Automi, calcolabilità e complessità

Introduzione

Il corso è diviso in tre macroargomenti:

- teoria degli automi → primo passo per capire modelli semplici di computazione
- computabilità → considereremo modelli di calcolo molto più potente, ma esistono problemi che nessun computer sa risolvere (es. stabilire se un programma termina) indipendentemente dalle risorse
- complessità → quantificheremo le risorse (spazio e tempo); in particolare sembrano esistere problemi difficili da risolvere in modo efficiente, però è facile dimostrare la correttezza di una soluzione

N=NP ?

classe dei problemi polinomiali

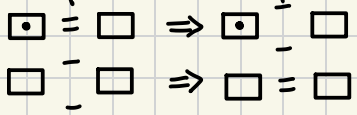
↳ problemi la cui soluzione è verificabile in tempo polinomiale

Linguaggi regolari

Il primo modello esistente è l'automa a stati finiti (DFA). Risulta molto semplice a causa della poca memoria e per il fatto che deve processare i bit in maniera sequenziale

↳ DETERMINISTICO: un input va in solo stato

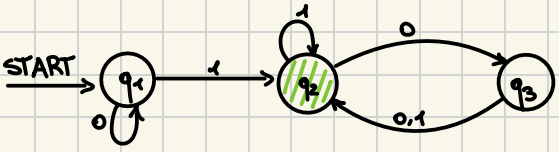
es apertura automatica porte



si può scrivere sotto forma di tabella

Astraendo un DFA è fatto così

$w = 1101$ ⇒ finisce in q_2 allora viene accettato (fosse finito in q_1 o q_3 no)



/// stato di accettazione
○ stati
↷ transizioni

Automa a stati finiti DEF

Un DFA è una tupla $(Q, \Sigma, \delta, q_0, F)$

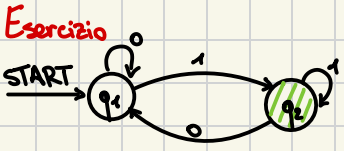
- Q ⇒ insieme finito degli stati
- Σ ⇒ insieme finito dei simboli in input
- δ ⇒ $Q \times \Sigma \to Q$ (le transizioni)
- q_0 ⇒ stato iniziale
- $F \subseteq Q$ ⇒ stati finali (di accettazione)

Inoltre se M è un DFA, l'insieme di stringhe riconosciute da M si denota con $L(M)$ ovvero il **linguaggio riconosciuto** da M (può essere che $L(M) = \emptyset$)

es

il DFA precedente ha linguaggio delle stringhe t.c. w contiene almeno un "1" e # pari di "0" che segue l'ultimo "1"

Esercizio



$L(M)$ = finisce con 1

Per determinare precisamente introduco la **funzione di transizione estesa**

$$\delta^* : Q \times \Sigma^* \to Q \qquad \begin{cases} \delta^*(q, \varepsilon) = \delta(q, \varepsilon) \\ \delta^*(q, ax) = \delta^*(\delta(q, a), x) \end{cases}$$

ε stringa vuota
$x \in \Sigma^*$ qualsiasi stringa di len arbitraria
$a \in \Sigma$

Altro concetto è la **configurazione** è coppia in $Q \times \Sigma^*$:
(1) lo stato
(2) cosa resta da leggere
In particolare dato $x \in \Sigma^*$, la configurazione iniziale è (q_0, x) e ogni passo di computazione porta da una configurazione all'altro rispettando la δ

Relazione binaria

$(p, ax) \vdash_M (q, x)$ sse $\delta(p,a) = q$ dove $p, q \in Q$, $a \in \Sigma$, $x \in \Sigma^*$

↳ relazione su M

La posso estendere considerando la relazione riflessiva e transitiva:

dove $r, p, q \in Q$, $a, b \in \Sigma$, $y \in \Sigma^*$

① $(q, x) \vdash_M^* (q, x)$

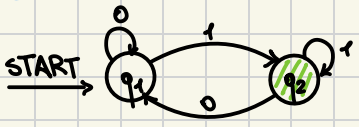
② $(q, aby) \vdash_M (p, by)$, $(p, by) \vdash_M (r, y)$ $\Rightarrow (q, aby) \vdash_M^* (r, y)$

Linguaggio accettato DEF

Diciamo che $x \in \Sigma^*$ è accettato da $M = (Q, \Sigma, \delta, q_0, F)$ se $\delta^*(q_0, x) \in F$
oppure $(q_0, x) \vdash_M^* (q, \varepsilon)$ $q \in F$

In altre parole $L(M) = \{x \in \Sigma^* : \delta^*(q_0, x) \in F\}$

es



$(q_1, 011) \vdash_M (q_1, 11) \vdash_M (q_2, 1) \vdash_M (q_2, \varepsilon)$

$\Downarrow$

$w = 011 \in L(M)$

Linguaggi regolari DEF

$REG=\{L\subseteq\Sigma^*: \exists \text{ DFA } H \text{ t.c. } L(H)=L\}$ → stringhe accettate Ovvero se esiste un automa che lo riconosce

Vogliamo capire come progettare DFA per un dato linguaggio

es
$L=\{x\in\{0,1\}^*: x=1y, y\in\{0,1\}^*\}$ → stringhe che iniziano per 1

Prove di correttezza

DFA accetta $x \leftrightarrow x\in L$

es prova di correttezza automa precedente

oss $\delta^*(q_1,v)=q_1 \quad \forall v\in\{0,1\}^*$
$\delta^*(q_2,v)=q_2 \quad \forall v\in\{0,1\}^*$

per induzione dimostriamo che $x\in L \leftrightarrow$ DFA accetta x

base: $|x|=0$ se $x=\varepsilon$, $\delta^*(q_0,\varepsilon)=\delta(q_0,\varepsilon)=q_0\notin F$

induttivo: sia $n>0$, supponendo che $|w|\leq n$

$$\delta^*(q_0,w)=\begin{cases}q_1 & \text{se } w \text{ inizia con } 1\\ q_2 & \text{se } w \text{ inizia con } 0\end{cases}$$

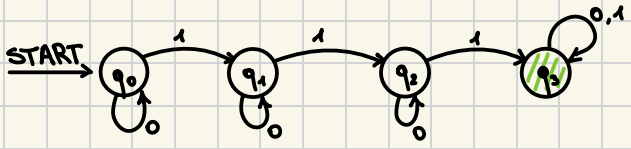
prendo x t.c. $|x|=n+1$ e lo penso $x=av$ con $a\in\{0,1\}$, $v\in\{0,1\}^*$

$\delta^*(q_0,x)=\delta^*(q_0,av)=\delta^*(\delta(q_0,a),v)=$

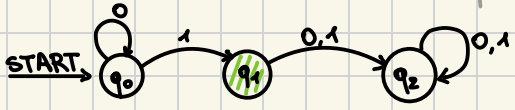
$\delta(q_0,a)=q_2$ se $a=0$
$\delta(q_0,a)=q_1$ se $a=1$
$\Rightarrow \delta^*(q_0,x)=q_1 \leftrightarrow a=1$

Esercizi

$L=\{x\in\{0,1\}^*: w_H(x)\geq 3\}$ dove $w_H(x)=\#1$ con $x\in\{0,1\}^*$



$L=\{x\in\{0,1\}^*: x=0^n1 \text{ con } n\in\mathbb{N}\}$ → qualsiasi # di 0 seguiti da un 1



Operazioni sui linguaggi

Fissiamo $\Sigma=\{0,1\}$. Per $n\in\mathbb{N}$ $[n]=\{1,2,\dots,n\}$

Siccome i linguaggi sono insiemi di stringhe posso considerarne su di essi operazioni

- UNIONE: $L_1\cup L_2=\{x\in\Sigma^*: x\in L_1 \vee x\in L_2\}$
- INTERSEZIONE: $L_1\cap L_2=\{x\in\Sigma^*: x\in L_1\wedge x\in L_2\}$
- COMPLEMENTO: $\overline{L}=\{x\in\Sigma^*: x\notin L\}$
- CONCATENAZIONE:

stringhe: $x=a_1,\dots,a_n \quad y=b_1,\dots,b_m \quad m,n>0$

$xy=a_1,\dots,a_n,b_1,\dots,b_m\in\Sigma^*$

$\varepsilon x=x\varepsilon=x$

$$\Rightarrow \begin{cases}x\varepsilon=x\\ x(ya)=(xy)a\\ x,y\in\Sigma^*, a\in\Sigma\end{cases}$$

linguaggio: $L_1\circ L_2=\{xy: x\in L_1\wedge y\in L_2\}$

es

$\Sigma = \{a, b\}$

$L_1 = \{a, ab, ba\}$ $\quad L_1 \circ L_2 = \{aaab, ab, abab, abb, baab, bab\}$

$L_2 = \{ab, b\}$

- POTENZA

stringhe

$x^n = x \dots x$ n volte con $x \in \Sigma^*$

$\begin{cases} x^0 = \varepsilon \\ x^{n+1} = x^n x \end{cases}$

linguaggi

$\begin{cases} L^0 = \{\varepsilon\} \\ L^{n+1} = L^n \circ L \end{cases}$

oss $L^* = \left(\bigcup_{n=0}^{\infty} L^n\right) = \{\varepsilon\} \cup L^1 \cup L^2 \cup \dots \cup \dots$ $\quad L = \{a, b\}$, $L^* = \{\varepsilon, a, b, aa, ab, ba, \dots\}$

es

$L = \{a, ab, ba\}$

$L^2 = \{aa, aab, aba, abab, abba, baa, baab, baba\}$

Domanda

Vogliamo studiare le proprietà di chiusura dei linguaggi regolari, ovvero: se $L_1, L_2 \in REG$ posso dire che $L_1 \cup L_2 \in REG$? $L_1 \cap L_2 \in REG$? $\overline{L_1} \in REG$? ...

TEOREMA

REG è chiusa per UNIONE

intuizione $L_1, L_2 \in REG \Rightarrow \exists M_1, M_2 \in DFA$ t.c. $L(M_1) = L_1$, $L(M_2) = L_2$

devo definire M t.c. $L(M) = L_1 \cup L_2$

problema dato x candidato non posso prima provare a vedere se $M_1(x)$ accetta (→ x viene consumata)

idea M deve eseguire M_1, M_2 in parallelo e accettare sse uno dei due accetta

dim

siano $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$ t.c. $L(M_1) = L_1$

$M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$ t.c. $L(M_2) = L_2$

devo costruire $M = (Q, \Sigma, \delta, q_0, F)$ t.c. $L(M) = L_1 \cup L_2$, dove:

- $Q = \{(r_1, r_2) : r_1 \in Q_1, r_2 \in Q_2\} = Q_1 \times Q_2$
- $\delta : Q \times \Sigma \to Q$

  $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta(r_2, a))$
- $F = \{(r_1, r_2) : r_1 \in F_1 \vee r_2 \in F_2\} = (F_1 \times Q_2) \cup (F_2 \times Q_1)$

Abbiamo dunque dimostrato questo teorema per costruzione, secondo cui la dimostrazione consiste nel costruire l'oggetto desiderato

A questo punto rimarrebbe solo di dimostrare la correttezza del DFA (evitabile), procediamo.

Dobbiamo dimostrare che $\forall x \in \Sigma^* \quad x \in L(M) \iff x \in (L_1 \cup L_2)$

→

se $x \in L(M)$, allora $\delta^*(q_0, x) \in F$ dove $F = (F_1 \times Q_2) \cup (Q_1 \times F_2)$

$\delta^*(q_0, x) = (p, q)$

$p = \delta^*(q_0^1, x)$

$q = \delta^*(q_0^2, x)$

$\Rightarrow p \in F_1 \vee q \in F_2 \Rightarrow x \in L_1 \vee x \in L_2$

←

se $x \in L_1$, allora $\delta^*(q_0^1, x) \in F_1$

$\delta^*(q_0, x) = (\delta_1^*(q_0^1, x), \delta_2^*(q_0^2, x)) \in F_1 \times Q_2 \subseteq F \Rightarrow M$ accetta x

stessa cosa vale per $x \in L_2$

■

TEOREMA

REG è chiusa per INTERSEZIONE

dim

siano $M_1 = (Q_1, \Sigma, \delta_1, q_0^1, F_1)$ t.c. $L(M_1) = L_1$
$M_2 = (Q_2, \Sigma, \delta_2, q_0^2, F_2)$ t.c. $L(M_2) = L_2$

devo costruire $M = (Q, \Sigma, \delta, q_0, F)$ t.c. $L(M) = L_1 \cap L_2$, dove:
- $Q = \{(r_1, r_2) : r_1 \in Q_1, r_2 \in Q_2\} = Q_1 \times Q_2$
- $\delta : Q \times \Sigma \to Q$
  $\delta((r_1, r_2), a) = (\delta_1(r_1, a), \delta(r_2, a))$
- $F = \{(r_1, r_2) : r_1 \in F_1 \wedge r_2 \in F_2\} = (F_1 \times F_2)$

TEOREMA

REG è chiusa per COMPLEMENTO

dim

sia $M_1(Q, \Sigma, \delta, q_0, F_1)$ t.c. $L_1 = L(M)_1$
devo costruire $M(Q, \Sigma, \delta, q_0, F)$ t.c. $L = \Sigma^* / L_1$, dove:
- $F = Q / F_1$

Per dimostrare che REG è chiusa rispetto alla concatenazione bisogna introdurre il non determinismo poiché non 01/10 siano in grado di spezzare x in $x_1 \circ x_2$ tramite i DFA

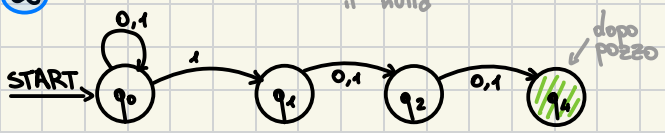
Non determinismo

In un automa non deterministico, a differenza di un DFA, possono esistere diverse scelte per lo stato successivo in ogni punto
Si differenziano dai DFA dunque per:
- si possono avere più stati successivi
- sono ammessi gli "ε-archi", che consentono di aprire rami di computazione parallelli senza leggere nulla
- accetto sse esiste almeno un ramo che accetta

NFA DEF

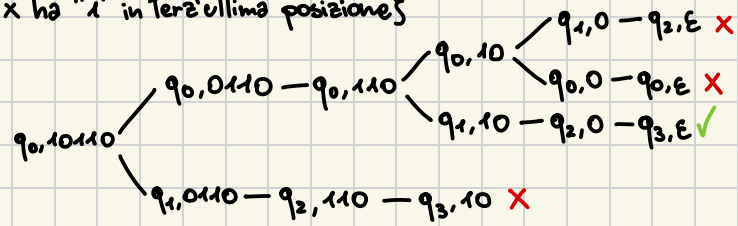
Un NFA è (Q, Σ, q_0, F) dove Q, Σ, q_0, F sono come nei DFA mentre invece
$\delta: Q \times \Sigma_\varepsilon \to \mathcal{P}(Q)$ ← insieme delle parti
dove $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$
↑ posso leggere il "nulla"

es

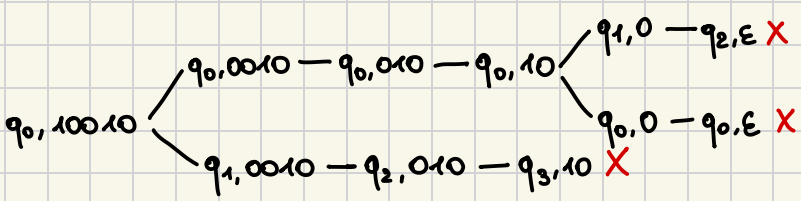


$L = \{x \in \{0,1\}^* : x \text{ ha "1" in terz'ultima posizione}\}$

$x = 100$ ✓
$x = 1000$ ✗
$x = 10110$

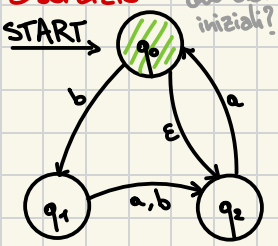


$x = 10010$



Esercizio

due stati iniziali?



Verificare computazione per:
- $w = \varepsilon$
- $w = a$
- $w = bb$
- $w = babba$

$w = \varepsilon$ ✓ q_0, ε ✓ q_2, ε ✗

$w = a$ ✓ q_0, a ✗ $q_2, a — q_0, \varepsilon$ ✓

$w = bb$ ✗ $q_0, bb — q_1, b — q_2, \varepsilon$ ✗ q_2, bb ✗

$w = babba$ ✗ $q_0, babba — q_1, abba — q_2, bba$ ✗ $q_2, babba$ ✗

TEOREMA

Per ogni NFA esiste un DFA
idea trasformare l'NFA in un DFA equivalente
dim
sia $N = (Q, \Sigma, \delta, q_0, F)$ l'NFA t.c. $L(N) = A$
costruiamo un DFA $M = (Q', \Sigma, \delta', q_0', F')$ t.c. $L(M) = A$

prima di fare la costruzione completa, consideriamo inizialmente il caso più semplice in cui N non ha ε-archi (in seguito li considereremo)

M' avrà:
- $Q' = \mathcal{P}(Q)$ → ogni stato di M è un insieme di stati di N
- $\delta'(R,a) = \{q \in Q : q \in \delta(r,a) \text{ dove } r \in R\}$ → quando M legge un simbolo a nello stato R, mostra dove a porta ogni stato in R
  dove $R \in Q'$ e $a \in \Sigma$
  ricordiamo $\delta' : \mathcal{P}(Q) \times \Sigma \to \mathcal{P}(Q)$
  ↳ $\bigcup_{r \in R} \delta(r,a)$

- $q_0' = \{q_0\} \in \mathcal{P}(Q)$
- $F' = \{R \in Q' \mid R \cap F \neq \emptyset\}$

Ora consideriamo gli ε-archi, introducendo notazione. Per ogni stato R di M, definiamo E(R) come la collezione di stati che possono essere raggiunti dagli elementi di R usando solo gli ε-archi, includendo gli stessi elementi di R

$E(R) = \{q \mid q \text{ raggiunto da } R \text{ tramite 0 o più } \varepsilon\text{-archi}\}$

Utilizziamo quindi E(R) in $\delta'(R,a)$
$\delta'(R,a) = \{q \in Q : q \in E(\delta(r,a)) \text{ dove } r \in R\}$
↳ oltre a mettere quelli raggiungibili tramite a metto anche quelli tramite ε-archi

Infine dobbiamo aggiungere gli stati raggiungibili tramite ε-archi dallo stato iniziale, quindi
$q_0' = E(\{q_0\})$