

Cose suel dicee computerree

Computerree significa reindurre un' informazione da une forme implicite ad une forme esplicite in modo effettivo

→ Rielaborazione informazioni → Esplicatoree

Definizione di modello computazionale

Lo promosso definisce come un modello o formalismo matematico, in cui uno viene rappresentato in modo astratto e se rappresentiamo un modo di definire un input, un output, un flusso e come organizzare i flussi degli elementi di computazione durante un procedimento effettivo (effettivo) che permette di rendere esplicito ciò che era implicito in pertemae

Diversi modelli computazionali

- Funzioni ricorsive
- Lambda-Calcolo
- Macchine di Turing (automi con la norma del pensiero computazionale)

Teorie dei linguaggi: Postulati

Gli AFDI raccomandano le stesse proprietà e linguaggi regolari

- Non formano memoriale informazioni a parte le stringhe di input

Analisi di Alan Turing \Rightarrow concetto di procedure effettive

Turing provò a formalizzare le classi di tutte le procedure effettive (1936) e questo rappresenta la 1^a analisi che descrive come funge la computazione

Le configurazioni

Prendiamo 2 configurazioni C_i e C_j e indichiamo con $C_i \xrightarrow{f} C_j$ la correlazione di S tramite una regola di transizione, vale a dire C_j deriva da C_i per effetto dell'applicazione delle funzioni di transizione di f . ovviamente se l'outcome a cui ci si riferisce è ben definito possiamo scrivere $C_i + C_j$

Se andiamo a considerare una sequenza di configurazioni C_0, \dots, C_n che ha lunghezza finita $\Rightarrow C_n \xrightarrow{\downarrow} \text{configurazione finale}$

termina in uno stato di accettazione o rifiuto

Quando la configurazione ha lunghezza finita ed è nominale sarà terminata

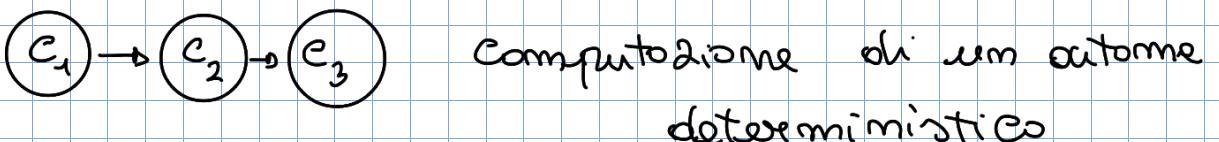
Tipi di configurazione

Accettazione: situazione in cui l'outcome, dette tutte le stringhe di input si trova in uno stato q finale ($q \in F$)

Rifiuto / non accettazione: situazione in cui l'outcome non ha fatto tutte le stringhe di input oppure se è fatto, non si trova in uno stato finale

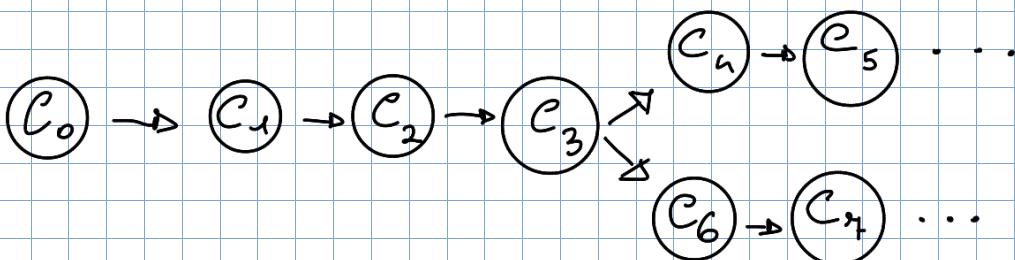
2.5.2 automi deterministici e non

Definizione: Un automa è detto **deterministico** se ogni stringa di input associa una computazione e quindi una simile sequenza di configurazioni. Possiamo dire che un automa deterministico, dato una stringa di input può eseguire una sola computazione: se la computazione termina in una configurazione di accettazione, allora la stessa viene accettata.



Definizione: un automa è non deterministico se uno associa ad ogni stringa di input un numero qualunque ($m > 1$) di computazioni. Osserviamo che l'automa deterministico è un caso particolare del non det per $m = 1$ non det \Leftrightarrow det

Indicheremo con **predo** di non determinismo di un automa il numero minimo di computazione che la funzione di transizione associa ad una configurazione



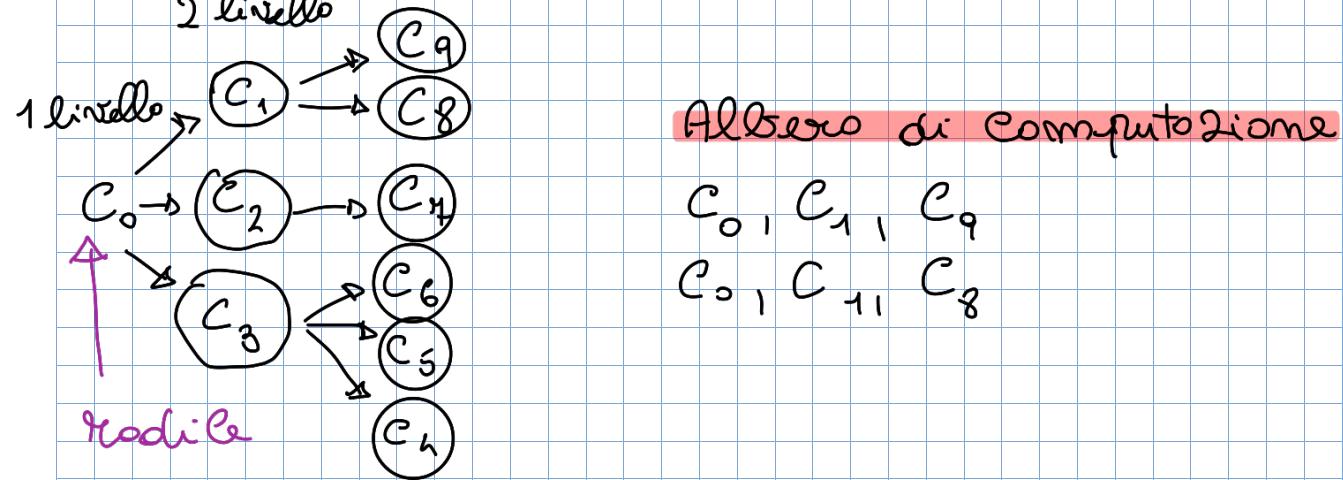
Abbriamo diverse continuazioni di una stessa computazione di un automa non deterministico

Observazione

Asimmetria tra accettazione e rifiuto di una struttura
tra det e non det. Infatti nel caso delle non det
le strutture viene accettate se sono qualsiasi
delle computazioni definite è di accettazione, mentre non
lo è se tutte le plausibili computazioni che terminano
non sono di accettazione.

Per un autome non deterministico, consideriamo come se
l'autome esegue una rete computazione non deterministica
per la quale, ad ogni passo, emette non una rete configura-
zioni ma un insieme di configurazioni, transitando ad
ogni passo, non da una config. a un'altra config ma
da un'insieme di config ad un insieme più config.

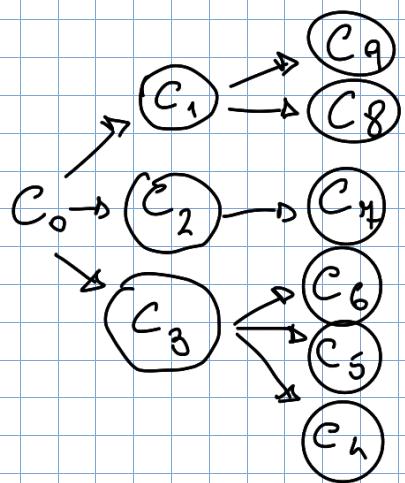
2 livello



Con un albero computazionale definisco "Simple"
Computazioni aventi come prefissi le sequenze di
configurazioni C_0, C_3, C_4 ; C_0, C_{11}, C_8 ; C_0, C_3, C_6
Corrispondenti ai cammini con origine nella
radice dell'albero.

Il concetto di non determinismo non deve essere
confuso con la definizione reale, Perche questo
concetto è sostanzialmente un artificio matematico e ei-

Consente di rappresentare un colpo, visto da una traiettoria definita in uno spazio di stati come un albero di treelettose. Ciò che ci interessa è poter definire un concetto di accettazione legato al fatto che oltremodo uno dei nodi dell'albero conduce ad uno stato finale



Espressioni regolari → descrivono tutti i linguaggi appartenenti a una classe

Dato un alfabeto Σ e dato l'insieme di simboli $\{+, \cdot, *, (,), \cdot, \emptyset\}$

Si definisce espressione regolare sull'alfabeto Σ una stringa

$$\pi \in (\Sigma \cup \{+, \cdot, *, (,), \cdot, \emptyset\})^+$$

tale che voleva una delle seguenti condizioni:

$$1. \pi = \emptyset$$

$$2. \pi \in \Sigma$$

$$3. \pi = (r + t), \text{ oppure } \pi = (r \cdot t) \text{ oppure } \pi = r^*$$

dove r e t sono espressioni regolari sull'alfabeto Σ

Corrispondenze tra le espressioni regolare e i linguaggi:

ESP. REG	LINGUAGGIO
\emptyset	$\{\}$
e	$\{e\}$
$(S + T)$	$L(S) \cup L(T)$
$(S \cdot T)$	$L(S) \circ L(T)$
S^*	$(L(S))^*$

con $L(\pi)$ generico denoto il linguaggio rappresentato dall'espressione regolare π

→ Se s e t sono due espressioni regolari sono
chiuse ($s t$) omliche ($s \cdot t$)

→ Diamo precedenze ai simboli * sul simbolo .,
che precedente su + e poniamo tener conto delle proprietà associativa di tali applicazioni

ES:

l'espressione regolare $(e + (b \cdot (c \cdot d)))$ definite sull'alfabeto $\Sigma = \{0, b, c, d\}$ può essere risolta $(e + (b(c)d)) = e + bcd$

anche per espressioni regolari, come per i simboli correttivi o stringhe poniamo introdurre le parentesi scrivendo $(\pi)^3 \rightarrow \underbrace{\pi \pi \pi}_{\text{espressione regolare}}$

Le chiavi non definiscono π^+ per indicare $\pi(\pi)^*$

Osservazione

Per rappresentare le stringhe vuote può essere utile a volte usare il simbolo ϵ nelle

esprimomi riflessi, con lo scopo di indicare
il linguaggio $\{\epsilon\}$

$$\Delta = \{\epsilon\}$$