Automi e Linguaggi Formali

Parte 11 – La Tesi di Church-Turing

Davide Bresolin
Ultimo aggiornamento: 21 aprile 2021

Università degli Studi di Padova

Sommario

Un nuovo modello di calcolo

- Automi finiti: dispositivi con ridotta quantità di memoria
- Automi a pila: memoria illimitata con accesso LIFO
- Ci sono linguaggi che vanno oltre le capacità di FA e PDA
- FA e PDA sono limitati come modelli di computer

Macchine di Turing

https://youtu.be/FTSAiF9AHN4

- Modello di calcolo proposto da Alan Turing nel 1936
- Memoria illimitata e senza restrizioni
- Può fare tutto ciò che può fare un computer reale

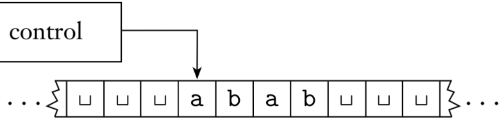
https://youtu.be/FTSAiF9AHN4

- Modello di calcolo proposto da Alan Turing nel 1936
- Memoria illimitata e senza restrizioni
- Può fare tutto ciò che può fare un computer reale

Tuttavia...

- ci sono problemi che una Macchina di Turing non può risolvere
- questi problemi vanno oltre le capacità di un computer

Macchine di Turing



- un nastro infinito come memoria illimitata
- una testina che legge e scrive simboli sul nastro
- all'inizio il nastro contiene l'input
- per memorizzare informazione si scrive sul nastro
- la testina si può muovere ovunque sul nastro
- stati speciali per accetta e rifiuta

Primo esempio di TM

- Costruiamo una macchina di Turing per il linguaggio

$$B = \{w\#w \mid w \in \{0,1\}^*\}$$

- M_2 deve accettare se l'input sta in B, e rifiutare altrimenti

Primo esempio di TM

- Costruiamo una macchina di Turing per il linguaggio

$$B = \{w\#w \mid w \in \{0,1\}^*\}$$

- M_2 deve accettare se l'input sta in B, e rifiutare altrimenti
- $M_2 =$ "Su input w:
  1. Si muove a zig-zag lungo il nastro, raggiungendo posizioni corrispondenti ai due lati di $\#$ per controllare se contengono lo stesso simbolo. In caso negativo, o se non trovi $\#$, rifiuta. Barra gli elementi già controllati.
  2. Se tutti i simboli a sinistra di $\#$ sono stati controllati, verifica i simboli a destra di $\#$. Se c'è qualche simbolo ancora da controllare rifiuta, altrimenti accetta."

`turingmachine.io/?import-gist=87ac7c04c0dc18a3061188142a2ae5e4`

Automi finiti vs Macchine di Turing

1. Una TM può sia scrivere che leggere sul nastro
2. Una TM può muoversi sia a destra che a sinistra
3. Il nastro è infinito
4. Gli stati di rifiuto e accettazione hanno effetto immediato

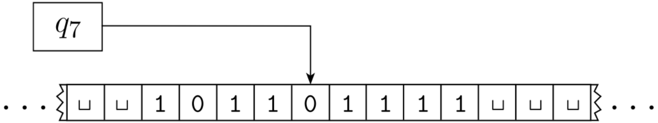
TM: Definizione formale

UNIVERSITÀ DEGLI STUDI DI PADOVA

Una Macchina di Turing (o Turing Machine, TM) è una tupla $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$:

- Q è l'insieme finito di stati
- Σ è l'alfabeto di input che non contiene il simbolo blank $\sqcup$
- Γ è l'alfabeto del nastro che contiene $\sqcup$ e Σ
- $\delta : Q \times \Gamma \mapsto Q \times \Gamma \times \{L, R\}$ è la funzione di transizione
- $q_0 \in Q$ è lo stato iniziale
- $q_{accept} \in Q$ è lo stato di accettazione
- $q_{reject} \in Q$ è lo stato di rifiuto (diverso da q_{accept})

TM: configurazioni



- Lo stato corrente, la posizione della testina ed il contenuto del nastro formano la configurazione di una TM
- Dalla configurazione possiamo sapere la prossima mossa
- Le configurazioni sono rappresentate da una tripla uqv:
  - q è lo stato corrente
  - u è il contenuto del nastro prima della testina
  - v è il contenuto del nastro dalla testina in poi
  - la testina si trova sul primo simbolo di v
- la configurazione in figura è $1011q_701111$

TM: computazione

Università degli Studi di Padova

- La configurazione C_1 **produce** C_2 se la TM può passare da C_1 a C_2 in un passo
- Se $a, b, c \in \Gamma$, $u, v \in \Gamma^*$ e q_i, q_j sono stati, allora:

$$uaq_ibv \text{ produce } uq_jacv \qquad \text{se } \delta(q_i, b) = (q_j, c, L)$$
$$uaq_ibv \text{ produce } uacq_jv \qquad \text{se } \delta(q_i, b) = (q_j, c, R)$$

TM: computazione

- La configurazione iniziale con input w è $q_0 w$
- In una configurazione di accettazione lo stato è q_{accept}
- In una configurazione di rifiuto lo stato è q_{reject}
- Le configurazioni di accettazione e rifiuto sono configurazioni di arresto

Linguaggi Turing-riconoscibili

- una TM M accetta l'input w se esiste una sequenza di configurazioni $C_1, C_2, \ldots, C_k$ tale che:
  - C_1 è la configurazione iniziale con input w
  - ogni C_i produce C_{i+1}
  - C_k è una configurazione di accettazione
- Linguaggio riconosciuto da M: insieme delle stringhe accettate da M

Definition

Un linguaggio è Turing-riconoscibile (o anche ricorsivamente enumerabile) se esiste una macchina di Turing che lo riconosce.

Linguaggi Turing-decidibili

- Se forniamo un input ad una TM, ci sono tre risultati possibili:
  - la macchina accetta
  - la macchina rifiuta
  - la macchina va in loop e non si ferma mai
- la TM può non accettare sia rifiutando che andando in loop
- una TM che termina sempre la computazione è un decisore
- Un decisore decide un linguaggio se lo riconosce

Definition

Un linguaggio è Turing-decidibile (o anche ricorsivo) se esiste una macchina di Turing che lo decide.

Sommario

Esempio 1

TM che decide il linguaggio di tutte le stringhe di 0 la cui lunghezza è una potenza di 2:

$$A = \{0^{2^n} \mid n \geq 0\}$$

Esempio 1

TM che decide il linguaggio di tutte le stringhe di 0 la cui lunghezza è una potenza di 2:

$$A = \{0^{2^n} \mid n \geq 0\}$$

M_1 = "su input w:

1. Scorri il nastro da sinistra a destra, cancellando ogni secondo 0
2. Se il nastro conteneva un solo 0, accetta
3. Se il nastro conteneva un numero dispari di 0, rifiuta
4. Ritorna all'inizio del nastro
5. Vai al passo 1."

Esempio 2

TM che decide il linguaggio:

$$B = \{w\#w \mid w \in \{0,1\}^*\}$$

Esempio 2

TM che decide il linguaggio:

$$B = \{w\#w \mid w \in \{0,1\}^*\}$$

$M_2 =$ "Su input w:

1. Si muove a zig-zag lungo il nastro, raggiungendo posizioni corrispondenti ai due lati di $\#$ per controllare se contengono lo stesso simbolo. In caso negativo, o se non trovi $\#$, rifiuta. Barra gli elementi già controllati.
2. Se tutti i simboli a sinistra di $\#$ sono stati controllati, verifica i simboli a destra di $\#$. Se c'è qualche simbolo ancora da controllare rifiuta, altrimenti accetta."

Esempio 3

TM che esegue operazioni aritmetiche. Decide il linguaggio:

$$C = \{a^i b^j c^k \mid k = i \cdot j \text{ e } i, j, k \geq 1\}$$

Esempio 3

$M_3 =$ "su input w:

1. Scorri il nastro da sinistra a destra e controlla se l'input sta in $a^+b^+c^+$. Rifiuta se non lo è.
2. Ritorna all'inizio del nastro
3. barra una a e scorri a destra fino a trovare una b. Fai la spola tra b e c, barrando le b e le c fino alla fine delle b. Se tutte le c sono barrate e rimangono ancora b, rifiuta
4. Ripristina le b barrate e ripeti 3 finché ci sono a da barrare.
5. Quanto tutte le a sono barrate, controlla se tutte le c sono barrate: se si accetta, altrimenti rifiuta."

Esempio 4

TM che risolve il problema degli elementi distinti. Prende in input una sequenza di stringhe separate da $\#$ e accetta se tutte le stringhe sono diverse. Decide il linguaggio:

$$D = \{\#x_1\#x_2\# \cdots \#x_\ell \mid x_i \in \{0,1\}^* \text{ e } x_i \neq x_j \text{ per ogni } i \neq j\}$$

Conclusioni

- I linguaggi A, B, C e D sono decidibili
- Tutti i linguaggi Turing-decidibili sono anche Turing-riconoscibili
- I linguaggi A, B, C e D sono anche Turing-riconoscibili

Conclusioni

- I linguaggi A, B, C e D sono decidibili
- Tutti i linguaggi Turing-decidibili sono anche Turing-riconoscibili
- I linguaggi A, B, C e D sono anche Turing-riconoscibili
- Vedremo che ci sono linguaggi Turing-riconoscibili ma non decidibili