SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Automi a stati finiti

Prof. Daniele Gorla

Un formalismo per reti sequenziali

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

I FF sono le reti sequenziali più semplici, ma già esibiscono le caratteristiche fondamentali di tali reti:

- memorizzazione di valori booleani (*stato*)
- modifica dei valori memorizzati in base ai segnali in ingresso (*transizioni di stato*)

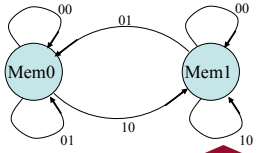
Così come le TV erano il formalismo per rappresentare reti combinatorie, vorremmo un formalismo per rappresentare reti sequenziali.

Per descrivere il comportamento dei FF abbiamo usato delle TV "estese", in cui abbiamo introdotto il fattore tempo (y e Y); tale formalismo può essere reso più intuitivo rappresentandolo in maniera grafica.

s	r	y	Y
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	0
1	0	0	1
1	0	1	1

Uno stato per ogni valore memorizzabile

Per ogni riga della tabella ho una freccia dallo stato associato al valore di y a quello di Y etichettato con la sequenza di valori in ingresso



Sistemi di Transizioni Etichettate

SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Un *Sistema di Transizioni Etichettate* (*LTS*, dall'inglese Labelled Transition System) è una quadrupla (Q,Σ,q_0,δ) dove:

- Q è l'insieme degli *stati*;
- Σ è l'*alfabeto* (di input);
- $q_0 \in Q$ è lo *stato iniziale*;
- $\delta: Q\times\Sigma \to Q$ è la *funzione di transizione*.

N.B.: δ è una funzione; quindi, ad ogni coppia $(q,a) \in (Q\times\Sigma)$ corrisponde uno ed un solo stato raggiunto dall'automa!!

Nell'esempio di prima abbiamo

- Q = {Mem0, Mem1};
- Σ = {00,01,10};
- δ: (Mem0,00) → Mem0 (Mem1,00) → Mem1
  (Mem0,01) → Mem0 (Mem1,01) → Mem0
  (Mem0,10) → Mem1 (Mem1,10) → Mem1

E q_0? Dipende dal valore iniziale nel FF (tipicamente si assume 0)

Automi a stati finiti

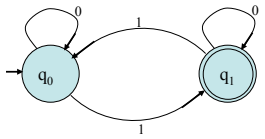
SAPIENZA
UNIVERSITÀ DI ROMA
DIPARTIMENTO DI INFORMATICA

Se alla definizione di LTS aggiungo anche un insieme $F \subset Q$ di *stati finali* (o *di accettazione*) e impongo il vincolo che tutti gli insiemi considerati siano finiti, ottengo un *automa a stati finiti*, che è quindi una quintupla $A = (Q,\Sigma,q_0,\delta,F)$.

Graficamente, gli stati finali si indicano con un cerchio doppio, mentre lo stato iniziale con una freccetta entrante.

Gli stati finali servono per indicare quali sequenze di input sono accettate dall'automa: sono tutte quelle che portano l'automa dallo stato iniziale a uno qualsiasi stato finale. Formalmente, A accetta $a_1 \ldots a_n$ (per $a_i \in \Sigma$) se esiste una n-pla di stati $q_1,\ldots,q_n$ t.c. $q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \ldots \xrightarrow{a_n} q_n \in F$ (cioè, $\delta(q_i,a_{i+1}) = q_{i+1}$ per ogni i e $q_n \in F$).

Es.:



Accetta 1, 01, 001, 01101, …
Rifiuta 0, 11, 011, 0110, …
In generale, accetta tutte e sole le sequenze di input con un numero dispari di 1.

Automi con output

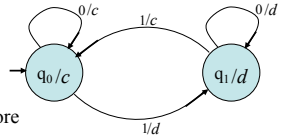
Se invece aggiungo alla definizione di LTS un *alfabeto di output* Δ, una *funzione di output* λ e impongo il vincolo che tutti gli insiemi considerati siano finiti, ottengo un *automa con output*, che è quindi una sestupla $M = (Q,\Sigma,\Delta,q_0,\delta,\lambda)$.

Bisogna definire la funzione di output. L'output può naturalmente essere associato agli stati o alle transizioni; questo genera due modelli:

- Automi *di Moore*, in cui $\lambda : Q \rightarrow \Delta$
- Automi *di Mealy*, in cui $\lambda : Q \times \Sigma \rightarrow \Delta$

Graficamente, gli output si indicano scrivendo "/ *b*" (per $b \in \Delta$) dopo il nome dello stato (nel modello di Moore) o dopo il carattere di input (nel modello di Mealy).

Es.:



Modello di Moore

Modello di Mealy

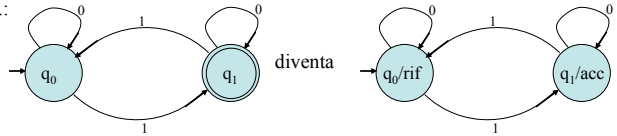
Automi accettori tramite automi con output

È facile convincersi che gli automi finiti (accettori di sequenze di input) possono essere resi da automi con output (mentre il viceversa non vale!!).

Possiamo dire che un automa di Moore con $\Delta = \{acc, rif\}$ accetta una sequenza $a_1 \ldots a_n$ (per $a_i \in \Sigma$) se esiste una n-pla di stati $q_1,\ldots,q_n$ t.c.

$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \ldots \xrightarrow{a_n} q_n$ e $\lambda(q_n) = acc$.

Es.:



N.B.: lo stesso si può fare con un automa di Mealy, etichettando tutte le transizioni entranti in uno stato finale con "acc" e tutte le altre con "rif".

Quindi, d'ora in poi considereremo solo automi con output e specificheremo solo il modello usato.

Esempio: distributore di bibite

Usiamo un automa per modellare un sistema reale:

- Un distributore di bibite che costano 30c;
- Il distributore accetta in input monete da 10c e da 20c;
- Il distributore dà in output la bibita se sono stati inseriti almeno 30c e non dà resto (ma mantiene il resto per l'acquisto della bibita successiva).

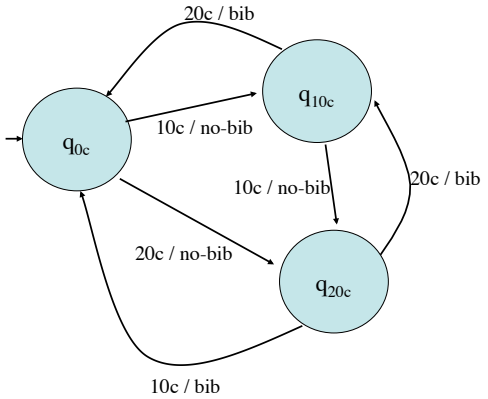
Il passo fondamentale per definire un automa è capire cosa serve memorizzare per il eseguire le specifiche richieste, cioè definire gli stati e il loro significato.

In questo esempio bisogna tener traccia dell'importo ricevuto fino a quel momento. Quanti stati diversi?

0c, 10c, 20c → 3 stati

N.B.: non serve avere uno stato associato a 30c o a 40c perchè in questi casi il distributore emette la bibita e torna allo stato 0c o 10c, rispettivamente.

Distributore di bibite (rappresentaz. grafica dell'automa)



Distributore di bibite (rappr. matematica dell'automa)

SAPIENZA Università di Roma – Dipartimento di Informatica

$Q = \{q_{0c}, q_{10c}, q_{20c}\}$
$\Sigma = \{10c, 20c\}$
$\Delta = \{no\text{-}bib, bib\}$
Stato iniziale: q_{0c}

δ: $(q_{0c},10c) \rightarrow q_{10c}$
$(q_{0c},20c) \rightarrow q_{20c}$
$(q_{10c},10c) \rightarrow q_{20c}$
$(q_{10c},20c) \rightarrow q_{0c}$
$(q_{20c},10c) \rightarrow q_{0c}$
$(q_{20c},20c) \rightarrow q_{10c}$

λ: $(q_{0c},10c) \rightarrow$ no-bib
$(q_{0c},20c) \rightarrow$ no-bib
$(q_{10c},10c) \rightarrow$ no-bib
$(q_{10c},20c) \rightarrow$ bib
$(q_{20c},10c) \rightarrow$ bib
$(q_{20c},20c) \rightarrow$ bib

Per semplicità, tutte queste informazioni si possono scrivere in maniera più compatta nella cosiddetta *rappresentazione tabellare*:

	10c	20c
→ q_{0c}	q_{10c} / no-bib	q_{20c} / no-bib
q_{10c}	q_{20c} / no-bib	q_{0c} / bib
q_{20c}	q_{0c} / bib	q_{10c} / bib

Distributore di bibite evoluto

SAPIENZA Università di Roma – Dipartimento di Informatica

Possiamo rendere l'esempio di prima più realistico dando all'utente la possibilità di scegliere la bibita che vuole (per es., coca o fanta)

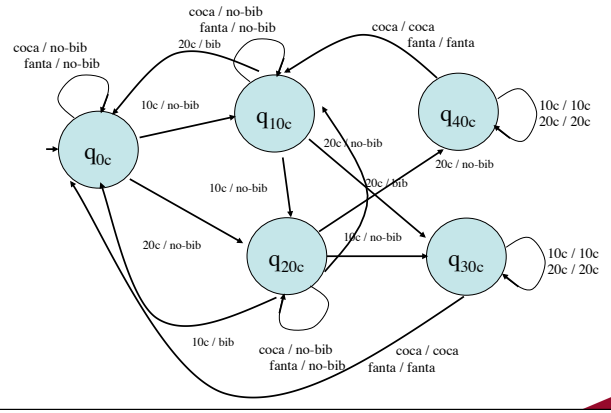
Questa nuova specifica modifica sostanzialmente l'automa. Infatti, non bastano più 3 stati, ma ne servono 5:

→ prima si erogava la bibita appena introdotto un importo sufficiente
→ ora bisogna aspettare la specifica di quale bibita (nuovi stati: 30c e 40c)
→ inoltre, in tali nuovi stati, se l'utente continua ad inserire monete bisogna restituirgliele, restando sempre nello stesso stato
→ invece nei vecchi stati bisogna ignorare l'input se questo è la selezione di una bibita

N.B.: devo considerare tutti gli input in tutti gli stati, poiché δ e λ sono funzioni

Automa per il distributore di bibite evoluto

SAPIENZA Università di Roma – Dipartimento di Informatica

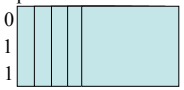


Esempio: automa per la somma di naturali

SAPIENZA Università di Roma – Dipartimento di Informatica

IN: due sequenze di bit, $A = a_0\, a_1\, a_2 \ldots$ e $B = b_0\, b_1\, b_2 \ldots$
OUT: una sequenza di bit $s_0\, s_1\, s_2 \ldots$ t.c., per ogni i, $s_i \ldots s_0 = a_i \ldots a_0 + b_i \ldots b_0$, ignorando un eventuale riporto finale.

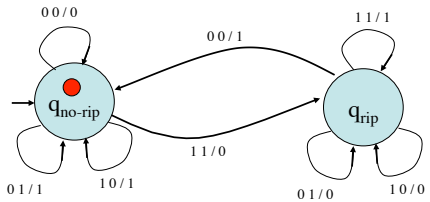
Esempio:
A: 0
B: 1
output: 1



SOLUZIONE:

- L'automa riceve i bit di input dal meno significativo e similmente produce i bit in output
- Questo ci ricorda il procedimento per fare la somma (visto sia nella lezione sui naturali che in quella in cui abbiamo sviluppato l'addizionatore combinatorio)
- L'unica informazione che ci serve passando dal bit i-esimo al bit $(i+1)$-esimo è sapere se al passo i c'è stato un riporto o meno
  → questa è la semantica degli stati, che saranno pertanto solo 2

Automa per la somma



Verifichiamo che l'automa si comporti come richiesto dall'esempio dato:

A: 0 1 1 0 0 1 0
B: 1 0 1 0 1 1 1
output: 1 1 0 1 1 0 0

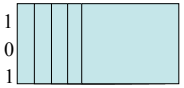
Esempio: automa per il resto modulo 4

IN: una sequenza di bit, B = $b_0\, b_1\, b_2 \ldots$

OUT: una sequenza di coppie bit $r_0\, r'_0\, r_1\, r'_1 \ldots$ t.c., per ogni i,

$r_i\, r'_i = b_0 \ldots b_i$ MOD 4.

Esempio: B: 1
output: 0
1

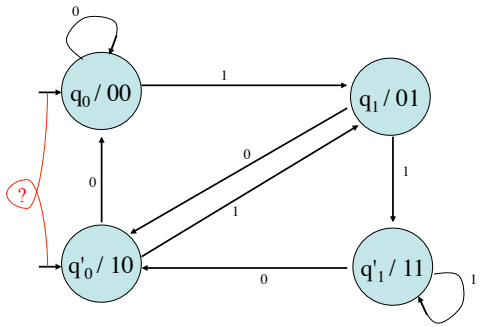


SOLUZIONE:

- Questa volta l'automa riceve i bit di input dal più significativo (mette i nuovi bit in coda)
- Ricordiamo il teorema del resto, secondo cui il resto di una divisione di un naturale in base 2 per la potenza k-esima di 2 è dato dai k bit meno significativi del numero in questione.
- L'unica informazione che ci serve all'arrivo del bit i-esimo è il valore del bit $(i–1)$-esimo, tranne per il primo bit (per cui nessuna informazione è richiesta)
  - → avremo quindi solo 2 stati
  - → lo stato iniziale è quello associato al bit 0, tanto il naturale rappresentato dal bit b ha lo stesso valore di $0b$

Automa per i resti modulo 4

Questa volta usiamo il modello di Moore:



Si può scegliere come stato iniziale q_0 o q'_0, tanto nel modello di Moore il primo output va sempre ignorato (è un output di default che viene sempre prodotto, indipendentemente dall'input)