1. (12 punti) Una *Macchina di Turing ad accesso casuale (RATM)* è una variante della macchina di Turing che estende il modello standard introducendo un meccanismo per accedere direttamente a una qualsiasi posizione dell'input, senza dover scorrere sequenzialmente le celle del nastro. Una RATM è dotata di tre nastri:

- un nastro di input a sola lettura, che contiene l'input;
- un nastro di lavoro dove la macchina può leggere, scrivere e spostarsi a piacere;
- un nastro puntatore con alfabeto binario. Anche in questo nastro la macchina può leggere, scrivere e spostarsi a piacere.

Oltre alle consuete operazioni di lettura, scrittura e spostamento delle testine, una RATM può eseguire un'operazione aggiuntiva di accesso diretto all'input. Per eseguire questa operazione, la macchina legge il numero binario p sul nastro puntatore e poi scrive il p-esimo simbolo dell'input sulla cella corrente del nastro lavoro. I simboli dell'input sono numerati da sinistra a destra a partire dalla posizione 0.

(a) Dai una definizione formale della funzione di transizione di una RATM.

(b) Dimostra che le RATM riconoscono la classe dei linguaggi Turing-riconoscibili. Usa una descrizione a livello implementativo per definire le macchine di Turing.

2. (12 punti) Data una parola w su un alfabeto Σ, si dice che $u \in \Sigma^*$ è un suffisso di w se esiste una stringa $v \in \Sigma^*$ tale che $w = vu$. Un linguaggio $L \subseteq \Sigma^*$ è *chiuso per suffisso* se per ogni parola $w \in L$, tutti i suffissi u di w appartengono anch'essi a L. Considera il problema di determinare se il linguaggio di una TM M è chiuso per suffisso.

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

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

3. (12 punti) Un sottoinsieme S di vertici di un grafo non orientato $G = (V, E)$ è *quasi indipendente* se esiste esattamente un arco di G che ha entrambi gli estremi in S. Considera il seguente problema:

$$\text{ALMOSTINDEPENDENTSET} = \{\langle G, k\rangle \mid \text{esiste } S \subseteq V \text{ quasi indipendente di cardinalità } k\}$$

(a) Dimostra che ALMOSTINDEPENDENTSET è un problema NP.

(b) Dimostra che ALMOSTINDEPENDENTSET è NP-hard, usando INDEPENDENTSET come problema NP-hard di riferimento.

Problema 1 (12 punti) - Macchina di Turing ad Accesso Casuale (RATM)

(a) Definizione formale della funzione di transizione

Una RATM è una tupla **M = (Q, Σ, Γ, δ, q_0, q_{accept}, q_{reject})** dove:

- Q è l'insieme finito di stati
- Σ è l'alfabeto di input (non contiene ⊔)
- Γ è l'alfabeto del nastro (contiene ⊔ e Σ)
- q_0 ∈ Q è lo stato iniziale
- q_{accept}, q_{reject} ∈ Q sono gli stati finali

La **funzione di transizione** è definita come:

$\delta : Q \times \Gamma \times \Gamma \rightarrow Q \times \Gamma \times \{L,R\} \times \{L,R\} \times \{DIRECT\}$

dove $\delta(q, a, b) = (q', a', d_1, d_2, op)$ significa:

- **q**: stato corrente
- **a**: simbolo letto dal nastro di lavoro
- **b**: simbolo letto dal nastro puntatore
- **q'**: nuovo stato
- **a'**: simbolo scritto sul nastro di lavoro
- **d_1**: direzione movimento testina nastro di lavoro
- **d_2**: direzione movimento testina nastro puntatore
- **op**: operazione speciale (DIRECT per accesso diretto, ∅ altrimenti)

Semantica operazione DIRECT: La macchina legge il numero binario p dal nastro puntatore, accede alla p-esima posizione dell'input (con indicizzazione da 0), e scrive il simbolo corrispondente sulla cella corrente del nastro di lavoro.

(b) Dimostrazione equivalenza con linguaggi Turing-riconoscibili

Teorema: Le RATM riconoscono esattamente la classe dei linguaggi Turing-riconoscibili.

Dimostrazione: Dobbiamo dimostrare due inclusioni.

Direzione 1: Ogni linguaggio riconosciuto da una RATM è Turing-riconoscibile

Sia **R** una RATM. Costruiamo una TM standard **S** che simula **R**.

S = "Su input w:

1. **Inizializzazione**: Memorizza w in una sezione speciale del nastro preceduta dal marcatore #. Simula i tre nastri di R usando un nastro singolo con separatori.
2. **Simulazione mosse standard**: Per transizioni senza DIRECT, simula normalmente gli spostamenti sui nastri di lavoro e puntatore.
3. **Simulazione accesso diretto**: Quando R esegue DIRECT:
   - Legge il valore binario p dal nastro puntatore simulato
   - Si sposta alla sezione dell'input e accede alla posizione p
   - Copia il simbolo nella posizione corrente del nastro di lavoro
   - Continua la simulazione
4. **Terminazione**: Accetta/rifiuta quando R raggiunge i rispettivi stati finali."

Direzione 2: Ogni linguaggio Turing-riconoscibile è riconosciuto da una RATM

Sia **M** una TM standard. Costruiamo una RATM **R** equivalente.

R = "Su input w:

1. **Copia sequenziale**: Usa il nastro puntatore per mantenere un contatore binario inizializzato a 0.
2. **Simulazione**: Per ogni mossa di M:
   - Se M si sposta a destra: incrementa il contatore binario sul nastro puntatore, usa DIRECT per leggere il simbolo successivo dell'input
   - Se M si sposta a sinistra: decrementa il contatore binario, usa DIRECT per accedere alla posizione precedente
   - Simula scrittura e cambio di stato normalmente
3. **Terminazione**: Termina negli stessi stati di M."

Entrambe le simulazioni sono corrette e preservano il linguaggio riconosciuto. □

Problema 2 (12 punti) - Linguaggi chiusi per suffisso

(a) Formulazione del problema SuffixClosedTM

SuffixClosedTM = {⟨M⟩ | M è una TM e L(M) è chiuso per suffisso}

dove un linguaggio L ⊆ Σ è ***chiuso per suffisso*** *se:* $\forall w \in L,\ \forall u \in \Sigma$: (∃v ∈ Σ* : w = vu) ⟹ u ∈ L

(b) Dimostrazione indecidibilità

Teorema: SuffixClosedTM è indecidibile.

Dimostrazione: Mostriamo ATM $\leq_m$ SuffixClosedTM mediante riduzione.

Sia **F** la seguente macchina che calcola la riduzione:

F = "Su input ⟨M,w⟩:

1. Costruisce la seguente macchina **M'**:

   M' = "Su input x:
   1. Se x = λ (stringa vuota), accetta.
   2. Se x = 0, esegui M su input w:
      - Se M accetta w, accetta

        - Se M rifiuta w, rifiuta
        - Se M va in loop su w, va in loop
    3. Per tutti gli altri input x, rifiuta."
2. Restituisce ⟨M'⟩."

Correttezza della riduzione:

Caso 1: ⟨M,w⟩ ∈ ATM (M accetta w)

- M' accetta {λ, 0}
- I suffissi di λ sono: {λ} ⊆ L(M')
- I suffissi di 0 sono: {λ, 0} ⊆ L(M')
- L(M') è chiuso per suffisso
- Quindi ⟨M'⟩ ∈ SuffixClosedTM

Caso 2: ⟨M,w⟩ ∉ ATM (M non accetta w)

- M' accetta solo {λ}
- Ma 0 ∉ L(M') mentre λ è suffisso di qualsiasi stringa che inizia con 0
- Se esistesse una stringa della forma v0 in L(M'), allora dovremmo avere 0 ∈ L(M')
- L(M') non è chiuso per suffisso
- Quindi ⟨M'⟩ ∉ SuffixClosedTM

La riduzione è calcolabile in tempo polinomiale, dunque SuffixClosedTM è indecidibile. □

Problema 3 (12 punti) - AlmostIndependentSet

(a) Dimostrazione che AlmostIndependentSet ∈ NP

Teorema: AlmostIndependentSet ∈ NP.

Dimostrazione: Costruiamo un verificatore polinomiale **V**.

Certificato: Un sottoinsieme S ⊆ V di cardinalità k.

Verificatore V = "Su input ⟨⟨G,k⟩, S⟩:

1. **Verifica cardinalità**: Controlla |S| = k
2. **Conta archi interni**: Conta il numero di archi (u,v) ∈ E con u,v ∈ S
3. **Verifica quasi-indipendenza**: Accetta sse il conteggio è esattamente 1."

Analisi complessità:

- Passo 1: O(k) ≤ O(|V|)

- Passo 2: $O(|S|^2) \le O(|V|^2)$
- Passo 3: $O(1)$
- **Totale**: $O(|V|^2)$ = polinomiale

Il verificatore è corretto: accetta sse S è quasi indipendente di cardinalità k. □

(b) Dimostrazione NP-hardness

Teorema: AlmostIndependentSet è NP-hard.

Dimostrazione: Riduciamo IndependentSet $\le_p$ AlmostIndependentSet.

Sia **R** la seguente funzione di riduzione:

R = "Su input ⟨G,k⟩ dove G = (V,E):

1. **Costruzione nuovo grafo**: Crea G' = (V', E') dove:
   - $V' = V \cup \{a, b\}$ (aggiungi due nuovi vertici)
   - $E' = E \cup \{(a,b)\}$ (aggiungi arco tra i nuovi vertici)
2. **Nuovo parametro**: $k' = k + 2$
3. Restituisce ⟨G', k'⟩."

Correttezza della riduzione:

Direzione (⇒): Se G ha un insieme indipendente **I** di cardinalità k:

- In G', l'insieme $S = I \cup \{a,b\}$ ha cardinalità k + 2
- Gli archi interni a S sono solo (a,b) (esattamente 1)
- Quindi S è quasi indipendente in G'

Direzione (⇐): Se G' ha un insieme quasi indipendente S di cardinalità k + 2:

- S deve contenere esattamente un arco interno
- **Caso 1**: Se $\{a,b\} \subseteq S$, allora S{a,b} è indipendente in G di cardinalità k
- **Caso 2**: Se $\{a,b\} \not\subset S$, allora S contiene al più uno tra a,b. Ma allora i restanti vertici di S devono formare un insieme quasi-indipendente in G, contraddizione poiché G è un sottografo di G'

Solo il Caso 1 è possibile, quindi G ha un insieme indipendente di cardinalità k.

La riduzione è chiaramente polinomiale, quindi AlmostIndependentSet è NP-hard. □

Conclusione: AlmostIndependentSet è NP-completo. □

Note metodologiche

Queste soluzioni seguono l'approccio formale di Bresolin, caratterizzato da:

- Definizioni matematiche precise
- Dimostrazioni strutturate con casi espliciti
- Analisi di complessità rigorosa
- Tecniche di riduzione standard per indecidibilità e NP-hardness
- Verifiche di correttezza complete per entrambe le direzioni delle equivalenze