Appunti del corso IN110 Algoritmi e Strutture Dati

3 – Modelli di calcolo

Prof. Marco Liverani

(liverani@mat.uniroma3.it – http://www.mat.uniroma3.it/users/liverani/IN110)

Sommario

- Modelli di calcolo
- Cenni sulla macchina di Turing
- Un esempio di grafo degli stati e di matrice di transizione
- Macchina di Turing Universale, Tesi di Church-Turing, problemi non calcolabili
- Cenni sulla macchina di Von Neumann

Modelli di calcolo

- Prima di specializzarci nella costruzione di algoritmi (e poi di programmi) per un particolare calcolatore è bene convincersi che **non** esiste un unico *modello di calcolo*
- Ogni modello richiede un differente approccio alla soluzione dei problemi, proponendo un modello **teorico** di macchina di calcolo (di esecutore) con *caratteristiche* e *capacità* differenti
- Basarsi su un modello di macchina astratta è necessario per poter confrontare l'**efficienza** di due differenti *algoritmi*, a prescindere dalla loro implementazione e dalla velocità del computer su cui eseguiremo il programma codificato in linguaggio macchina
- Basandoci su un modello di calcolo **astratto**, ma **semplificato**, è più semplice dimostrare in termini rigorosi la *correttezza* di un determinato algoritmo

La Macchina di Turing

Il matematico Alan Turing (1912 – 1954)

- Ideata nel 1936 dal matematico inglese Alan Turing una delle figure più importanti che hanno contribuito alla definizione e allo sviluppo della teoria Informatica
- È una macchina **astratta** basata su due componenti:
  - Un **nastro infinito** (da questa caratteristica ne segue il fatto che la macchina di Turing non è realizzabile praticamente) su cui la macchina può leggere e scrivere mediante
  - Una **testina di lettura e scrittura** che, scorrendo sul nastro, è in grado di leggerne e modificarne il contenuto
- La macchina di Turing è un Automa a Stati Finiti Deterministico con in più un nastro (che è al tempo stesso memoria, unità di input e di output) di lunghezza infinita

Automa a Stati Finiti Deterministico

- Un **automa a stati finiti deterministico** è una quintupla $\langle \mathcal{A}, \mathcal{S}, \delta, s_0, F \rangle$, dove:
  - $\mathcal{A}$ è un **alfabeto di simboli** che l'automa è in grado di elaborare (confrontare, leggere, scrivere)
  - $\mathcal{S}$ è l'**insieme degli stati** in cui si può trovare l'automa nel corso dell'elaborazione
  - δ è la **funzione di transizione** che fa passare l'automa da uno stato di $\mathcal{S}$ ad un altro sulla base del simbolo che sta elaborando e dello stato in cui si trova:

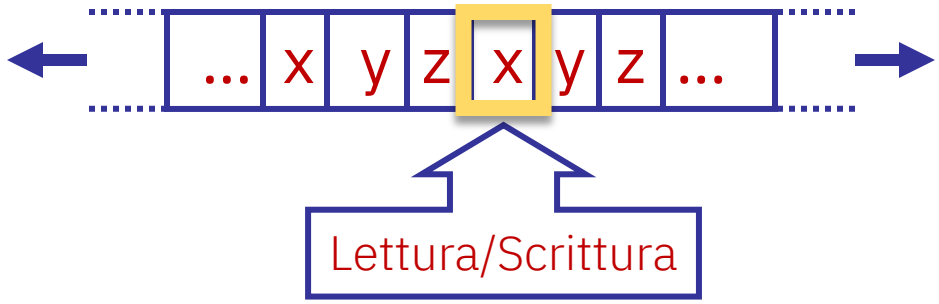
$$\delta : \mathcal{A} \times \mathcal{S} \rightarrow \mathcal{A} \times \mathcal{S} \times \{\textit{avanti}, \textit{indietro}, \textit{ferma}\}$$

  - $s_0 \in \mathcal{S}$ è lo **stato iniziale**, lo stato in cui si trova l'automa all'inizio dell'elaborazione
  - $F \subset \mathcal{S}$ è l'**insieme degli stati terminali**: quando l'automa raggiunge uno di questi stati l'elaborazione termina
- Gli stati terminali possono avere un significato diverso, a seconda del problema che si intende risolvere con l'automa (es.: successo, insuccesso, ecc.)

La Macchina di Turing 2

- Il funzionamento della macchina è basato sul **cambiamento di stato** sulla base del contenuto (del simbolo) che è presente sul nastro in corrispondenza della testina di lettura/scrittura
- Dopo aver letto il contenuto della posizione corrente del nastro, la macchina, sulla base dello stato in cui si trova, è in grado di passare in un altro stato, scrivere qualcosa nella posizione corrente del nastro ed infine spostarsi a destra o a sinistra sul nastro stesso
- Il **nastro è infinito**, mentre gli **stati sono in quantità finita**
- Il fatto che sia **deterministico** sta ad indicare che per ogni coppia "stato/simbolo" che viene incontrata dalla macchina, è **univocamente** determinata l'azione compiuta (scrittura e spostamento del nastro) e il nuovo stato in cui passa la macchina stessa; in questo modo la macchina può eseguire solo un'operazione per volta

La Macchina di Turing



- Per ogni problema che si intende risolvere è necessario progettare una macchina di Turing adeguata
- Per farlo è necessario definire:
  - L'**alfabeto** dei simboli che è possibile leggere e scrivere sul nastro
  - Gli **stati** in cui si può trovare la macchina
  - Le **transizioni** da uno stato ad un altro
  - Lo **stato iniziale** ed un insieme di **stati finali**
- Per rappresentare l'automa possiamo disegnare un **grafo degli stati**, ovvero una **matrice di transizione**

La Macchina di Turing

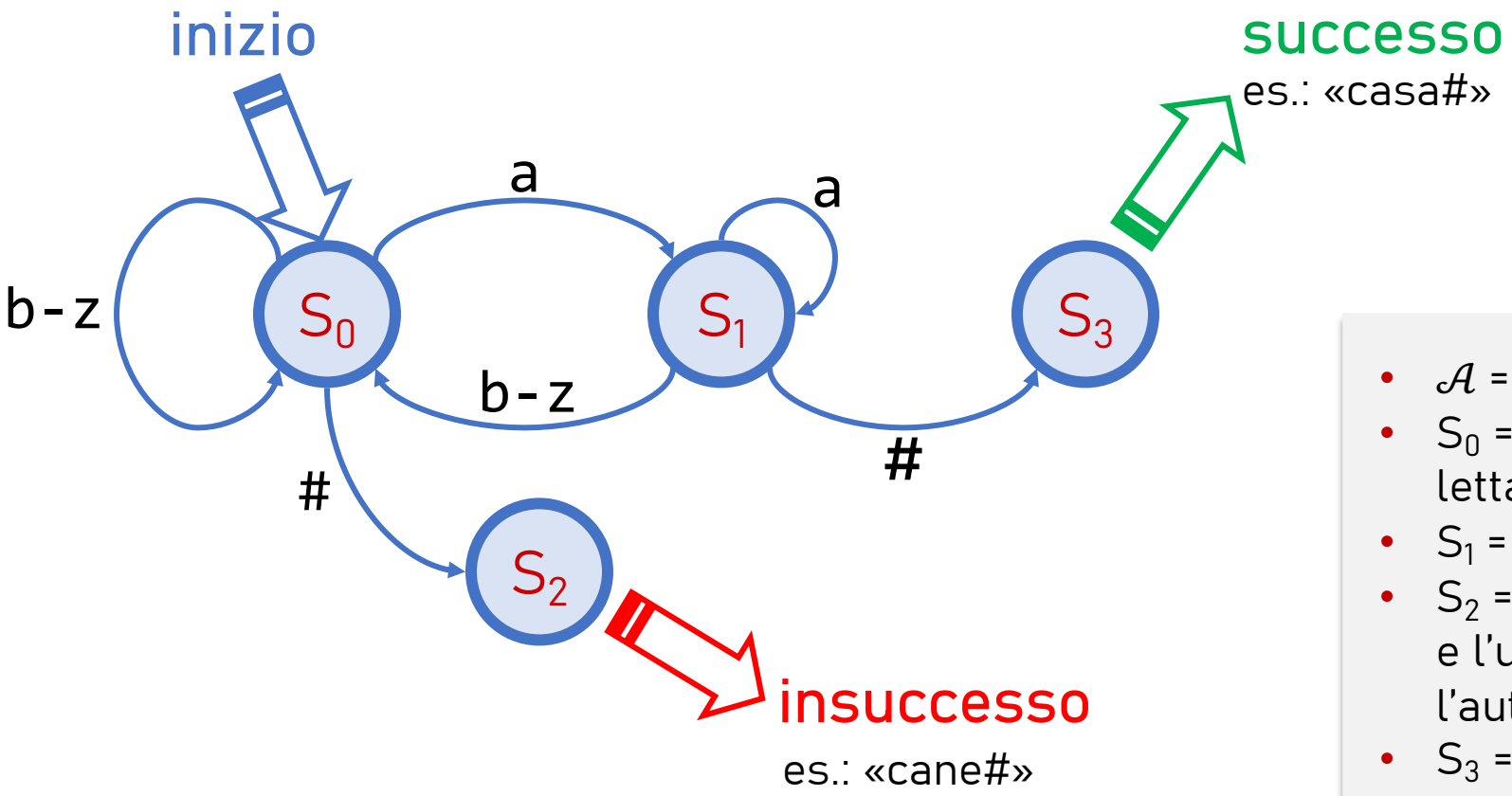
- Esempio: letta una stringa di caratteri alfabetici, stabilire se termina o meno con la lettera «a»

- Alfabeto: a, b, c, …, x, y, z, # (il carattere «#» marcherà la fine della stringa)

- Strategia risolutiva:
  - Partendo dal primo carattere della stringa la macchina scorre il nastro verso destra fino a quando non incontro il carattere «#»
  - Scorrendo il nastro verso destra, ogni volta che la macchina trova un carattere «a» si pone in uno *stato di preallarme* (S_1) e torna nello *stato di quiete* (S_0) quando invece incontra una lettera dalla «b» alla «z»
  - Se quando si incontra il «#» la macchina si trova nello stato S_0 allora questo significa che la stringa non termina con la lettera «a» (e passa quindi nello stato S_2), altrimenti, se si trova in S_1, vuol dire che la stringa termina con la lettera «a» (e passa quindi nello stato S_3)

La Macchina di Turing

Grafo delle transizioni di stato:



- $\mathcal{A}$ = {a, b, c, ..., x, y, z, #}
- S_0 = stato iniziale, oppure non è stata letta la lettera «a»
- S_1 = è stata letta la lettera «a»
- S_2 = la parola è finita (è stato letto «#») e l'ultima lettera non era la «a» perché l'automa era nello stato S_0
- S_3 = la parola è finita (è stato letto «#») e l'ultima lettera era la «a» perché l'automa si trovava nello stato S_1

La Macchina di Turing

- È possibile descrivere una macchina di Turing attraverso una matrice di transizione; ad esempio:

	S_0	S_1	S_2	S_3
a	-, Dx, S_1	-, Dx, S_1	n.d.	n.d.
b-z	-, Dx, S_0	-, Dx, S_0	n.d.	n.d.
#	-, -, S_2	-, -, S_3	No	Sì

- Per ogni coppia carattere/stato la matrice indica l'azione che deve intraprendere la macchina (scrivere, spostarsi di una posizione sul nastro, passare ad un altro stato)

La Macchina di Turing

- Una Macchina di Turing (MdT) possiamo quindi descriverla come un insieme di «istruzioni» definite attraverso una quintupla $\langle s_i , c , s_f, c', m \rangle$, dove s_i è lo stato in cui si trova la macchina e c è il carattere che legge sul nastro, s_f è lo stato in cui passa la macchina (può essere $s_f = s_i$), c' è il carattere che viene scritto sul nastro (può essere $c' = c$), m è lo spostamento sul nastro (può essere «sinistra», «destra», «fermo»)
- Siccome i valori che possono essere assunti da ciascun elemento di una quintupla (gli stati, i caratteri e gli spostamenti) sono di cardinalità finita, possiamo associare a ciascun valore un **numero naturale dispari** (es.: 1 = spostamento a sinistra, 3 = spostamento a destra, 5 = fermo, 7 = stato S_0, 9 = stato S_1, ...)
- Ogni istruzione/quintupla può essere così identificata da una quintupla di numeri naturali **dispari**: $\langle a, b, c, d, e \rangle$

La Macchina di Turing

- Se $\langle a, b, c, d, e \rangle$ sono i cinque numeri che rappresentano la k-esima quintupla/istruzione della MdT, possiamo identificare tale quintupla/istruzione con un numero naturale **pari** costruito con l'espressione

$$I_k = 2^a \times 3^b \times 5^c \times 7^d \times 11^e$$

  usando come basi delle potenze i primi cinque numeri primi

- L'intera MdT è data quindi dalla sequenza (finita) di numeri $I_1, I_2, I_3, \ldots, I_n$ e possiamo quindi identificarla con il numero naturale

$$M = 2^{I_1} \times 3^{I_2} \times 5^{I_3} \times \cdots \times p_n^{I_n}$$

  dove $2, 3, 5, \ldots, p_n$ sono i primi n numeri primi

- I numeri I_k e M sono tutti diversi tra loro, anche perché gli I_k hanno esponenti dispari, mentre i numeri M hanno esponenti pari

La Macchina di Turing Universale

- Siccome ogni Macchina di Turing è quindi identificata univocamente da un determinato numero naturale M ottenuto mediante una fattorizzazione delle sue istruzioni, possiamo ipotizzare la definizione di una **Macchina di Turing Universale** che acquisisca in input
  - una Macchina di Turing M (rappresentata sul nastro da una opportuna codifica del numero naturale M che la rappresenta)
  - i dati su cui la MdT M deve operare
- La Macchina di Turing Universale «eseguirà» le istruzioni della MdT M e produrrà in output il risultato
- La Macchina di Turing Universale è quindi una MdT programmabile attraverso MdT

Problemi non calcolabili

- Problema della fermata («halting problem»):

  data una Macchina di Turing X (o un algoritmo deterministico), definire una Macchina di Turing Y (o un algoritmo deterministico) che verifichi se X termina dopo un numero finito di operazioni per qualsiasi istanza del problema

- È il primo esempio di problema non calcolabile, ossia non è possibile definire una Macchina di Turing (o un algoritmo deterministico) che lo risolva; in altri termini non è un problema risolubile (calcolabile) attraverso una Macchina di Turing Universale

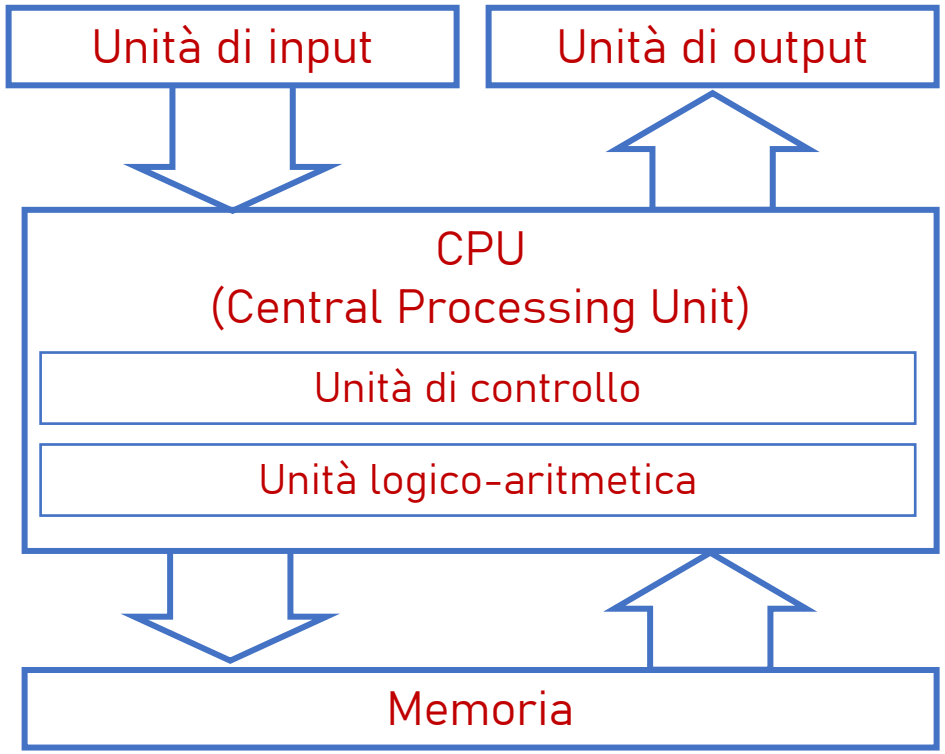
La Tesi di Church-Turing

- Anche assumendo altri modelli di calcolo (es.: **λ**-calcolo di Alonzo Church), diversi dalla macchina di Turing, si afferma che tutti questi modelli sono fra loro **equivalenti**
- Un problema **calcolabile** secondo il modello della macchina di Turing è calcolabile anche rispetto ad altri modelli

Il logico-matematico Alonzo Church (1903 – 1995)

Il modello di macchina di Von Neumann 1

- Gli algoritmi che progetteremo si basano sulle caratteristiche e sulle capacità di base di un calcolatore basato su un *modello di calcolo ideale*, noto come **Modello di Von Neumann**, proposta nel **1945** dallo scienziato ungherese/statunitense John Von Neumann



John Von Neumann
(1903 – 1957)

Il modello di macchina di Von Neumann 2

- Nel modello di Von Neumann è facile riconoscere gli elementi costitutivi di ogni moderno calcolatore
- Le componenti sono le seguenti:
  - Le **unità di input** tramite cui la macchina acquisisce informazioni dall'esterno
  - Le **unità di output** tramite cui la macchina produce (stampa) informazioni all'esterno
  - L'**unità centrale di elaborazione** (CPU) che elabora le istruzioni del programma (i passi dell'algoritmo), composta da due elementi:
    - **Unità di controllo**: stabilisce l'ordine con cui devono essere eseguite le operazioni
    - **Unità logico-aritmetica**: esegue operazioni aritmetiche e risolve espressioni logiche
  - La **memoria** in cui l'unità centrale deposita ed estrae le informazioni per poterle elaborare
- I dati e il programma eseguito dalla macchina sono memorizzati nella stessa unità di memoria

Evoluzione del pensiero scientifico nel ‘900

Lo sviluppo della teoria della calcolabilità e dell’informatica è andata di pari passo con lo sviluppo della matematica, della fisica quantistica e della fisica nucleare, contaminandosi e supportandosi vicendevolmente

Nome	Nazionalità	Nascita	Morte	Principali contributi
Max Plank	Germania	1858	1947	Fisica, Fisica quantistica
David Hilbert	Germania	1862	1943	Matematica
Bertrand Russel	Gran Bretagna	1872	1970	Logica matematica
Albert Einstein	Germania / Svizzera / Stati Uniti	1879	1955	Fisica, Teoria della relatività
Niels Bohr	Danimarca	1885	1962	Fisica quantistica
Erwin Schrödinger	Austria	1887	1961	Fisica quantistica
Wolfgang Pauli	Austria / Svizzera	1900	1958	Fisica quantistica, Principio di esclusione
Enrico Fermi	Italia / Stati Uniti	1901	1954	Fisica nucleare
Werner Heisenberg	Germania	1901	1976	Fisica quantistica, Principio di indeterminazione
John Von Neumann	Ungheria / Stati Uniti	1903	1957	Teoria dei giochi, Fisica nucleare, Informatica, …
Alonzo Church	Stati Uniti	1903	1995	Logica, calcolabilità, lambda calcolo
Robert Oppenheimer	Stati Uniti	1904	1967	Fisica nucleare
Kurt Gödel	Austria / Stati Uniti	1906	1978	Logica matematica, Teoremi di incompletezza
Konrad Zuse	Germania	1910	1995	Elettronica, informatica
Alan Turing	Gran Bretagna	1912	1954	Logica, crittografia, informatica, calcolabilità
Claude Shannon	Stati Uniti	1916	2001	Teoria dell'informazione, teoria dei codici, …
Richard Feynman	Stati Uniti	1918	1988	Fisica quantistica
Corrado Böhm	Italia	1923	2017	Logica, informatica, calcolabilità
Giuseppe Jacopini	Italia	1936	2001	Logica, informatica, calcolabilità
Federico Faggin	Italia / Stati Uniti	1941		Elettronica, microprocessori