Informatica Teorica 2022/2023 - Esercitazione 1

1 Marzo 2023

`melissa.antonelli2@unibo.it`

Schema di Codifica

Quesito 1. Cosa si intende per codifica di un problema decisionale? Quali proprietá deve rispettare?

Soluzione. Per dare risposta a un problema decisionale serve un programma che riceva in input dati *del tipo corretto* e dia output SI/NO. Quindi, poiché le TM calcolano funzioni $\mathbb{N}^k \to \mathbb{N}$, dobbiamo *codificare* il problema come funzione caratteristica di un linguaggio formale. Il linguaggio che lo codifica é

$$L = \{x \in \Sigma^* \mid x = code(\alpha) \textit{ per dato } \alpha \textit{ e } \alpha \textit{ istanza positiva del problema}\}$$

ove code$(\cdot)$ gode delle seguenti proprietá:

- se $\alpha \neq \beta$, allora code$(\alpha) \neq$ code(β)
- possiamo verificare se $x \in \Sigma^*$ é code(α) per qualche α
- possiamo calcolare α a partire da code(α).

Problema 1. Consideriamo la quadratica in x

$$q(x) = ax^2 + bx + c,$$

dove x é una variable e a, b, c sono interi. Assumi il nostro problema decisionale sia costituito dall'insieme delle quadratiche su interi le cui istanze SI siano quelle per cui esiste un intero x tale che $q(x) = 0$. Presenta un *sistema di codifica* tramite cui questo problema possa essere processato da una TM. Presenta almeno un'istanza di tale codifica. (Non é necessario programmare la TM.)
Suggerimento. Nota che ogni quadratica considerata é definita dai tre interi a, b, c. Quindi, é sufficiente un sistema che codifichi di tre interi consecutivi.

Soluzione. Presentiamo un sistema di codifica con alfabeto $\Sigma = \{\texttt{0}, \texttt{1}\}$:

- ogni numero positivo n é codificato come stringa di `1` di lunghezza n.
- ogni numero negativo -n é codificato come `0` seguito da stringa di `1` di lunghezza n.

- il numero `0` é codificato da `000`.
- la tripla di numeri a, b, c é codificata dalla codifica di ciascun a, b e c separate da `0`.

Questo é sufficiente. Es., la quadratica $x^2 - 2x + 3$ é codificata dalla stringa `100110111`.

Macchine di Turing

Quesito 2. Definisci intuitivamente e formalmente la TM standard.
Soluzione. La TM é un modello astratto di computazione. É costituita da un nastro infinito sul lato destro e diviso in celle. Ogni cella puó contenere un simbolo o essere vuota. Una TM ha una testina che si muove sul nastro e si trova sempre in un certo stato. La configurazione é un'istantanea di un passo di computazione della macchina. Nella dinamica della macchina, la testina legge il contenuto della cella e, sulla base della lettura puó: (i) fermarsi o (ii) cambiare stato, leggere un nuovo simbolo nella cella e spostarsi nella cella a sinistra o destra.
Formalmente, una TM é una tupla $\langle \Sigma, Q, q_0, H, \delta \rangle$ dove: Σ é un alfabeto di simboli che include un simbolo speciale per la cella vuota ($\sqcup$); Q é un insieme finito di stati; $q_0 \in Q$ é lo stato iniziale; $H \subseteq Q$ é l'insieme di stati accettanti o finali; δ é la funzione di transizione

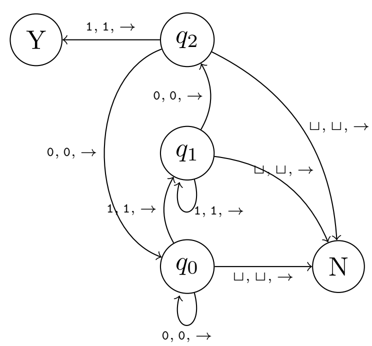
$$\delta : (Q \setminus H) \times \Sigma \longrightarrow Q \times \Sigma \times \{\rightarrow, \leftarrow\}.$$

Problema 2.1. Considera l'alfabeto $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ e rappresenta tramite diagrammi una TM che per ogni stringa presa in input:

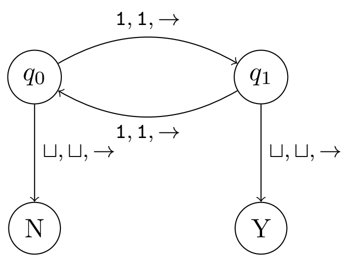
- si fermi e accetti l'input quando esso contiene `101`,
- si fermi e rifiuti l'input altrimenti.

(Per semplicitá assumi l'input sia sempre una stringa finita di caratteri di Σ seguita da un numero infinito di $\sqcup$.)

Soluzione.



Problema 2.2. Descrivi tramite diagrammi la TM che decide il linguaggio delle stringhe dispari su alfabeto $\Sigma = \{\mathtt{1}\}$.
Soluzione.

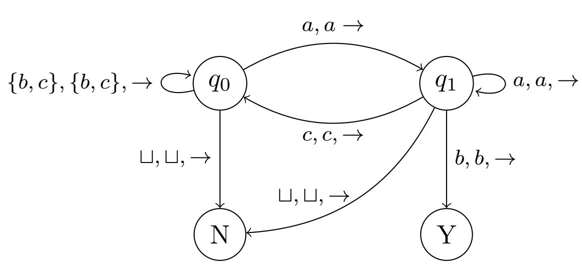


Problema 2.3. Assumi l'alfabeto Σ sia costituito dai simboli a, b, c:

a. Definisci una TM $\mathcal{T}$ che si fermi e accetti l'input quando esso contiene la sequenza ab; si fermi e rigetti l'input altrimenti. (Per semplicitá assumi l'input sia sempre definito da una stringa finita di caratteri di Σ seguita da un numero infinito di $\sqcup$.)

b. Quale *regular expression* descrive il linguaggio deciso da $\mathcal{T}$?

Soluzione. (a.)



(b.) Le stringhe che definiscono il linguaggio deciso da $\mathcal{T}$ sono della forma

$$(a|b|c)^*ab(a|b|c)^*.$$

Linguaggi Decidibili e Riconoscibili

Quesito 3. Cosa si intende per linguaggio decidibile? E per linguaggio riconoscibile?
Soluzione. Diciamo che una TM $\mathcal{M}$ *decide* un linguaggio L quando:

- se $x \in L$, allora $\mathcal{M}$ accetta x
- se $x \notin L$, allora $\mathcal{M}$ rigetta x.

Un linguaggio (problema decisionale) é decidibile se c'é una TM che lo decide.
Diciamo che una TM $\mathcal{M}$ *riconosce* L quando:

- se $x \in L$, allora $\mathcal{M}$ si ferma (raggiugne stato finale)
- se $x \notin L$, allora $\mathcal{M}$ non si ferma.

Un linguaggio (problema decisionale) é riconoscibile se c'é una TM che lo riconosce.

Problema 3. Esiste un linguaggio che sia decidibile ma non riconoscibile?

Soluzione. No. Assumi tale linguaggio L esista. Allora esso sarebbe decidibile, ovvero esisterebbe una TM $\mathcal{M}$ che decide L. Possiamo modificare tale macchina $\mathcal{M}$ sostituendo lo stato di interruzione N con un loop infinito, definendo $\mathcal{M}'$. Poiché per definizione la nuova TM $\mathcal{M}'$ riconosce il linguaggio, ma se L é riconoscibile abbiamo una contraddizione. (Nota che esistono invece linguaggi riconoscibili ma *non* decidibili.)

Halting Problem

Quesito 4. Enuncia la tesi di Church-Turing. Perché in questo contesto la TM é un modello rilevante? Sapresti indicare almeno due modelli equivalenti alla TM standard?

Soluzione. La tesi di Church-Turing afferma che se la soluzione di un problema puó essere calcolata attraverso una procedura algoritmica, allora puó essere calcolata da una TM. Sebbene concretamente questo modello mostra alcuni limiti (per es. sequenzialitá e difficoltá di programmazione), le TM sono concettualmente interessanti perché: forniscono fondazione matematicamente chiara per definire inn modo rigoroso cosa sia un algoritmo e (per tesi di Church-Turing) possono simulare qualsiasi altro modello di calcolo. Dunque, possiamo usarle per dimostrare enunciati matematici su possibilitá e limiti di ció che é calcolabile. Inoltre, la definizione di TM é robusta. Alcuni modelli equivalenti a TM standard sono la macchina a piú nastri, la TM non-deterministica o la URM (*ma* cf. problema 5).

Problema 4. Dimostra, per assurdo, che $HALT^-$ non é riconoscibile.

Suggerimento. Ricorda che il complemento di $HALT$ é definito:

$$\begin{aligned} HALT^- = \{\langle y, x\rangle \in \Sigma^* \times \Sigma^* \mid\ & y \neq \text{code}(\mathcal{M}) \text{ per qualsiasi } \mathcal{M} \text{ o} \\ & y = \text{code}(\mathcal{M}) \text{ \& } \mathcal{M} \text{ non si ferma su } x\}. \end{aligned}$$

La dimostrazione é per assurdo. Assumiamo $HALT^-$ sia riconoscibile, quindi, per definizione, esista una TM $\mathcal{M}_{H^-}$ che lo riconosce. Definiamo una nuova TM $\mathcal{M}''$ tale che per ogni $z \in \Sigma^*$, simula $\mathcal{M}_{H^-}$ su input $\langle z, z\rangle$ e si ferma se la macchina si ferma; altrimenti entra in loop. Eseguiamo $\mathcal{M}''$ su input $code(\mathcal{M}'')$.

Soluzione. Se $\mathcal{M}_{H^-}$ si ferma su $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle$. Allora, per definizione di $\mathcal{M}''$, $\mathcal{M}''$ si ferma su $\text{code}(\mathcal{M}'')$. Per definizione di $HALT^-$, $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle \notin HALT^-$. Allora, per definizione di $\mathcal{M}_{H^-}$, $\mathcal{M}_{H^-}$ non si ferma su $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle$. Questo da contraddizione.

Se $\mathcal{M}_{H^-}$ non si ferma su $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle$, allora per definizione di $\mathcal{M}''$, $\mathcal{M}''$ non si ferma su $\text{code}(\mathcal{M}'')$. Quindi, per definizione di $HALT^-$, $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle \in HALT^-$. Per definizione di $\mathcal{M}_{H^-}$ concludiamo che $\mathcal{M}_{H^-}$ si ferma su $\langle \text{code}(\mathcal{M}''), \text{code}(\mathcal{M}'')\rangle$. Quindi, anche in questo caso si ha contraddizione.

L'unica assunzione necessaria per costruire $\mathcal{M}''$ é che una TM $\mathcal{M}_{H^-}$ che riconosce $HALT^-$ esista. Quindi, visto che $\mathcal{M}_{H^-}$ non esiste, $HALT^-$ non é riconoscibile.

Esercizi Difficili

Problema 5. Definiamo una classe $\mathcal{C}$ di modelli di computazione simili alla TM standard ma privi del comando $\leftarrow$, quindi tali per cui la testina non possa muovere a sinistra ma solo a destra. Formalmente, la funzione di transizione é quindi data da $\delta : (Q \setminus H) \times \Sigma \to Q \times \Sigma \times \{\to\}$ e non da $\delta : (Q \setminus H) \times \Sigma \to Q \times \Sigma \times \{\leftarrow, \to\}$. Dunque,

a. Fornisci un argomento informale del fatto che il modello definito da $\mathcal{C}$ non sia equivalente a quello definito dalla classe di TM standard.

b. Fornisci una prova formale.

c. Farebbe differenza il fatto che il modello di computazione di $\mathcal{C}$ fosse definito sostituendolo il comando $\leftarrow$ con il comando $-$ che mantiene la testina dove si trova? Motiva informalmente.

Suggerimento. Considera L con alfabeto $\Sigma = \{\mathtt{0}, \mathtt{1}\}$ definito su stringhe con uguale numero di `0` e `1`. Esiste una TM (standard) $\mathcal{M}$ che decide L. Consideriamo $\mathcal{M}' \in \mathcal{M}$, ovvero sia $\mathcal{M}$ una TM senza $\leftarrow$. Assumiamo $\mathcal{M}'$ decida L e abbia n stati.

Problema 6. Un *enumeratore* é definito da una TM annessa a una stampante e che rispetta i seguenti vincoli: (1) non c'é lo stato di fermata, (2) esiste uno stato q_{print}, (3) il nastro inizia con $\sqcup$, (4) ogni volta che la macchina entra nello stato q_{print} il contenuto corrente dell nastro viene stampato (il contenuto del nastro immediatamente successivo a $\triangleright$ fino al primo simbolo $\sqcup$). Quindi un enumeratore E inizia con $\sqcup$, su cui scrive alcune stringhe dall'alfabeto, talvolta stampandone e puó stampare la stessa stringa piú volte. Il linguaggio enumerato da E é la raccolta di tutte le stringhe stampate da E. Dunque,

a. Descrivi un enumeratore che generi il linguaggio

$$L_{even} := \{x \in \Sigma^* \ : \ |x| \ pari\}$$

sull'alfabeto $\Sigma = \{1\}$.

b. Dimostra che un linguaggio L é riconoscibile quando esiste un enumeratore E che lo enumera.

Suggerimento (b.) $\Rightarrow$ Se L é riconoscibile, allora esiste un enumeratore E che enumera L. Descrivi come costruire E. E puó usare TM come componenti...
$\Leftarrow$ Se esiste un enumeratore E per il linguaggio L, allora L é riconoscibile. Assumiamo di avere un enumeratore E per L...