $A_{TM} \leq_m A_{1010}$ e sappiamo che A_{TM} è indecidibile, allora possiamo concludere che A_{1010} è indecidibile.

2. Dimostra che il seguente linguaggio è indecidibile:

$$L_2 = \{\langle M, w \rangle \mid M \text{ accetta la stringa } ww^R\}.$$

Dimostriamo che L2 = {⟨M, w⟩ | M accetta la stringa ww^R} è indecidibile:

Useremo una riduzione dal problema dell'arresto A_TM = {⟨M, w⟩ | M si ferma su input w}.

Definiamo una funzione f che mappa ⟨M, w⟩ in ⟨M', w⟩, dove M' è una TM che:

1. Su input x: a. Se x non ha la forma yy^R, rifiuta. b. Altrimenti, estrae y dalla prima metà di x. c. Simula M su y. d. Se M si ferma su y, accetta. Altrimenti, cicla.

Ora, ⟨M, w⟩ ∈ A_TM se e solo se ⟨M', w⟩ ∈ L2. Infatti, M' accetta ww^R se e solo se M si ferma su w.

Poiché A_TM è indecidibile e abbiamo una riduzione da A_TM a L2, concludiamo che L2 è indecidibile.

2. (12 punti) Data una Turing Machine M, definiamo

$$\text{HALTS}(M) = \{w \mid M \text{ termina la computazione su } w\}.$$

Considera il linguaggio

$$F = \{\langle M \rangle \mid \text{HALTS}(M) \text{ è un insieme finito}\}.$$

Dimostra che F è indecidibile.

Soluzione. La seguente macchina G calcola una riduzione $\overline{A_{TM}} \leq_m F$:

$G =$ "su input $\langle M, w \rangle$, dove M è una TM e w una stringa:

1. Costruisci la seguente macchina M':
   $M' =$ "Su input x:
   1. Esegue M su input w.
   2. Se M accetta, *accetta.*
   3. Se M rifiuta, va in loop."
2. Ritorna $\langle M' \rangle$."

Mostriamo che G calcola una funzione di riduzione g da $\overline{A_{TM}}$ a F, cioè una funzione tale che

$$\langle M, w \rangle \in \overline{A_{TM}} \text{ se e solo se } M' \in F.$$

- Se $\langle M, w\rangle \in \overline{A_{TM}}$ allora la macchina M su input w rifiuta oppure va in loop. In entrambi i casi la macchina M' va in loop su tutte le stringhe, quindi $\text{HALTS}(M) = \emptyset$ che è un insieme finito. Di conseguenza $M' \in F$.
- Viceversa, se $\langle M, w\rangle \notin \overline{A_{TM}}$ allora la macchina M accetta w. In questo caso la macchina M' accetta tutte le parole, quindi $\text{HALTS}(M) = \Sigma^*$ che è un insieme infinito. Di conseguenza $M' \notin F$.

2. (12 punti) Data una Turing Machine M, definiamo

$$\text{HALTS}(M) = \{w \mid M \text{ termina la computazione su } w\}.$$

Considera il linguaggio

$$I = \{\langle M\rangle \mid \text{HALTS}(M) \text{ è un insieme infinito}\}.$$

Dimostra che I è indecidibile.

Soluzione. La seguente macchina F calcola una riduzione mediante funzione $A_{TM} \leq_m I$:

$F =$ "su input $\langle M, w\rangle$, dove M è una TM e w una stringa:

1. Costruisci la seguente macchina M':

   $M' =$ "Su input x:
   1. Esegue M su input w.
   2. Se M accetta, *accetta*.
   3. Se M rifiuta, va in loop."

2. Ritorna $\langle M'\rangle$."

Mostriamo che F calcola una funzione di riduzione f da A_{TM} a I, cioè una funzione tale che

$$\langle M, w\rangle \in A_{TM} \text{ se e solo se } M' \in I.$$

- Se $\langle M, w\rangle \in A_{TM}$ allora la macchina M accetta w. In questo caso la macchina M' accetta tutte le parole, quindi $\text{HALTS}(M) = \Sigma^*$ che è un insieme infinito. Di conseguenza $M' \in I$.
- Viceversa, se $\langle M, w\rangle \notin A_{TM}$, allora la macchina M su input w rifiuta oppure va in loop. In entrambi i casi la macchina M' va in loop su tutte le stringhe, quindi $\text{HALTS}(M) = \emptyset$ che è un insieme finito. Di conseguenza $M' \notin I$.

2. (12 punti) Considera il linguaggio

$$\textsc{Forty-Two} = \{\langle M, w\rangle \mid M \text{ termina la computazione su } w \text{ avendo solo 42 sul nastro}\}.$$

Dimostra che $\textsc{Forty-Two}$ è indecidibile.

Useremo una riduzione dal problema della fermata (Halting Problem), che sappiamo essere indecidibile.

Sia H = {⟨M, w⟩ | M è una TM che si ferma su input w}

Costruiamo una funzione di riduzione f da H a Forty-Two:

f(⟨M, w⟩) = ⟨M'⟩, dove M' è una TM che:

1. Simula M su input w
2. Se M si ferma, M' cancella il suo nastro e scrive 42
3. Se M non si ferma, M' continua a girare all'infinito

Ora, ⟨M, w⟩ ∈ H se e solo se ⟨M'⟩ ∈ Forty-Two.

Se Forty-Two fosse decidibile, potremmo decidere H usando questa riduzione, che è una contraddizione. Quindi Forty-Two è indecidibile.

2. (12 punti) Considera il linguaggio

$$\text{EVEN-HALTS} = \{\langle M\rangle \mid \text{per ogni numero naturale } n \text{ pari, } M \text{ termina la computazione su } n\}.$$

Dimostra che EVEN-HALTS è indecidibile.

Useremo una riduzione dal problema della fermata (A_TM), che sappiamo essere indecidibile.

Sia f una funzione che mappa ⟨M, w⟩ a ⟨M'⟩, dove M' è definita come segue:

M' = "Su input n:

1. Se n è dispari, entra in un loop infinito
2. Se n è pari, simula M su input w
3. Se M si ferma su w, M' si ferma
4. Se M non si ferma su w, M' entra in un loop infinito"

Ora, ⟨M, w⟩ ∈ HALTS se e solo se ⟨M'⟩ ∈ EVEN-HALTS.

Se EVEN-HALTS fosse decidibile, potremmo decidere A_TM usando questa riduzione, che è una contraddizione. Quindi EVEN-HALTS è indecidibile.

3 **3. (12 punti)** Considera le stringhe sull'alfabeto $\Sigma = \{1, 2, \ldots, 9\}$. Una stringa w di lunghezza n su Σ si dice *ordinata* se $w = w_1 w_2 \ldots w_n$ e tutti i caratteri $w_1, w_2, \ldots, w_n \in \Sigma$ sono tali che $w_1 \leq w_2 \leq \cdots \leq$ le w_n. Ad esempio, la stringa 1112778 è ordinata, ma le stringhe 5531 e 44427 non lo sono (la stringa vuota viene considerata ordinata). Diciamo che una Turing machine è *ossessionata dall'ordinamento* se ogni stringa che accetta è ordinata (ma non è necessario che accetti tutte queste stringhe). Considera il problema di determinare se una TM con alfabeto $\Sigma = \{1, 2, \ldots, 9\}$ è ossessionata dall'ordinamento.

(a) Formula questo problema come un linguaggio SO_{TM}.

(b) Dimostra che il linguaggio SO_{TM} è indecidibile.

Soluzione.

(a) $SO_{\text{TM}} = \{\langle M\rangle \mid M$ è una TM con alfabeto $\Sigma = \{1, 2, \ldots, 9\}$ che accetta solo parole ordinate$\}$

(b) La seguente macchina F calcola una riduzione $\overline{A_{TM}} \leq_m UA$:

$F =$ "su input $\langle M, w\rangle$, dove M è una TM e w una stringa:

1. Costruisci la seguente macchina M':

   $M' =$ "Su input x:
   1. Se $x = 111$, accetta.
   2. Se $x = 211$, esegue M su input w e ritorna lo stesso risultato di M.
   3. In tutti gli altri casi, rifiuta.

2. Ritorna $\langle M'\rangle$."

$\langle M, w\rangle \in A_{TM}$ se e solo se $\langle M'\rangle \in UA$.

Mostriamo che F calcola una funzione di riduzione da $\overline{A_{TM}}$ a SO_{TM}, cioè una funzione tale che

$$\langle M, w\rangle \in \overline{A_{TM}} \text{ se e solo se } \langle M'\rangle \in SO_{\text{TM}}.$$

- Se $\langle M, w\rangle \in \overline{A_{TM}}$ allora la macchina M rifiuta o va in loop su w. In questo caso la macchina M' accetta la parola ordinata 111 e rifiuta tutte le altre, quindi è ossessionata dall'ordinamento e di conseguenza $\langle M'\rangle \in SO_{\text{TM}}$.
- Viceversa, se $\langle M, w\rangle \notin \overline{A_{TM}}$ allora la macchina M accetta w. Di conseguenza, la macchina M' accetta sia la parola ordinata 111 che la parola non ordinata 211, quindi non è ossessionata dall'ordinamento. Di conseguenza $\langle M'\rangle \notin SO_{\text{TM}}$.

3. (12 punti) Data una Turing Machine M, considera il problema di determinare se esiste un input tale che M scrive "$xyzzy$" su cinque celle adiacenti del nastro. Puoi assumere che l'alfabeto di input di M non contenga i simboli x, y, z.

(a) Formula questo problema come un linguaggio $MAGIC_{\text{TM}}$.

(b) Dimostra che il linguaggio $MAGIC_{\text{TM}}$ è indecidibile.

Soluzione.

(a) $MAGIC_{\text{TM}} = \{\langle M \rangle \mid M \text{ è una TM che scrive } xyzzy \text{ sul nastro per qualche input } w\}$

(b) La seguente macchina F calcola una riduzione $A_{\text{TM}} \leq_m MAGIC_{\text{TM}}$:

$F =$ "su input $\langle M, w \rangle$, dove M è una TM e w una stringa:

1. Verifica che i simboli x, y, z non compaiano in w, né nell'alfabeto di input o nell'alfabeto del nastro di M. Se vi compaiono, sostituiscili con tre nuovi simboli X, Y, Z nella parola w e nella codifica di M.
2. Costruisci la seguente macchina M':

   $M' =$ "Su input x:
   1. Simula l'esecuzione di M su input w, senza usare i simboli x, y, z.
   2. Se M accetta, scrivi $xyzzy$ sul nastro, altrimenti rifiuta senza modificare il nastro.
3. Ritorna $\langle M' \rangle$."

Mostriamo che F calcola una funzione di riduzione da A_{TM} a $MAGIC_{\text{TM}}$, cioè una funzione tale che

$$\langle M, w \rangle \in A_{\text{TM}} \text{ se e solo se } \langle M' \rangle \in MAGIC_{\text{TM}}.$$

- Se $\langle M, w \rangle \in A_{\text{TM}}$ allora la macchina M accetta w. In questo caso la macchina M' scrive $xyzzy$ sul nastro per tutti gli input. Di conseguenza $\langle M' \rangle \in MAGIC_{\text{TM}}$.
- Viceversa, se $\langle M, w \rangle \notin A_{TM}$ allora la macchina M rifiuta o va in loop su w. Per tutti gli input, la macchina M' simula l'esecuzione di M su w senza usare i simboli x, y, z (perché sono stati tolti dalla definizione di M e di w se vi comparivano), e rifiuta o va in loop senza scrivere mai $xyzzy$ sul nastro. Di conseguenza $\langle M' \rangle \notin MAGIC_{\text{TM}}$.

4. (9 punti) Considera il seguente problema: data una TM M a nastro semi-infinito, determinare se esiste un input w su cui M sposta la testina a sinistra partendo dalla cella numero 2023 (ossia se in qualche momento durante la computazione la testina si muove dalla cella 2023 alla cella 2022).

(a) Formula questo problema come un linguaggio 2023_{TM}.

(b) Dimostra che il linguaggio 2023_{TM} è indecidibile.

Per dimostrare che 2023_TM è indecidibile, riduciamo il problema della fermata A_TM a 2023_TM.

Sia R un decisore ipotetico per 2023_TM. Costruiamo un decisore S per A_TM:

S = "Su input ⟨M,w⟩:

1. Costruisci una TM M' che:
    - Scrive w sul nastro
    - Si sposta alla cella 2023
    - Simula M su w
    - Se M si ferma, M' si sposta alla cella 2022
2. Esegui R su ⟨M'⟩
3. Se R accetta, accetta. Se R rifiuta, rifiuta."

S decide correttamente A_TM perché:

- M si ferma su w ⇔ M' si sposta dalla cella 2023 alla 2022 ⇔ R accetta ⟨M'⟩ ⇔ S accetta ⟨M,w⟩

Ma sappiamo che A_TM è indecidibile, quindi 2023_TM deve essere indecidibile

4. (9 punti) Una Turing Machine *somma correttamente* se, dati in input due numeri binari separati da $\#$, termina la computazione con la loro somma (in binario) sul nastro. (Non importa cosa fa sugli altri input.) Considera il problema di determinare se una TM somma correttamente.

(a) Formula questo problema come un linguaggio SUM_{TM}.

(b) Dimostra che il linguaggio SUM_{TM} è indecidibile.

(a) Formuliamo SUM_TM come: SUM_TM = {⟨M⟩ | M è una TM che, dati due numeri binari x e y separati da #, termina con x+y (in binario) sul nastro}

(b) Per dimostrare che SUM_TM è indecidibile, riduciamo A_TM a SUM_TM.

Sia R un decisore ipotetico per SUM_TM. Costruiamo un decisore S per A_TM:

S = "Su input ⟨M,w⟩:

1. Costruisci una TM M' che:
    - Su input x#y:
        - Ignora x e y
        - Simula M su w
        - Se M si ferma, calcola x+y e lo scrive sul nastro

2. Esegui R su ⟨M'⟩
3. Se R accetta, accetta. Se R rifiuta, rifiuta."

S decide correttamente A_TM perché:

- M si ferma su w ⇔ M' somma correttamente per ogni input ⇔ R accetta ⟨M'⟩ ⇔ S accetta ⟨M,w⟩

Ma sappiamo che A_TM è indecidibile, quindi SUM_TM deve essere indecidibile.

4. (9 punti) Una Turing Machine *moltiplica correttamente* se, dati in input due numeri binari separati da #, termina la computazione con la loro moltiplicazione (in binario) sul nastro. (Non importa cosa fa sugli altri input.) Considera il problema di determinare se una TM moltiplica correttamente.

(a) Formula questo problema come un linguaggio MUL_{TM}.

(b) Dimostra che il linguaggio MUL_{TM} è indecidibile.

6. (a) Formuliamo MUL_TM come: MUL_TM = {⟨M⟩ | M è una TM che, dati due numeri binari x e y separati da #, termina con x*y (in binario) sul nastro}

(b) Per dimostrare che MUL_TM è indecidibile, riduciamo A_TM a MUL_TM.

Sia R un decisore ipotetico per MUL_TM. Costruiamo un decisore S per A_TM:

S = "Su input ⟨M,w⟩:

1. Costruisci una TM M' che:
    - Su input x#y:
        - Ignora x e y
        - Simula M su w
        - Se M si ferma, calcola x*y e lo scrive sul nastro
2. Esegui R su ⟨M'⟩
3. Se R accetta, accetta. Se R rifiuta, rifiuta."

S decide correttamente HALT_TM perché:

- M si ferma su w ⇔ M' moltiplica correttamente per ogni input ⇔ R accetta ⟨M'⟩ ⇔ S accetta ⟨M,w⟩

Ma sappiamo che A_TM è indecidibile, quindi MUL_TM deve essere indecidibile.

2. (12 punti) Una stringa w è *palindroma* se rimane uguale letta da sinistra a destra e da destra a sinistra, cioè se $w = w^R$. Un linguaggio $B \subseteq \{0,1\}^*$ è *quasi-palindromo* se contiene al più una stringa non palindroma. Ad esempio, sia $\{00, 11011, 1001\}$ che $\{00, 101\}$ sono linguaggi quasi-palindromi, mentre $\{00, 10, 100\}$ non lo è. Considera il problema di determinare se il linguaggio di una TM M è quasi-palindromo.

(a) Formula questo problema come un linguaggio $QPAL_{TM}$.

(b) Dimostra che il linguaggio $QPAL_{TM}$ è indecidibile.

a) Formulazione come linguaggio: QPALTM = {w | w è una stringa quasi-palindroma su {0,1}*} dove una stringa è quasi-palindroma se rimane uguale letta da sinistra a destra e da destra a sinistra, eccetto al più una stringa.

(b) Dimostrazione che QPALTM è indecidibile: Riduzione dal problema della fermata (A_TM). Costruiamo una TM M' che, dato input ⟨M,w⟩:

1. Simula M su w.
2. Se M si ferma su w, M' produce l'output 1001001.
3. Se M non si ferma, M' entra in un loop infinito.

Ora, definiamo una funzione f che mappa ⟨M,w⟩ in 1001001. f è una riduzione da A_TM a QPALTM perché:

- Se M si ferma su w, f(⟨M,w⟩) = 1001001 ∈ QPALTM
- Se M non si ferma su w, f(⟨M,w⟩) ∉ QPALTM

Poiché A_TM è indecidibile, QPALTM deve essere indecidibile.

2. (12 punti) I *grawlix* sono sequenze di simboli senza senso che sostituiscono le parolacce nei fumetti.

Un linguaggio è *volgare* se contiene almeno un grawlix. Considera il problema di determinare se il linguaggio di una TM è volgare.

(a) Formula questo problema come un linguaggio $GROSS_{TM}$.
(b) Dimostra che il linguaggio $GROSS_{TM}$ è indecidibile.

(a) Formulazione come linguaggio: GROSS_TM = {⟨M⟩ | M è una TM e L(M) contiene almeno un grawlix}

(b) Dimostrazione che GROSS_TM è indecidibile:

Riduzione dal problema della fermata (HALT_TM). Costruiamo una TM M' che, dato input ⟨M,w⟩:

1. Simula M su w.
2. Se M si ferma su w, M' produce l'output "@#$%&!".
3. Se M non si ferma, M' entra in un loop infinito.

Ora, definiamo una funzione f che mappa ⟨M,w⟩ in ⟨M'⟩. f è una riduzione da HALT_TM a GROSS_TM perché:

- Se M si ferma su w, L(M') contiene un grawlix, quindi ⟨M'⟩ ∈ GROSS_TM
- Se M non si ferma su w, L(M') = ∅, quindi ⟨M'⟩ ∉ GROSS_TM

Poiché HALT_TM è indecidibile, GROSS_TM deve essere indecidibile.

2. **(12 punti)** Un linguaggio B è *emozionato* se ogni stringa in B assume la forma ww per qualche $w \in \{0,1\}^*$. Ad esempio, sia $\{00, 1111, 1010\}$ che $\emptyset$ sono linguaggi emozionati, mentre $\{00, 10\}$ non lo è. Considera il problema di determinare se il linguaggio di una TM M è emozionato.

(a) Formula questo problema come un linguaggio EX_{TM}.

(b) Dimostra che il linguaggio EX_{TM} è indecidibile.

(a) EXT_M = { 〈B, w〉 | B è una TM M ed esiste una stringa ww ∈ {0,1}* tale che ww ∈ L(M)}

(b) Dimostrazione che EXT_M è indecidibile per contraddizione riducendo il problema dell'accettazione A_TM. Supponiamo per assurdo che EXT_M sia decidibile e sia H una TM che lo decide. Costruiamo la TM D: D = "Su input 〈M, w〉 :

1. Costruisci la TM M' che accetta se l'input è nella forma xx e x ∈ L(M), rifiuta altrimenti
2. Esegui H su input 〈M', w〉
3. Se H accetta rifiuta, se H rifiuta accetta"

D decide A_TM: Se 〈M,w〉 ∈ A_TM allora w ∈ L(M), quindi ww ∈ L(M') e 〈M',w〉 ∈ EXT_M, quindi H accetta e D rifiuta Se 〈M,w〉 ∉A_TM allora w ∉ L(M), quindi ww ∉L(M') e 〈M',w〉 ∉EXT_M, quindi H rifiuta e D accetta Ma A_TM è indecidibile, quindi abbiamo una contraddizione e EXT_M non può essere decidibile.

2. **(12 punti)** Un linguaggio $B \subseteq \{0,1\}^*$ è *palindromo* se ogni stringa in B è palindroma, cioè se $w = w^R$ per ogni $w \in B$. Ad esempio, sia $\{00, 11011, 1001\}$ che $\emptyset$ sono linguaggi palindromi, mentre $\{00, 10\}$ non lo è. Considera il problema di determinare se il linguaggio di una TM M è palindromo.

(a) Formula questo problema come un linguaggio PAL_{TM}.

(b) Dimostra che il linguaggio PAL_{TM} è indecidibile.

(a) PAL_TM = { 〈M〉 | M è una TM e L(M) ⊆ {0,1}* è palindromo}

(b) PAL_TM è indecidibile. Dimostrazione per riduzione dal problema della cofinalità E_TM = { 〈M〉 | L(M) = ∅}. Costruiamo la funzione di riduzione f: f(〈M〉) = 〈M̂〉 dove M̂ è la TM che su input w ∈ {0, 1}*:

1. Simula M su input ε
2. Se M accetta, accetta se w = w^R, rifiuta se w ≠ w^R
3. Se M non accetta, accetta qualunque input w

Se 〈M〉 ∈ E_TM allora L(M) = ∅, quindi L(M̂) = {0,1}* che è palindromo, perciò 〈M̂〉 ∈ PAL_TM. Se 〈M〉 ∉ E_TM allora L(M) ≠ ∅, quindi L(M̂) = {w ∈ {0,1}* | w = w^R} che non è palindromo, perciò 〈M̂〉 ∉ PAL_TM. Siccome E_TM è indecidibile e si riduce a PAL_TM, anche PAL_TM è indecidibile.