Cosa vuol dire computare?

Computare significa ridurre un informazione da una forma implicita ad una forma esplicita in modo effettivo

→ RIELABORARE INFORMAZIONI → ESPLICITARLE

DEFINIZIONE DI MODELLO COMPUTAZIONALE

lo possiamo definire come un modello o formalismo matematico, in cui esso viene rappresentato in modo astratto. E rappresentiamo un modo di definire un input, un output, un passo e come organizzare i passi degli elementi di computazione durante un processo effettivo (ALGORITMO) che permette di rendere esplicitato ciò che era implicito.

DIVERSI MODELLI COMPUTAZIONALI

- FUNZIONI RICORSIVE
- LAMBDA CALCOLO
- MACCHINE DI TURING → AUTOMI CON LA MASSIMA POTENZA COMPUTAZIONALE

TEORIA LINGUAGGI FORMALI

GLI ASFD RICONOSCONO LE STRINGHE APPARTENENTI A LINGUAGGI REGOLARI

- non possono memorizzare informazioni a parte la stringa di input

ANALISI DI TURING → concetto di procedura effettiva

Turing poté formalizzare la classe di tutte le procedure effettive (1936)
Questa rappresenta la 1° analisi che descrive come ha luogo la computazione

---

LE CONFIGURAZIONI

(q, x) → STATO, STRINGA

prendiamo 2 configurazioni c_i, c_j indichiamo con $c_i \vdash_A c_j$ la correlazione di esse tramite la relazione di transizione, vale a dire che c_j deriva da c_i per effetto dell'applicazione della funzione di transizione di A

Se andiamo a considerare una sequenza di configurazioni $c_0 \dots c_m$ con lunghezza finita $\Rightarrow c_m \vdash_A c$

terminare → ACCETTAZIONE / RIFIUTO

Quando questa configurazione ha lunghezza finita ed è massimale essa termina

ACCETTAZIONE: situazione in cui l'automa, letto tutta la stringa in input si trova in uno stato q finale ($q \in F$)

RIFIUTO: è una situazione in cui l'automa non ha letto tutta la stringa oppure se l'ha letta non si trova in uno stato finale

DEFINIZIONE

Un automa è detto deterministico se per ogni stringa di input associa una sola computazione e quindi una sola sequenza di configurazioni.

Possiamo dire che un ASFD data una stringa in input può eseguire una sola computazione: se la computazione termina con accettazione (RIFIUTO) allora la stringa viene accettata (RIFIUTATA)

$$\underbrace{C_0 \to C_1 \to C_2 \to}_{\text{1 SOLA COMPUTAZIONE}} \dots$$

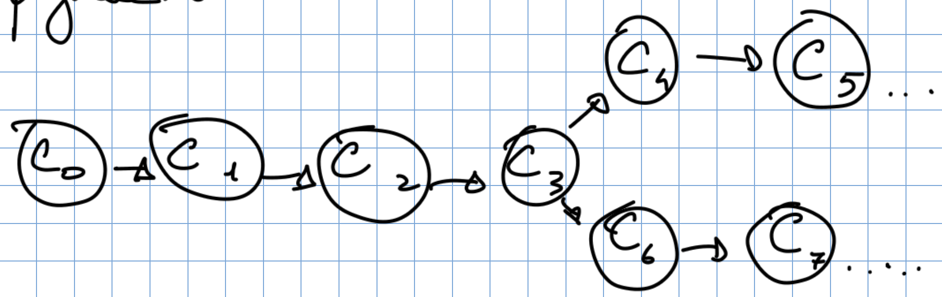
DEFINIZIONE

Un automa è non det. se esso associa ad ogni stringa di input un numero qualunque (>1) di computazioni

Osserviamo che l'automa deterministico è un caso particolare del non det.

$\boxed{1 \text{ COMPUTAZIONE} \rightarrow \text{ASFD} \qquad >1 \rightarrow \text{ASF}}$

INDICHEREMO CON GRADO DI NON DET. DI UN AUTOMA il numero massimo di computazioni che la Δ associa ad una configurazione



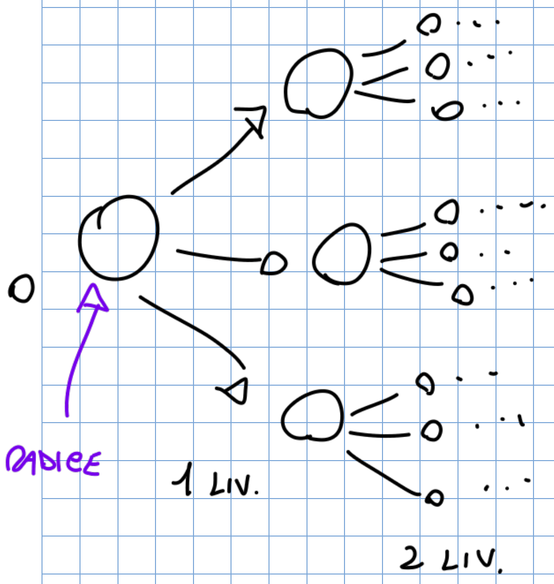
ABBIAMO DIVERSE CONTINUAZIONI DI UNA STESSA COMPUTAZIONE DI UN AUTOMA NON DET.

OSSERVAZIONE

ASIMMETRIA TRA ACCETTAZIONE E RIFIUTO DI UNA STRINGA TRA DET E NON DET

NON DET: la stringa viene accettata se una qualunque delle computazioni definite è di accettazione, mentre non lo è se tutte le possibili computazioni che terminano o non sono di accettazione.

Per un ASF consideriamo come se l'automa esegue una sola computazione non det., per la quale ad ogni passo assume non una sola configurazione ma un insieme di configurazioni transitando ad ogni passo, non solo una conf. a un altra, ma da un insieme di conf. ad un insieme di conf.



ALBERO DI COMPUTAZIONE

La radice dell'albero appartiene a tutte le configurazioni

Con un albero comp. definisco "singole" computazioni aventi come prefissi le sequenze di configurazioni C_0, C_1, C_n corrispondenti ai cammini con origine nella radice dell'albero.

Il concetto di non det. non deve essere confuso con la definizione nel reale

Perché questo concetto è un artificio matematico che ci consente di rappresentare un calcolo, invece di una traiettoria definita in uno spazio di stati come un albero di traiettorie.

Ciò che ci interessa è poter definire un concetto di accettazione legato al fatto che almeno uno dei rami conduce ad uno stato finale.

ESPRESSIONI REGOLARI

Descrivono tutti i linguaggi appartenenti ad una classe

DEF.

Dato un alfabeto Σ e dato l'insieme di simboli $\{+, *, (,), \cdot, \phi\}$ si definisce espressione regolare sull'alfabeto Σ una stringa $r \in \left(\Sigma \cup \{+, *, (,), \cdot, \phi\}\right)^+$ chiusura positiva

tale che valga una delle seguenti condizioni:

1) $r = \phi$

2) $r \in \Sigma$

3) $r = (s + t)$, opp. $r = (s \cdot t)$ oppure $r = s^*$

dove s e t sono espressioni regolari su Σ

corrispondenza tra espressioni regolari e i linguaggi

ESP. REG.	LINGUAGGI
$\emptyset$	Λ
a	$\{a\}$
$(s+t)$	$L(s) \cup L(t)$
$(s \cdot t)$	$L(s) \circ L(t)$
s^*	$(L(s))^*$

con $L(r)$ generico denotato il linguaggio rappresentato dall'espressione regolare r

→ se s e t sono due espressioni regolari posso scrivere (st) anziché $(s \cdot t)$

→ diamo precedenza al simbolo $*$ sul simbolo $\cdot$, ha precedenza su $+$ e possiamo tenere conto delle proprietà associative di tali applicazioni

Es.

l'espressione regolare $(a+(b\cdot(c\cdot d)))$ definita sull'alfabeto $\Sigma=\{a,b,c,d\}$

può essere riscritta $(a+(b(cd)))=a+bcd$

Anche per espressioni regolari, come per i singoli caratteri o stringhe possiamo introdurre la potenza scrivendo:

$(r)^3 \longrightarrow rrr \longrightarrow$ espressione regolare

la chiusura non riflessiva r^+ per indicare $r(r)^*$

OSS.

Per rappresentare la stringa vuota può essere utile a volte usare il simbolo ε nelle espressioni regolari, con lo scopo di indicare il linguaggio $\{\varepsilon\}$

$\Lambda=\{\varepsilon\}$