

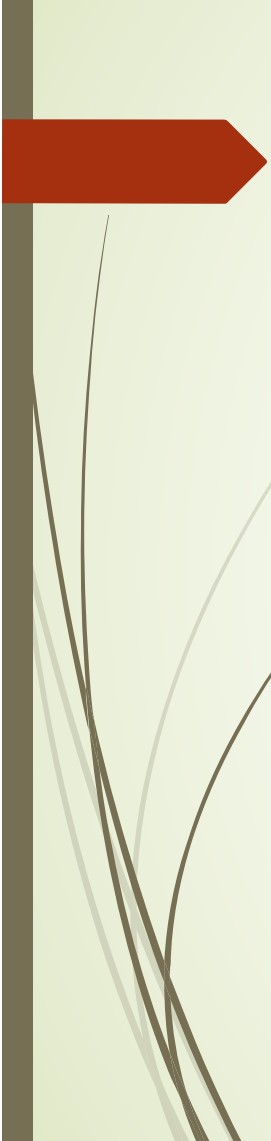


Lezione 2 – macchine di Turing

Lezione del 06/03/2025

Definizione di Macchina di Turing

- La macchina di Turing che abbiamo visto informalmente durante la scorsa lezione utilizza 3 nastri: sui primi due nastri, prima che la macchina inizi ad operare, vengono scritti (dall'utente) i due numeri da sommare, sul terzo, inizialmente vuoto, la macchina scrive il risultato nel corso della sua computazione
- Dobbiamo, ora, formalizzare questi concetti e, allo scopo, cominciamo con il limitarci a considerare macchine di Turing che utilizzano un solo nastro
- La definizione 1.3 a pag. 9 della dispensa 1 presenta una macchina di Turing ad un nastro come:
 - una unità di controllo che, ad ogni istante, può trovarsi in uno stato interno appartenente ad un certo insieme Q che contiene, fra gli altri, lo stato particolare q_0 che fa partire la computazione e un sottoinsieme Q_f di stati che fanno terminare la computazione
 - un nastro suddiviso in un infinito numero di celle, ciascuna delle quali, ad ogni istante, può essere vuota o contenere un simbolo appartenente ad un alfabeto Σ , e sul quale nastro si muove una testina di lettura/scrittura
 - ad ogni istante, dipendentemente dallo stato interno e da ciò che è letto dalla testina, viene eseguita una quintupla scelta in un insieme P di quintuple



Definizione di Macchina di Turing

- E come funziona una macchina di Turing?
- All'inizio, quando l'unità di controllo si trova nello stato q_0 : la testina legge il simbolo contenuto nella cella che sta scandendo e la macchina cerca una quintupla i cui primi due elementi sono q_0 e il simbolo letto dalla testina (che può anche essere il simbolo **blank** ) e, se trova una tale quintupla, la esegue
 - se non la trova ... non compie alcuna azione (ci torneremo più avanti) e la computazione termina
- *Eseguire una quintupla* significa eseguire le tre azioni in essa indicate:
 - sovrascrivere il simbolo nella cella scandita dalla testina con il simbolo indicato nella quintupla
 - cambiare (eventualmente) stato interno
 - eventualmente, ossia, a meno che nella quintupla non sia indicato di rimanere nel medesimo stato in cui ci si trovava prima della sua esecuzione
 - muovere (eventualmente) la testina
 - eventualmente, ossia a meno che nella quintupla sia indicato "ferma"
- Eseguita la prima quintupla, si cerca un'altra quintupla da eseguire (ossia, una quintupla i cui primi due elementi sono lo stato in cui si trova la macchina e il simbolo letto dalla testina) e così via, fino a quando nessuna quintupla può essere eseguita

Esempio di macchina di Turing

- Consideriamo una macchina di Turing ad un nastro, $T_{\text{parità}}$, definita sull'alfabeto $\Sigma = \{0, 1, p, d\}$ e sull'insieme di stati $Q = \{q_0, q_p, q_d, q_F\}$

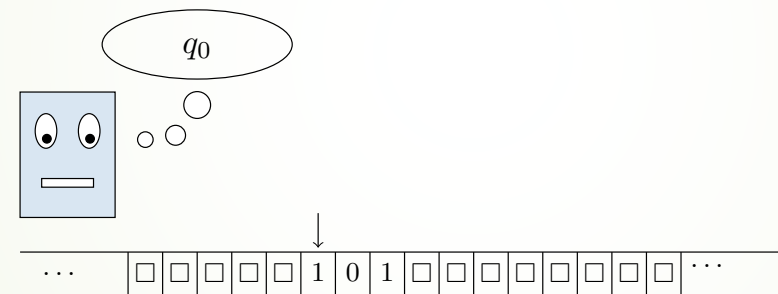
con stato iniziale q_0 e stato finale q_F il cui insieme delle quintuple è

$$P = \{ \langle q_0, 0, \blacksquare, q_p, \text{destra} \rangle, \langle q_0, 1, \blacksquare, q_d, \text{destra} \rangle, \\ \langle q_p, 0, \blacksquare, q_p, \text{destra} \rangle, \langle q_d, 0, \blacksquare, q_d, \text{destra} \rangle, \\ \langle q_p, 1, \blacksquare, q_d, \text{destra} \rangle, \langle q_d, 1, \blacksquare, q_p, \text{destra} \rangle, \\ \langle q_p, \blacksquare, p, q_F, \text{fermo} \rangle, \langle q_d, \blacksquare, d, q_F, \text{fermo} \rangle \}$$

- La macchina $T_{\text{parità}}$ scandisce la sequenza di caratteri scritta sul suo nastro, cancellandoli via via che vengono scanditi, e verificando se tale sequenza contiene un numero pari o un numero dispari di '1': al termine della scansione, nel primo caso scrive 'p' e termina, nel secondo caso scrive 'd' e termina
 - verificarlo per esercizio

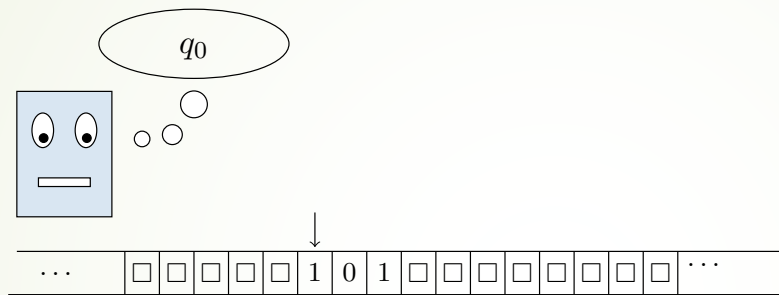
Esempio di macchina di Turing

- Vediamo ora la macchina $T_{\text{parità}}$ in azione:
 - poniamo la macchina nello stato q_0
 - scriviamo una sequenza di caratteri sul nastro – *che era precedentemente vuoto*
 - posizioniamo la testina sul carattere più a sinistra fra quelli scritti sul nastro

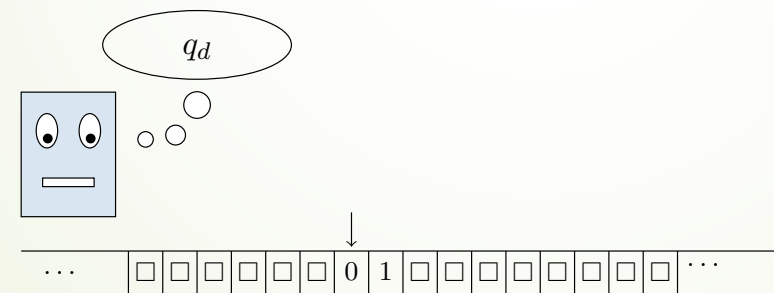


- .. e vediamo cosa succede
- Osserviamo che P contiene la quintupla $\langle q_0, 1, \square, q_d, \text{destra} \rangle$ e che essa può essere eseguita

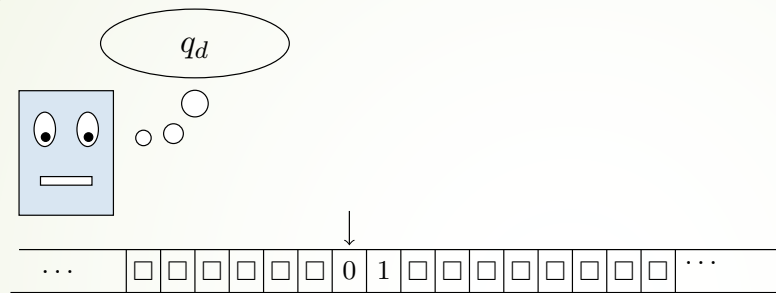
Esempio di macchina di Turing



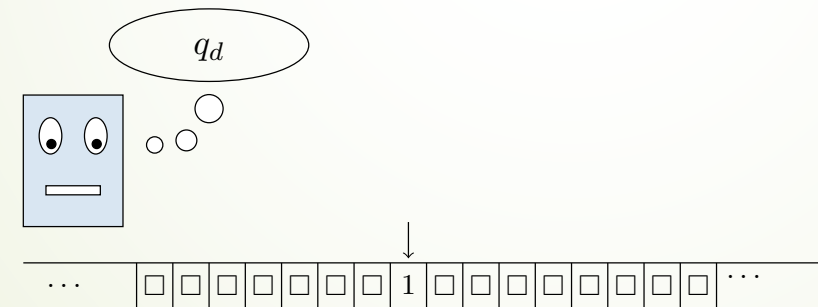
➡ eseguiamo, dunque, la quintupla $\langle q_0, 1, \blacksquare, q_d, \text{destra} \rangle$:



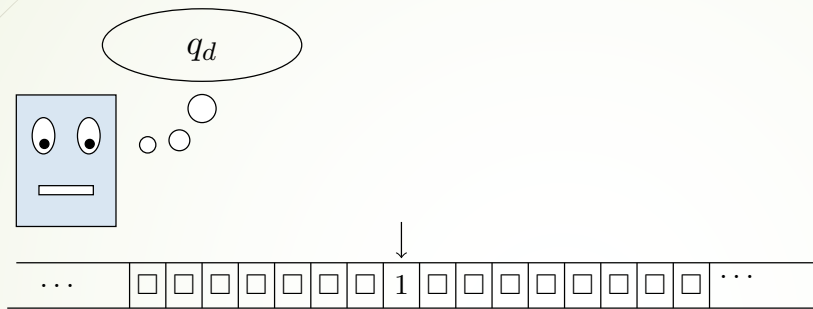
Esempio di macchina di Turing



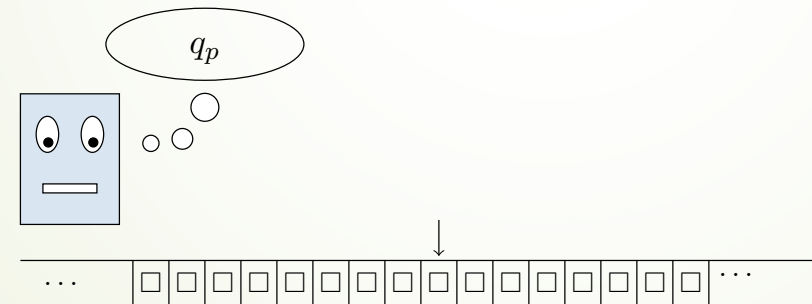
➡ ora possiamo eseguire la quintupla $\langle q_d, 0, \blacksquare, q_d, \text{destra} \rangle \in P$:



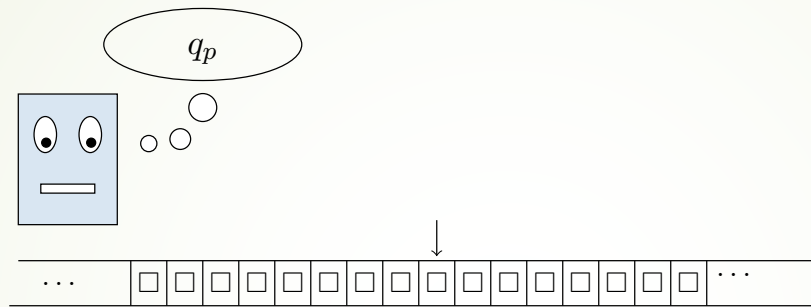
Esempio di macchina di Turing



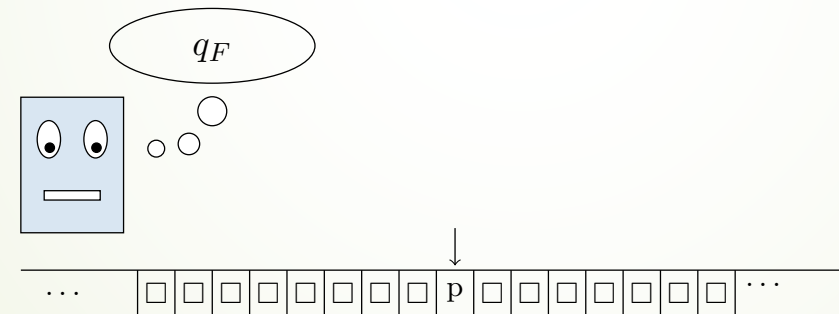
➡ ora possiamo eseguire la quintupla $\langle q_d, 1, \blacksquare, q_p, \text{destra} \rangle \in P$:



Esempio di macchina di Turing



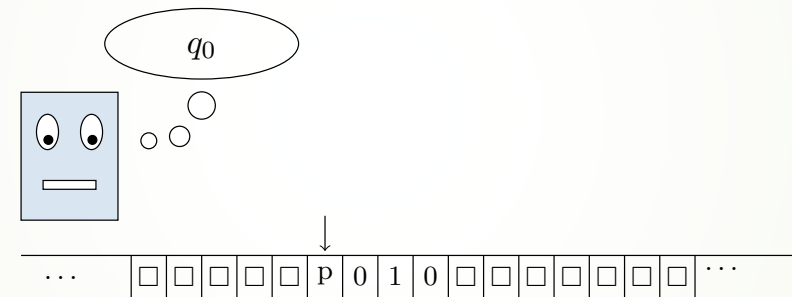
- ora possiamo eseguire la quintupla $\langle q_p, \square, p, q_F, \text{ferma} \rangle \in P$:



- Computazione terminata!

Esempio di macchina di Turing

- Naturalmente, sul nastro di $T_{\text{parità}}$ possiamo scrivere ciò che vogliamo:
 - ad esempio, possiamo scrivere la sequenza di caratteri p010
 - e vedere cosa succede facendo partire questa nuova computazione:



- .. Ohibò! P non contiene alcuna quintupla che inizia con la coppia (q_0, p)
 - quindi, nessuna quintupla può essere eseguita
- E di questo torneremo a parlare nelle prossime lezioni

Definizione di macchina di Turing

- Riassumiamo: una macchina di Turing ad un nastro è completamente caratterizzata dai cinque elementi
 - Σ , ossia, un insieme finito di caratteri che prende il nome di **alfabeto**
 - Q , ossia, un insieme finito di **stati interni**
 - uno stato interno particolare (generalmente indicato come q_0) chiamato **stato iniziale**
 - un sottoinsieme Q_F di Q di **stati finali**
 - un insieme $P \subseteq Q \times \Sigma \times \Sigma \times Q \times \{\text{sinistra, fermo, destra}\}$ di **quintuple**
 - che, sappiamo, deve essere non ambiguo
 - ossia, non contiene coppie di quintuple che hanno uguali i primi due elementi
 - ossia, in effetti, **P è una funzione**: $P: Q \times \Sigma \rightarrow \Sigma \times Q \times \{\text{sinistra, fermo, destra}\}$
- Quindi, possiamo dire che:
 - una macchina di Turing (ad un nastro) è una quintupla $\langle \Sigma, Q, q_0, Q_F, P \rangle$**
 - e dare per assodata l'esistenza di unità di controllo e nastro

Definizione di macchina di Turing

- ▶ E che dire di una macchina di Turing a più nastri? È (quasi) la stessa cosa:
- ▶ una macchina di Turing a k nastri è completamente caratterizzata da
 - ▶ un **alfabeto** Σ , ossia, un insieme finito di caratteri
 - ▶ un insieme finito Q di **stati interni**
 - ▶ uno stato interno **iniziale**
 - ▶ un sottoinsieme Q_F di Q di **stati finali**
 - ▶ un insieme P di **quintuple**, ove in questo caso una quintupla ha la forma
 $\langle q_1, (a_1, a_2, \dots, a_k), (b_1, b_2, \dots, b_k), q_2, (m_1, m_2, \dots, m_k) \rangle$
 - ▶ $(a_1, a_2, \dots, a_k)$ sono i caratteri che devono essere letti sui k nastri
 - ▶ a_1 è il carattere che deve essere letto sul nastro 1, a_2 è il carattere che deve essere letto sul nastro 2, ...
 - ▶ $(b_1, b_2, \dots, b_k)$ sono i caratteri che devono essere scritti sui k nastri (sovrascrivendo $(a_1, a_2, \dots, a_k)$)
 - ▶ b_1 è il carattere che deve essere scritto sul nastro 1, ...
 - ▶ $(m_1, m_2, \dots, m_k)$, sono i movimenti che devono essere eseguiti dalle k testine
 - ▶ m_1 è il movimento che deve essere compiuto dalla testina sul nastro 1, ...

Definizione di macchina di Turing

- Dunque, possiamo dire che, in generale,

una macchina di Turing è una quintupla $\langle \Sigma, Q, q_0, Q_F, P \rangle$

- e sulle macchine a k nastri torneremo nelle prossime lezioni

- OSSERVAZIONE

per capire quale sia il numero di nastri di una macchina di Turing $\langle \Sigma, Q, q_0, Q_F, P \rangle$ è sufficiente osservare le quintuple contenute in P:

- il numero di componenti del secondo elemento di una quintupla in P (che specifica ciò che deve essere letto sul/sui nastro/nastri per poter eseguire una quintupla) corrisponde al numero di nastri!
- ad esempio, se le quintuple di una macchina di Turing hanno la forma $\langle q_1, a_1, \dots \rangle$ allora si tratta di una macchina ad un nastro
- se le quintuple di una macchina di Turing hanno la forma $\langle q_1, (a_1, a_2), \dots \rangle$ allora si tratta di una macchina a due nastri
- e così via

Definizione di **M**acchina di Turing

- Dunque, possiamo dire che, in generale,
una macchina di Turing è una quintupla $\langle \Sigma, Q, q_0, Q_F, P \rangle$
- e, come dovremmo aver compreso, una macchina di Turing è la descrizione di un procedimento di calcolo
- ossia, è un **algoritmo descritto utilizzando le regole introdotte da Alan Turing**
 - in qualche modo, dunque, un programma scritto nel linguaggio progettato da Turing
- Le regole introdotte da Turing per descrivere procedimenti di calcolo costituiscono un **modello di calcolo**
 - tanto quanto ciascun linguaggio di programmazione, ad esempio, è un modello di calcolo
 - o tanti altri modelli ai quali accenneremo
- un modello di calcolo che prende il nome di **Macchina di Turing**
 - repetita iuvant

Esercizi: progetto di macchine di Turing

- Vediamo, ora, un esempio di macchina di Turing a due nastri
- ESERCIZIO: progettare una macchina di Turing a due nastri che, avendo sul primo nastro due numeri interi della stessa lunghezza, calcola il valore della loro somma scrivendo il risultato sul secondo nastro – ossia, si richiede di progettare una macchina di Turing che esegua la somma “in riga” di due numeri
- OSSERVAZIONE 1: poiché i due numeri devono essere scritti entrambi sul primo nastro e ciascuno di essi è una sequenza di cifre ‘0’, ‘1’, ... , ‘9’, è necessario utilizzare un ulteriore carattere (un *carattere separatore*) che permetta di separare i due numeri
 - scegliamo, quindi, il ‘+’ come carattere separatore
 - e, di conseguenza, assumiamo che sul primo nastro siano scritte due sequenze di cifre ‘0’, ‘1’, ... , ‘9’ separate da un ‘+’
- OSSERVAZIONE 2: nella macchina che stiamo per progettare, i due nastri hanno funzioni (e, dunque, significati) differenti
 - il secondo nastro serve soltanto a contenere il risultato – è il **nastro di output**
 - il primo nastro serve a contenere i dati del problema e a svolgere le azioni richieste per ottenere il risultato – è il **nastro di input e di lavoro**

Esercizi: progetto di macchine di Turing

- ESERCIZIO: progettare una macchina di Turing a due nastri che, avendo sul primo nastro due sequenze di cifre '0', '1', ... , '9' della stessa lunghezza separate da un '+', calcola il valore della loro somma scrivendo il risultato sul secondo nastro
- IDEA DELLA SOLUZIONE:
 - a partire dallo stato iniziale e con la testina posizionata sul carattere più a sinistra sul primo nastro, ci posizioniamo sul carattere più a destra del primo numero ossia, sul carattere che si trova immediatamente a sinistra di '+', senza mai muovere la testina sul secondo nastro
 - (*) poi, ricordando la cifra letta e il valore del riporto, cancelliamo quella cifra sostituendola con un '+' e ci posizioniamo sul carattere più a destra del secondo numero (che si trova a sinistra del ■)
 - poi, eseguiamo la somma fra la cifra del primo numero che ci stiamo ricordando e quello che stiamo leggendo: cancelliamo la cifra che stiamo leggendo sul primo nastro sostituendola con un ■ e scriviamo sul secondo nastro la cifra appena calcolata muovendo, in seguito, la sua testina a sinistra e ricordando il nuovo riporto
 - poi, ricordando il valore del riporto della somma delle due cifre appena calcolata, spostiamo la testina sul primo nastro a sinistra dell'ultimo '+'
 - e ripetiamo da (*)

Esercizi: progetto di macchine di Turing

- ▶ a partire dallo stato iniziale e con la testina posizionata sul carattere più a sinistra sul primo nastro, ci posizioniamo sul carattere più a destra del primo numero ossia, sul carattere che si trova immediatamente a sinistra di '+', senza mai muovere la testina sul secondo nastro
 - ▶ chiamiamo q_i lo stato iniziale,
 - ▶ indichiamo (in breve) con s sinistra, con f fermo e con d destra
 - ▶ e utilizziamo le quintuple $\langle q_i, (0, \blacksquare), (0, \blacksquare), q_i, (d, f) \rangle, \dots, \langle q_i, (9, \blacksquare), (9, \blacksquare), q_i, (d, f) \rangle,$
 - ▶ che abbreviamo con: **per ogni $x \in \{0, \dots, 9\}$ $\langle q_i, (x, \blacksquare), (x, \blacksquare), q_i, (d, f) \rangle$**
 - ▶ è una forma che useremo molto frequentemente
 - ▶ e la quintupla $\langle q_i, (+, \blacksquare), (+, \blacksquare), q_{is}, (s, f) \rangle$
 - ▶ osserviamo che lo stato q_i corrisponde all'azione "vai a destra finché incontri '+' "
 - ▶ quando incontriamo '+' dobbiamo smettere di muoverci a destra sul primo nastro e tornare indietro di una posizione: dobbiamo, cioè, eseguire un'azione diversa da quella regolata da q_i
 - ▶ per questo, quando sul primo nastro leggiamo '+', entriamo nello stato q_{is} al quale corrisponde l'azione "muoviti a sinistra sul primo nastro"

Esercizi: progetto di macchine di Turing

- (*) poi, ricordando la cifra letta e il valore del riporto, cancelliamo quella cifra sostituendola con un '+' e ci posizioniamo sul carattere più a destra del secondo numero (che si trova a sinistra del $\blacksquare$)
- memorizziamo la cifra letta e il valore del riporto nello stato: per ogni $x \in \{0, \dots, 9\}$, entriamo nello stato q_x^0 se leggiamo x e il riporto è 0, entriamo nello stato q_x^1 se leggiamo x e il riporto è 1
- e utilizziamo le quintuple: per ogni $x \in \{0, \dots, 9\}$ $\langle q_{is}, (x, \blacksquare), (+, \blacksquare), q_x^0, (d, f) \rangle$
 - in questo modo, si crea una sequenza di '+' via via che le cifre del primo numero vengono cancellate
- poi, per posizionarci sul carattere più a destra del secondo numero (che si trova a sinistra del $\blacksquare$) utilizziamo le quintuple:
per ogni $x \in \{0, \dots, 9\}$ $\langle q_x^0, (x, \blacksquare), (x, \blacksquare), q_x^0, (d, f) \rangle$ e $\langle q_x^1, (x, \blacksquare), (x, \blacksquare), q_x^1, (d, f) \rangle$
- e poi le quintuple $\langle q_x^0, (\blacksquare, \blacksquare), (\blacksquare, \blacksquare), q_{xs}^0, (s, f) \rangle$ e $\langle q_x^1, (\blacksquare, \blacksquare), (\blacksquare, \blacksquare), q_{xs}^1, (s, f) \rangle$
- di nuovo: l'azione corrispondente a q_x^0 e a q_x^1 è "vai a destra finché incontri $\blacksquare$ "
- quando incontriamo $\blacksquare$ dobbiamo smettere di muoverci a destra sul primo nastro e tornare indietro di una posizione: dobbiamo, cioè, eseguire un'azione diversa da quella regolata da q_x^0 e a q_x^1 e per questo, quando sul primo nastro leggiamo $\blacksquare$, entriamo in uno degli stati q_{xs}^0 o q_{xs}^1 ai quale corrisponde l'azione "muoviti a sinistra sul primo nastro"

Esercizi: progetto di macchine di Turing

- poi, eseguiamo la somma fra la cifra del primo numero che ci stiamo ricordando e quello che stiamo leggendo: cancelliamo la cifra che stiamo leggendo sul primo nastro sostituendola con un $\blacksquare$, scriviamo sul secondo nastro la cifra appena calcolata muovendo, in seguito, la sua testina a sinistra, e memorizziamo il nuovo riporto

- utilizziamo le (200) quintuple: $\langle q_{0s}^0, (0, \blacksquare), (\blacksquare, 0), q^0, (f, s) \rangle$, $\langle q_{0s}^1, (0, \blacksquare), (\blacksquare, 1), q^0, (f, s) \rangle$,

$$\langle q_{1s}^0, (0, \blacksquare), (\blacksquare, 1), q^0, (f, s) \rangle \text{ e } \langle q_{1s}^1, (0, \blacksquare), (\blacksquare, 2), q^0, (f, s) \rangle$$

...

$$\langle q_{6s}^0, (3, \blacksquare), (\blacksquare, 9), q^0, (f, s) \rangle \text{ e } \langle q_{6s}^1, (3, \blacksquare), (\blacksquare, 0), q^1, (f, s) \rangle$$

...

$$\langle q_{9s}^0, (9, \blacksquare), (\blacksquare, 8), q^1, (f, s) \rangle \text{ e } \langle q_{9s}^1, (9, \blacksquare), (\blacksquare, 9), q^1, (f, s) \rangle$$

- in cui q^0 e q^1 sono gli stati che corrispondono a "torno a sinistra a cercare la nuova cifra del primo addendo ricordando il riporto"
- anche in questo caso, il valore del riporto è memorizzato nello stato

Esercizi: progetto di macchine di Turing

- poi, ricordando il valore del riporto della somma delle due cifre appena calcolata, spostiamo la testina sul primo nastro a sinistra dell'ultimo '+'
 - utilizziamo le quintuple:
per ogni $x \in \{0, \dots, 9\}$ $\langle q^0, (x, \blacksquare), (x, \blacksquare), q^0, (s, f) \rangle$ e $\langle q^1, (x, \blacksquare), (x, \blacksquare), q^1, (s, f) \rangle$
 - non appena viene letto un '+' sul primo nastro, è necessario cambiare stato e continuare a muoversi a sinistra: $\langle q^0, (+, \blacksquare), (+, \blacksquare), q_s^0, (s, f) \rangle$ e $\langle q^1, (+, \blacksquare), (+, \blacksquare), q_s^1, (s, f) \rangle$
 - per fare in modo che la prima cifra incontrata al termine della sequenza di '+' venga riconosciuta
 - quando, poi, viene letta una cifra sul primo nastro, si ricomincia dal passo (*):
per ogni $x \in \{0, \dots, 9\}$ $\langle q_s^0, (x, \blacksquare), (+, \blacksquare), q_x^0, (d, f) \rangle$ e $\langle q_s^1, (x, \blacksquare), (+, \blacksquare), q_x^1, (d, f) \rangle$ (⊙)
 - se, invece, sul primo nastro viene letto un $\blacksquare$ allora la somma è terminata (perché i due numeri hanno lo stesso numero di cifre) e, dunque, viene scritto '1' sul nastro di output se il riporto è 1 e poi la macchina termina la computazione:
 $\langle q_s^0, (\blacksquare, \blacksquare), (\blacksquare, \blacksquare), q_F, (d, f) \rangle$ e $\langle q_s^1, (\blacksquare, \blacksquare), (\blacksquare, 1), q_F, (d, f) \rangle$
- Osserviamo che il punto "e ripetiamo da (*)" si realizza tornando in uno degli stati q_x^0 o q_x^1 al passo (⊙)

Esercizi: progetto di macchine di Turing

- OSSERVAZIONE: la macchina che calcola la “somma in riga” di due numeri *funziona soltanto se i due numeri hanno lo stesso numero di cifre*
 - ESERCIZIO: progettare una macchina di Turing a due nastri che, avendo sul primo nastro due sequenze di cifre '0', '1', ... , '9' separate da un '+', scrive sul secondo nastro il valore della somma dei due numeri rappresentati dalle due sequenze quando i due numeri non sono composti dallo stesso numero di cifre
 - ESERCIZIO (complesso): progettare una macchina di Turing ad un solo nastro che, avendo sul nastro due sequenze di cifre '0', '1', ... , '9' separate da un '+', scrive (in una posizione opportuna) il valore della somma dei due numeri rappresentati dalle due sequenze – ossia, si richiede di progettare una macchina di Turing che esegua la somma “in riga” di due numeri
 - questa macchina è descritta nel paragrafo 1.6 della dispensa 1 e funziona anche quando gli operandi non hanno lo stesso numero di cifre: provare a svolgere l'esercizio **senza** guardare quella macchina
- Correggerò gli esercizi degli studenti che me li invieranno via e-mail

macchine di Turing

- Il modello di calcolo Macchina di Turing richiede che in ogni macchina l'insieme degli stati e l'alfabeto abbiano cardinalità **finita** – e lo stesso vale per il numero di nastri
- Cerchiamo di capire perché ripensando, ancora una volta, alla somma di due numeri
- Se fosse possibile avere un numero infinito di stati interni e un numero infinito di caratteri dell'alfabeto, il progetto di una macchina di Turing che esegue la “somma in riga” di due numeri (scrivendo il risultato sul secondo nastro) sarebbe banale: basterebbe porre $\Sigma = \mathbb{N} \cup \{+\}$ e $Q = \{q_x : x \in \mathbb{N}\} \cup \{q_i, q_F\}$ e utilizzare le quintuple
 - per ogni $n \in \mathbb{N}$ $\langle q_i, (n, \square), (n, \square), q_n, (d, f) \rangle$, che legge il primo numero (scritto in una singola cella del primo nastro), entra nello stato interno corrispondente e muove la testina del primo nastro a destra per andare a cercare il secondo numero
 - per ogni $n \in \mathbb{N}$ $\langle q_n, (+, \square), (+, \square), q_n, (d, f) \rangle$, che “scavalca” il ‘+’
 - per ogni $n, m \in \mathbb{N}$ $\langle q_n, (m, \square), (m, h), q_F, (d, f) \rangle$, dove $h = m+n$
 - Facile!

macchine di Turing

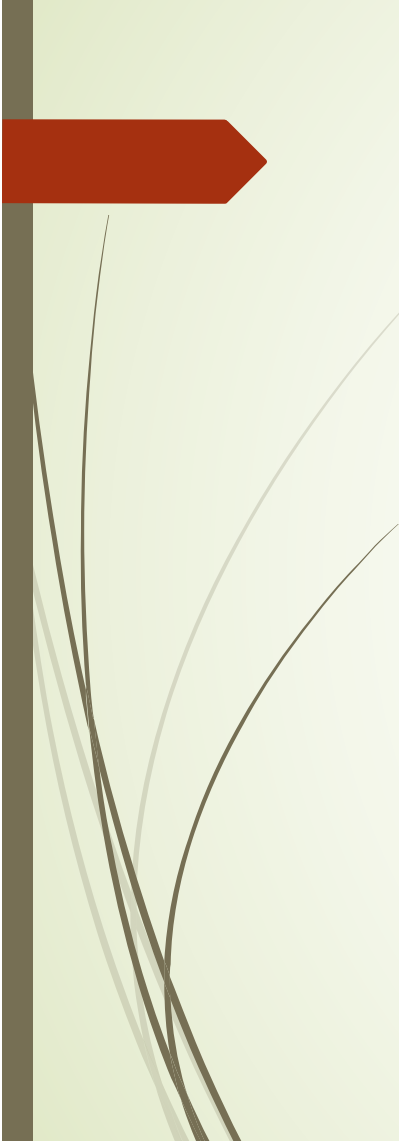
- Se fosse possibile avere un numero infinito di stati interni e un numero infinito di caratteri dell'alfabeto, il progetto di una macchina di Turing che esegue la "somma in riga" di due numeri (scrivendo il risultato sul secondo nastro) sarebbe banale: basterebbe porre $\Sigma = \mathbb{N}$ e $Q = \mathbb{N} \cup \{q_i, q_F\}$ e utilizzare le quintuple
 - per ogni $n \in \mathbb{N}$ $\langle q_i, (n, \square), (n, \square), q_n, (d, f) \rangle$,
 - per ogni $n \in \mathbb{N}$ $\langle q_n, (+, \square), (+, \square), q_n, (d, f) \rangle$,
 - per ogni $n, m \in \mathbb{N}$ $\langle q_n, (m, \square), (m, h), q_F, (d, f) \rangle$, dove $h = m+n$
- Facile! *Troppo* facile... E, infatti, la cosa non funziona
- Il punto è che questa "macchina" non potremmo **costruirla**
 - possiamo pensare che gli stati siano realizzati, ad esempio, mediante lampadine: a ciascuno stato corrisponde una lampadina (che è accesa o spenta a seconda che la macchina si trovi o meno in quello stato)
 - e che ciascuna quintupla sia una sorta di circuito che si occupa, fra l'altro, di controllare, accendere e spegnere le lampadine
 - dovremmo, dunque, predisporre tante lampadine e tanti circuiti quanti sono i numeri naturali
 - ... e mi sa che non ce la faremmo nel corso della nostra vita ...

macchine di Turing

- ▶ Fuor di metafora (di lampadine e bulloni), il punto è che la forma abbreviata “per ogni $x \in A$ ” dobbiamo poterla scrivere in forma esplicita
 - ▶ ossia, anche se la notazione implicita “per ogni $x \in A$ ” è parecchio comoda,
 - ▶ dobbiamo poter scrivere esplicitamente **tutti** gli stati e **tutte** le quintuple che occorrono a descrivere completamente una macchina di Turing
 - ▶ e lo stesso vale per il numero di nastri
- ▶ e affinché questo sia possibile è necessario che il numero di stati, il numero di simboli dell'alfabeto, il numero di quintuple e il numero di nastri siano **finiti**
- ▶ ossia, che numero di stati, numero di simboli dell'alfabeto, numero di quintuple e numero di nastri siano scelti una volta per tutte,
 - ▶ e non di volta in volta a seconda del dato particolare sul quale vogliamo operare
 - ▶ **non** possiamo, ribadiamo, scrivere per ogni $n \in \mathbb{N}$ $\langle q_i, (n, \blacksquare), (n, \blacksquare), q_n, (d, f) \rangle$,
- ▶ ossia, è necessario che numero di stati, numero di simboli dell'alfabeto, numero di quintuple e numero di nastri siano **costanti**
 - ▶ **ossia, indipendenti dall'input**

Tante definizioni per le macchine di Turing

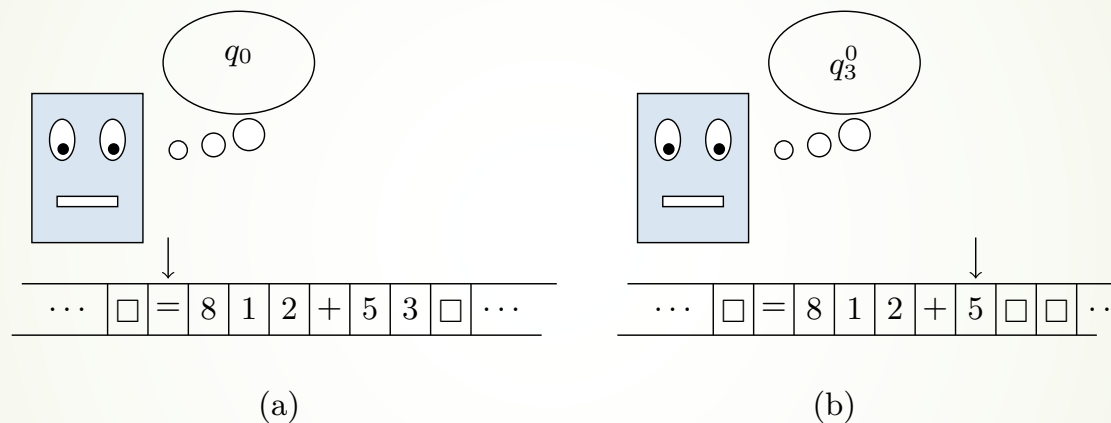
- Nel paragrafo 2.1 della dispensa 2 vengono presentate alcune definizioni formali relative alle macchine di Turing:
 - parole
 - stati globali
 - transizioni
 - computazioni
- Queste definizioni devono essere tenute **sempre presenti!!!!**
- Osservate che viene utilizzata la parola *deterministica*: questo significa che P è una funzione (avremo tempo e modo di affrontarla bene e a lungo, questa questione)
- Innanzi tutto: dato un alfabeto finito Σ , una **parola** su Σ è una sequenza **finita** di elementi di Σ
 - ad esempio, aba è una parola sull'alfabeto $\Sigma = \{ a, b, c \}$
- L'**insieme della parole** su un alfabeto Σ si indica con **Σ^***



Stati globali

- ▶ Uno stato globale SG di una macchina di Turing è una “fotografia” della macchina ad un certo istante
- ▶ Formalmente, uno *stato globale* di una macchina ad un nastro T ad un certo istante
 - ▶ contiene una descrizione della porzione non blank del nastro di T, della posizione della testina (e, dunque, del carattere da essa letto) e dello stato interno
 - ▶ ed è rappresentato mediante la sequenza di caratteri (non blank) contenuti sul nastro in cui al carattere letto dalla testina è premesso lo stato interno
 - ▶ vedremo un esempio alla prossima pagina
- ▶ naturalmente, possiamo definire anche lo stato globale di una macchina a k nastri (con k costante!)
 - ▶ ma non lo facciamo formalmente
 - ▶ lo vedremo, informalmente in uno dei prossimi esempi

Esempi: stati globali



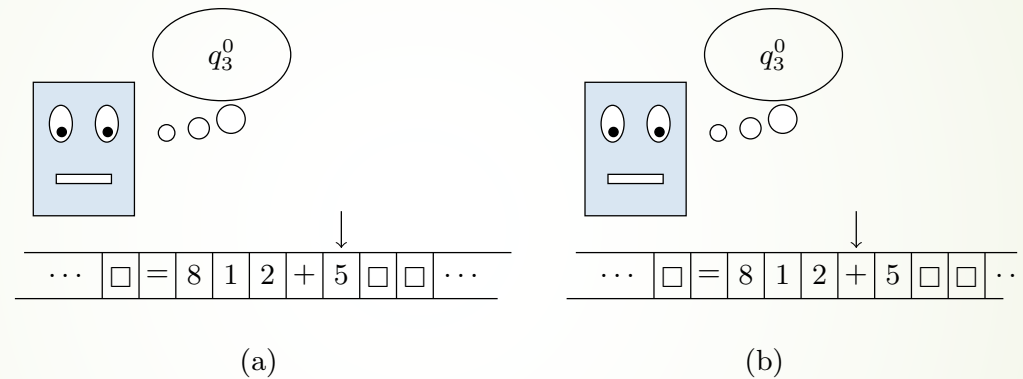
- (a) Lo stato globale iniziale SG_0 di una computazione della macchina che calcola la somma di due numeri vista a lezione: $q_0 = 8\ 1\ 2 + 5\ 3$
- (b) uno stato globale successivo SG della stessa computazione:
 $= 8\ 1\ 2 + q_0^3\ 5$



Transizioni

- ▶ Una transizione dallo stato globale SG_1 allo stato globale SG_2 avviene quando viene eseguita una quintupla che trasforma SG_1 in SG_2
- ▶ Formalmente, se $T = \langle \Sigma, Q, q_0, Q_F, P \rangle$ è una macchina di Turing ad un nastro, esiste una **transizione** da SG_1 a SG_2 se esiste una quintupla $\langle q, x, x', q', m \rangle \in P$ tale che
 - ▶ in SG_1 T si trova nello stato interno $q \in Q$
 - ▶ in SG_1 la testina di T sta scandendo una cella che contiene il carattere $x \in \Sigma$
 - ▶ in SG_2 la cella che in SG_1 conteneva il carattere x contiene il carattere $x' \in \Sigma$
 - ▶ in SG_2 T si trova nello stato interno $q' \in Q$
 - ▶ in SG_2 la testina di T sta scandendo la cella che si trova in posizione m rispetto a quella che stava scandendo in SG_1
- ▶ il concetto può essere facilmente esteso a macchine a più nastri
 - ▶ con qualche tecnicismo in più, che non affrontiamo

Esempi: transizioni



- Un esempio di transizione dallo stato globale $=812+ q_0^3 5$ (in figura (a)) allo stato globale $=812 q_0^3 +5$ (in figura (b)) a seguito dell'esecuzione della quintupla $\langle q_0^3, 5, 5, q_0^3, \text{sinistra} \rangle$

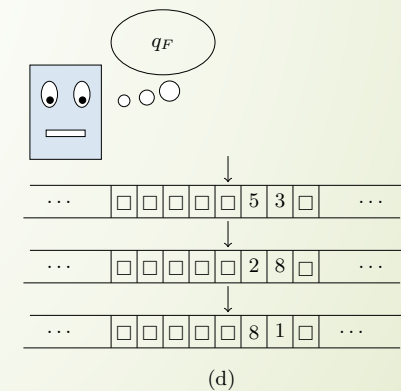
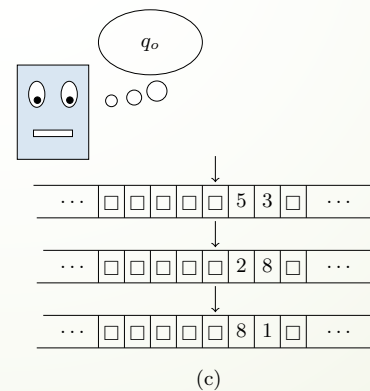
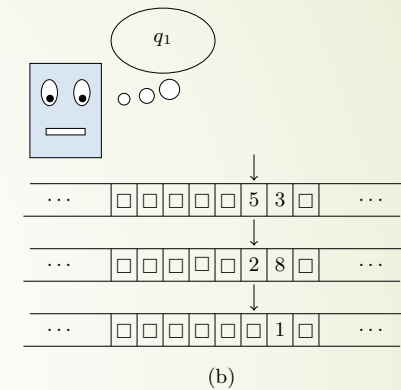
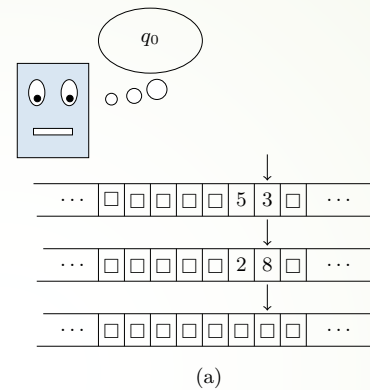
Computazione

- Informalmente, una computazione di una macchina di Turing **deterministica** a un nastro $T = \langle \Sigma, Q, q_0, Q_F, P \rangle$ è l'esecuzione delle quintuple di T su una sequenza di caratteri scritti sul suo nastro
 - e analogamente per le macchine a più nastri
- Formalmente: una **computazione** di una macchina di Turing T è una sequenza $SG_0, SG_1, SG_2, \dots, SG_h, \dots$ di stati globali di T tali che
 - SG_0 è uno **stato globale iniziale**, ossia, uno stato globale nel quale lo stato interno è q_0 e la testina è posizionata sul carattere più a sinistra scritto sul nastro
 - per ogni $0 \leq i \leq h-1$, esiste una transizione da SG_i a SG_{i+1} oppure nessuna quintupla può essere eseguita nello stato globale da SG_i e per ogni $h \geq i+1$ SG_h non è definito
- se esiste un indice h tale che SG_h è uno stato globale dal quale non può avvenire alcuna transizione allora la computazione **termina**
 - e questo accade quando lo stato interno nel quale T si trova in SG_h è uno stato finale oppure P non contiene una quintupla che possa essere eseguita in SG_h

Esempi: computazione

- Una computazione dallo stato globale $(5, 2, \square) q_0(3, 8, \square)$ (a) allo stato globale $q_1(5, 2, \square)(3, 8, 1)$ (b) allo stato globale $q_0(\square, \square, \square)(5, 2, 8)(3, 8, 1)$ (c) allo stato globale $q_F(\square, \square, \square)(5, 2, 8)(3, 8, 1)$ (d)

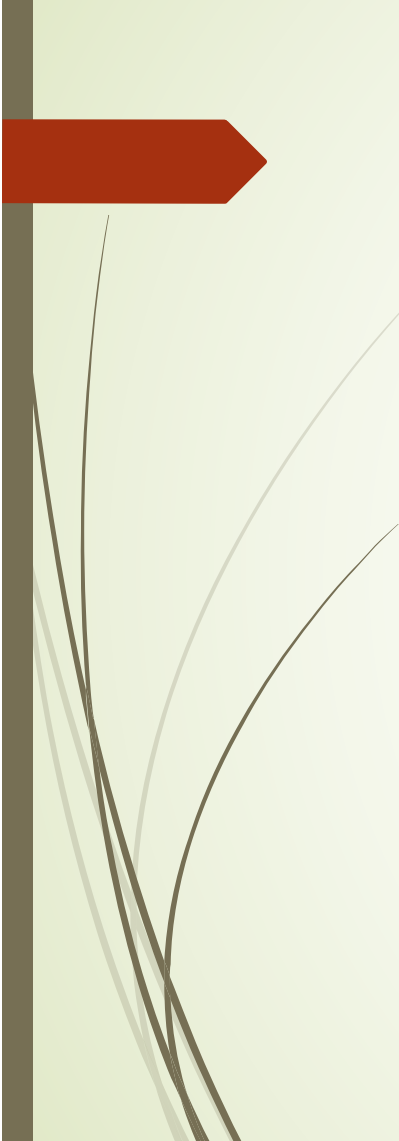
- E così, abbiamo visto anche *un modo* di rappresentare uno stato globale in una macchina a più nastri





Trasduttori e Riconoscitori

- ▶ Nel paragrafo 2.2 della dispensa 2 vengono definiti due tipi di macchine di Turing.
- ▶ Le macchine di tipo **trasduttore** sanno calcolare il valore di una funzione qualsiasi
 - ▶ ad esempio, un trasduttore sa calcolare la funzione $f(a,b)=a+b$.
- ▶ Assumeremo sempre che le macchine di Turing di tipo trasduttore dispongano di un **nastro di output** sul quale scrivono il valore della funzione che hanno calcolato
- ▶ Un trasduttore ha un solo stato finale con il quale terminare la computazione: lo stato q_F



Trasduttori e Riconoscitori

- ▶ Nel paragrafo 2.2 della dispensa 2 vengono definiti due tipi di macchine di Turing.
- ▶ Le macchine di tipo **riconoscitore** sanno calcolare soltanto il valore di funzioni booleane
 - ▶ ossia, funzioni il cui valore è 0 oppure 1
- ▶ e non dispongono di un nastro di output.
- ▶ **Il valore calcolato da un riconoscitore viene memorizzato nello stato interno con il quale la macchina termina la computazione:** q_A se il valore della funzione è 1, q_R se il valore della funzione è 0
 - ▶ quindi, ogni riconoscitore ha due stati finali: q_A e q_R
- ▶ Diciamo che un riconoscitore T **accetta** x se la computazione $T(x)$ termina in q_A e che la macchina T **rigetta** x se la computazione $T(x)$ termina in q_R

Esito di una computazione

- Sempre nel paragrafo 2.2 della dispensa 2 viene introdotto il concetto di *esito di una computazione*
- Data una macchina di Turing T ed un suo input x , l'esito della computazione $T(x)$ è indicato con $o_T(x)$ – informalmente è “il risultato” della computazione, la risposta che ci dà la macchina rispetto all'istanza x del problema che le abbiamo chiesto di risolvere
- Se T è una macchina di tipo *trasduttore*, allora $o_T(x)$ è la parola scritta da T sul nastro di output al termine della computazione $T(x)$ (ossia, **quando T ha raggiunto lo stato q_F**): ad esempio, se T è il trasduttore che calcola la funzione $f(a,b)=a+b$. allora $o_T(15,6)=21$.
- Se T è una macchina di tipo *riconoscitore*, allora $o_T(x)$ è lo stato interno con il quale la macchina termina la computazione $T(x)$: ad esempio, se T è la macchina che decide se una parola è palindroma, allora $o_T(abba) = q_A$ e $o_T(baaba) = q_R$.



Tanto per essere chiari

- ▶ Nel seguito di questo corso considereremo quasi sempre macchine di Turing di tipo riconoscitore
 - ▶ e questo ci semplificherà la vita, perché non dovremo occuparci del nastro di output
- ▶ Facciamo così: quando dirò “macchina di Turing” mi riferirò sempre ad una macchina di tipo riconoscitore
- ▶ Se vorrò riferirmi ad un trasduttore dirò “macchina di Turing di tipo trasduttore”