Complessità

Marco Sgobino

24 marzo 2022

INDICE

Parte I

Complessità

CAPITOLO 1

LA MACCHINA DI TURING

1.1 Descrizione

Una *macchina di Turing* è una macchina che è descritta da un insieme di *simboli* $\Gamma = \{\alpha, \beta, \gamma, \delta, \dots\}$ finito e da un insieme di *stati* $Q = \{q_1, q_2, \dots, q_n\}$ anch'esso finito. La macchina di Turing dispone di un nastro di memoria potenzialmente illimitato a destra e a sinistra; essa è in grado di identificare il simbolo nella posizione dove la *testina* della macchina è collocata sul nastro. Ad ogni iterazione della macchina di Turing, viene letto il simbolo ove sia collocata la sua testina; a seconda dello stato q_i, viene intrapresa un'azione, che può essere una fra le 3 seguenti:

- spostamento della testina a destra;
- spostamento della testina a sinistra;
- riscrittura del simbolo sotto la testina con uno qualsiasi appartenente all'alfabeto di simboli.

Una macchina di Turing può anche essere accompagnato da un alfabeto *ausiliario*, ovverosia un alfabeto $\mathcal{V}$ comprendente simboli simili a quelli di Γ, ma che vengono utilizzati qualora la macchina di Turing avesse *già processato* la cella in questione. Tipicamente, l'utilizzo dell'alfabeto ausiliario è importante nel caso specifico in cui si adoperino algoritmi e procedure per le quali è d'interesse ricordare se una cella sia già stata in precedenza processata dalla macchina di Turing, oppure no.

Nello specifico, una macchina di Turing è univocamente identificata dalla sua *matrice di transizione* $\delta : Q \times \Gamma \to Q \times (\Gamma \cup \{L, R\})$[1], dove i simboli L ed R sono rappresentativi dell'azione di spostarsi, rispettivamente, a sinistra e a destra del nastro di memoria. La matrice di transizione lega, dunque, ciascuno stato al simbolo collocato immediatamente sotto alla testina nel nastro di memoria, stabilendo in maniera univoca l'azione da intraprendere. L'insieme delle azioni che la macchina di Turing compie è quindi la realizzazione del programma stesso.

	q_1	q_2	$\cdots$	q_n
α	β/q_2	γ/q_2	$\cdots$	
β	L/q_1	γ/q_3	$\cdots$	
γ	γ/q_3	R/q_2	$\cdots$	
$\vdots$	$\vdots$	$\vdots$	$\ddots$	

Tabella 1.1: Possibile matrice di transizione per una macchina di Turing. Le righe corrispondono ai simboli dell'alfabeto Γ, mentre le colonne sono corrispondenti ai singoli stati dell'insieme Q.

Diversamente dal modello RAM, la quantità di memoria destinata ad ogni cella è *limitata*, poiché vi può essere collocato soltanto un numero finito di simboli, quelli appunto dell'insieme Γ. Ciononostante, la macchina di Turing si presta meglio alla trattazione di stringhe, poiché i simboli possono rappresentare qualsivoglia tipologia di entità astratta, mentre per il modello RAM si avrebbe necessità di una codifica fra numeri reali e simboli da trattare: non vi è più dunque la limitazione imposta dal fatto che all'interno di una cella possa risiedere esclusivamente un numero naturale, non importa quanto grande sia; nella macchina di Turing le celle possono contenere simboli di qualsiasi natura essi siano.

Tipicamente, all'alfabeto che definisce una macchina di Turing viene definito un sottoinsieme sigma di simboli di input, $\Sigma \subset \Gamma$, in concomitanza al quale viene definito un simbolo vuoto, ***blank***, $b \in \Gamma - \Sigma$. Il simbolo b rappresen-

[1] In talune circostanze, è possibile trovare una definizione differente, cioè

$$\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\};$$

in altre parole, è una macchina che *si muove sempre sul nastro,* poiché per ogni stato è sempre definito un movimento. Per questo tipo di macchina, sono necessarie diverse regole d'ingaggio, e i programmi in essa costruiti saranno radicalmente differenti. La differenza fra i due modelli rappresenta un'evidenza della versatilità della macchina di Turing.

ta dunque l'idea di *cella vuota*. Ricapitolando, ogni macchina di Turing viene univocamente definita da:

- un insieme finito di simboli Γ – essi comprendono sia i simboli di input Σ che il simbolo *blank* b;
- un insieme finito di stati Q;
- una funzione (matrice) di transizione $\delta : Q \times \Gamma \to Q \times (\Gamma \cup \{L, R\})$.

Una diversa maniera per definire una macchina di Turing è tramite la *quaterna* o *quadrupla* q_i, s_j, α, q_k, dove q_i è lo *stato in cui si trova la macchina*, s_j è il *simbolo letto dalla testina*, α è il *simbolo scritto nella matrice di transizione*, q_k è lo *stato successivo* in cui la macchina di Turing si troverà al termine dell'esecuzione di α. In particolare, l'operazione che la macchina di Turing effettua dipende dal simbolo α scritto nella matrice di transizione:

- se $\alpha = s_i$, sostituisci il simbolo s_j con s_i;
- se $\alpha = R$, muovi la testina a destra;
- se $\alpha = L$, muovi la testina a sinistra;

Lo *stop* della computazione di una macchina di Turing avviene qualora la coppia q_i, s_j **non** sia presente nella matrice di transizione. In tal caso, la macchina di Turing si fermerà, e vi sarà il riconoscimento del valore finale di computazione, espresso come nel caso del modello RAM con una convenzione che permetta di identificare il valore finale del risultato della computazione.

1.2 Uso di una macchina di Turing

Una macchina di Turing, come verrà illustrato in seguito, può essere espressa anche mediante *grafo* oltre che mediante matrice di sostituzione. In ogni caso, tutte le possibili definizioni di macchina di Turing sono del tutto equivalenti, e variano esclusivamente nella forma nella quale esse sono proposte. Una macchina di Turing, non importa come sia stata definita, può essere adoperata sostanzialmente per 3 diversi compiti:

1. per il *calcolo* di una funzione – la macchina di Turing è intesa come *calcolatore*, e lo scopo è quello di calcolare una funzione $f : \mathbb{N}^n \to \mathbb{N}$. In questo caso, la macchina di Turing risulta essere meno efficiente del modello RAM, per via dell'assenza del comodo sistema di istruzioni presente in quest ultimo;

2. per il *riconoscimento* di una stringa – la macchina di Turing è intesa come *accettore*. In questo contesto la MdT è molto più efficiente del modello RAM, poiché non è richiesta la codifica da numeri naturali a simboli;
3. per la *decisione* di un predicato – la macchina di Turing è intesa come *decisore*.

TODO aggiungi esito macchina di turing TODO aggiungi esito vettoriale mdt

Solitamente, è particolarmente difficile programmare sulla macchina di Turing -- questo è principalmente dovuto al fatto che la macchina di Turing è una macchina a stati, dove non vi è un insieme di istruzioni da applicare direttamente, ma bisogna determinare la matrice di transizione relativa a ciò che bisogna calcolare, stato per stato e simbolo per simbolo.

1.3 Equivalenza fra macchina di Turing e $\mathcal{R}$

Un importante teorema definisce l'equivalenza della macchina di Turing (avente insieme delle funzioni computabili $\mathcal{TC}$ all'insieme delle funzioni parziali ricorsive $\mathcal{R}$, e lo lega indissolubilmente all'insieme delle funzioni computabili dal modello RAM $\mathcal{C}$.

Teorema 1 *dell'equivalenza della macchina di Turing all'insieme $\mathcal{R}$*

$$\mathcal{R} \equiv \mathcal{TC} \equiv \mathcal{C}$$

Un possibile spunto di dimostrazione di $\mathcal{TC} \subseteq \mathcal{R}$ si ha per il fatto che la configurazione e lo stato della MdT durante la computazione possono essere codificati da un numero naturale; le operazioni sulla macchina sono rappresentate da funzioni ricorsive su questi numeri. Il viceversa è invece mostrabile tenendo conto che si può verificare che $\mathcal{TC}$ contiene le funzioni di base ed è chiusa rispetto a sostituzione, ricorsione e minimazione illimitata.

1.4 Potenziamento apparente della macchina di Turing

Una macchina di Turing può essere potenziata mediante l'estensione di essa tramite l'uso di *nastri multitraccia*. In altre parole, anziché adoperare un singolo nastro, si adoperano più nastri contemporaneamente. La macchina di Turing viene espansa tramite l'aggiunta di uno *stato della memoria suppletiva*, che sostanzialmente ci indica il numero del nastro dove la macchina di Turing sta agendo. Dunque, ci possiamo immaginare una macchina di Turing con tanti nastri e tante testine che lavorano contemporaneamente, come elencato in Figura 1.1.

Dal punto di vista della capacità computazione, una macchina di Turing multinastro non aumenta né diminuisce le capacità - tuttavia, tale rappresentazione può prestare una maggiore somiglianza con il tipo di computazione svolto all'interno di un computer moderno. Si pensi infatti alla memoria RAM, alla memoria cache, al disco rigido e così via; una macchina di Turing può dunque "simulare" qualsiasi computer moderno[2]. Una macchina di Turing multinastro deve essere implementata con un *sistema multitraccia* - in altre parole, vengono adoperate *tante testine quante sono i nastri.* Ci si può facilmente ricondurre alla macchina di Turing convenzionale semplicemente eliminando il sistema multitraccia, o per meglio dire, una macchina di Turing multitraccia può essere *simulata* da una macchina di Turing convenzionale: essa dunque, non produce alcun tipo di miglioramento dal punto di vista computazione, cioè le due macchine hanno la *stessa* potenza computazionale (Figura 1.2. Il medesimo discorso si applica anche tentando di *menomare* la macchina di Turing, nel senso che potremmo pensare di rendere il nastro semi-infinito, cioè illimitato solo a destra o solo a sinistra (Figura 1.3).

[2]Un computer può a sua volta simulare da una macchina di Turing, dal momento che possono essere applicati potenzialmente infiniti banchi di memoria al computer - nella pratica, tuttavia, sappiamo che ciò non è possibile, e i banchi di memoria non saranno mai del tutto *illmitati*

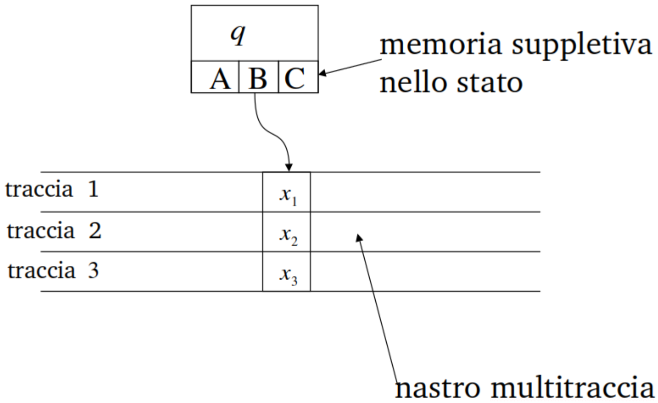


Figura 1.1: Macchina con memoria con nastro multitraccia.

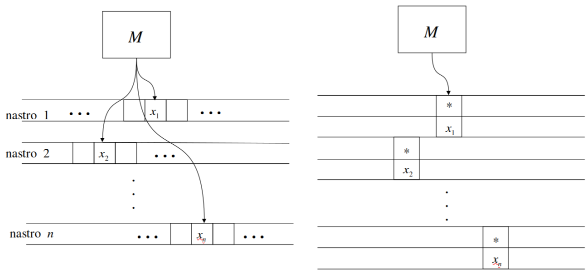


Figura 1.2: Equivalenza fra una macchina di Turing a multitraccia e una macchina di Turing a singola testina.

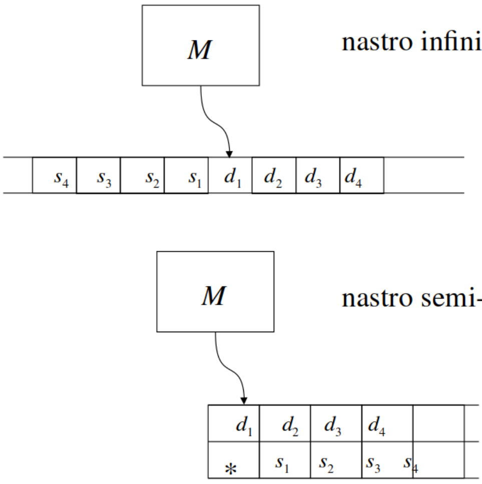


Figura 1.3: Equivalenza fra macchina di Turing a nastro illimitato solo a destra e a nastro illimitato da ambedue le parti.

1.5 Macchina RAM per *accettare* stringhe

Nel modello RAM, per accettare una stringa è necessario operare una codifica. In particolare, la stringa viene codificata in un numero naturale e la macchina risponde con "1" o "0" a seconda che la stringa venga o meno accettata. Con la macchina di Turing, invece, si può far intervenire direttamente uno *stato di accettazione* q_Y o uno *stato di rifiuto* q_N. Un'altra possibilità è quella di semi-decidere riguardo l'accettazione di una stringa, cioè una macchina in grado di riconoscere la stringa in questione, ma di non poter rifiutare la stringa, poiché ci sarebbe un'infinita computazione, e dunque divergenza. Tipicamente, lo stato iniziale viene denotato con q_0, e potrebbero esserci degli stati ulteriori "intermedi". Un esempio di accettazione è dato dalla macchina illustrata in Figura 1.4. La macchina riconosce il linguaggio dato dalle stringhe con due "zeri" nelle ultime due posizioni a destra, in particolare tutte le stringhe che terminano con "00".

TODO aggiungi mdt definita a grafo

Una macchina di Turing può anche essere "definita a grafo". In questo modello di definizione, la macchina di Turing,

- ha un *nastro semi-illimitato* a destra, diviso in celle;
- ha un alfabeto *ausiliario* $\mathcal{V}$;
- ha un simbolo di spaziatura Δ, equivalente al simbolo *blank* b;
- un puntatore, del tutto equivalente alla testina;
- un *programma*, definito come **grafo finito orientato**, con i vertici definiti come *stato*. Vi è uno stato di inizio, indicato con INIZIO, e un sottoinsieme eventualmente vuoto di stati di arresto, indicati con ACCETTAZIONE. I nodi del grafo sono collegati da *archi*.

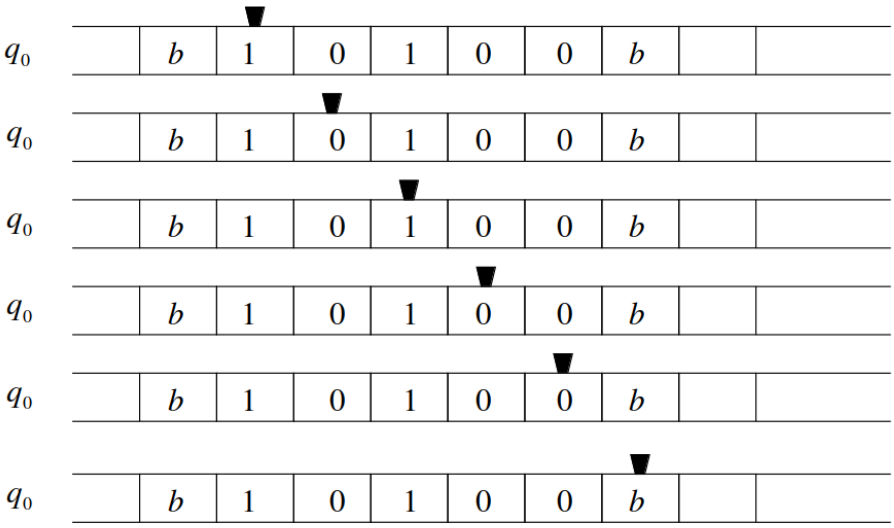
Ciascun arco è della forma

```
(i) ---(alpha, beta, gamma) ---> (j)
```

dove

$$\alpha \in \Sigma \cup \mathcal{V} \cup \{\Delta\}, \beta \in \Sigma \cup \mathcal{V} \cup \{\Delta\} \text{ e } \gamma \in \{L, R\}.$$

Dunque, siamo nello stato i; la macchina legge α, scrive β al posto di α, e infine va a destra oppure a sinistra a seconda che il simbolo γ sia pari ad R o ad L.



	0	1	b
q_0	(q_0,0,R)	(q_0,1,R)	(q_1,0,L)
q_1	(q_2,0,L)	(q_3,0,L)	(q_N,0,L)
q_2	(q_Y,0,L)	(q_N,0,L)	(q_N,0,L)
q_3	(q_N,0,L)	(q_N,0,L)	(q_N,0,L)

Figura 1.4: Macchina che riconosce il linguaggio dato dalle stringhe con due 0 nelle ultime due posizioni a destra, e suo funzionamento.