nguaggi Formali e Traduttori

2.1 Automi a stati finiti deterministici (DFA)

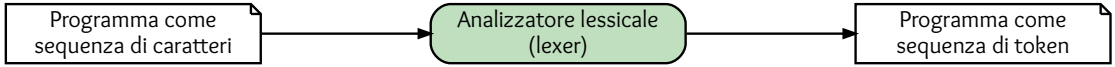
- Analisi lessicale
- Descrivere token (di Java)
- Esempio
- Automa a stati finiti
- Un semplice esempio di riconoscimento
- Un semplice esempio di riconoscimento
- La soluzione come automa a stati finiti
- Automi a stati finiti
- Linguaggio riconosciuto da un DFA
- Tabelle di transizione
- Esempio: numeri binari pari
- Esempio: esiste a seguita da bb
- Esempio: ogni a è seguita da bb
- Esempio: numeri binari multipli di 5
- Esercizi sulla definizione di DFA
- Esercizi sulla comprensione di DFA

Analisi lessicale

Scopo dell'analisi lessicale

Riconoscere **sequenze di caratteri** che rappresentano elementi atomici del programma

Tali sequenze di caratteri sono dette **token** o **lessemi**



Esempi di token

- costanti (42, `1.5`, `true`, `"questa è una stringa"`, ...)
- identificatori (`i`, `metodo`, `String`, ...)
- parole chiave (`class`, `public`, `for`, `if`, ...)
- operatori (<=, +, ==, ...)
- simboli di punteggiatura (`,`, `;`, `(`, `)`, `{`, `}`, ...)
- ...

Descrivere token (di Java)

Costante numerica intera

Una sequenza non vuota di cifre decimali, eventualmente preceduta da + o -

- `0`
- `-42`

Costante numerica con virgola

Due sequenze (di cui almeno una non vuota) di cifre decimali separate da `.`

- `0.5`
- `.5`

Identificatore

Una sequenza non vuota di lettere, numeri e _ che non inizia con un numero e che contiene almeno un carattere diverso da _

- `_i`
- `metodo`

Esempio

Sequenza di caratteri

```
public static int metodo(int n) {
    int r = 1;
    for (int i = 1; i <= n; i++)
        r = r * i;
    return r;
}
```

Sequenza di token

- parola chiave `public`
- parola chiave `static`
- identificatore `int`
- identificatore `metodo`
- parentesi aperta `(`
- ...

Nota

- alcune sequenze di caratteri (spazi/commenti) vengono **scartate** dal lexer, ma possono essere fondamentali per **separare token adiacenti** (es. `int` e `metodo`)

Automa a stati finiti

Che cos'è

- automa = macchina che **riconosce** stringhe
- stati finiti = con **memoria finita** (l'automa ricorda "poche" cose della stringa)
- input dell'automa = una stringa
- output dell'automa = **sì** se la stringa è riconosciuta, **no** altrimenti

Come funziona

- l'automa legge la stringa **un simbolo alla volta**, da sinistra verso destra

  ⇒ l'automa ha una visione locale e limitata

- ogni simbolo letto altera lo stato dell'automa

  ⇒ l'automa "ricorda" le caratteristiche della stringa letta usando lo stato

- quando la stringa è stata letta interamente, l'automa risponde "sì" se si trova in un cosiddetto **stato finale** e "no" altrimenti

Un semplice esempio di riconoscimento

Problema

Sono in una stanza con un recipiente contenente un numero imprecisato (ma all'apparenza molto grande) di biglie. A parte il recipiente, nella stanza c'è solo un interruttore che accende e spegne una lampada. Ho poca memoria. Ho il compito di svuotare il recipiente e dire "sì" se il numero di biglie è dispari, "no" altrimenti.

Soluzione (difettosa)

- Uso un **contatore** per ricordare il numero di biglie, inizialmente posto a 0
- Estraggo le biglie una alla volta, a ogni estrazione incremento il contatore di 1
- Quando il recipiente è vuoto, dico "sì" se il valore del contatore è dispari, "no" altrimenti

Un semplice esempio di riconoscimento

Problema

Sono in una stanza con un recipiente contenente un numero imprecisato (ma all'apparenza molto grande) di biglie. A parte il recipiente, nella stanza c'è solo un interruttore che accende e spegne una lampada. Ho poca memoria. Ho il compito di svuotare il recipiente e dire "sì" se il numero di biglie è dispari, "no" altrimenti.

Soluzione (difettosa)

- Uso un **contatore** per ricordare il numero di biglie, inizialmente posto a 0
- Estraggo le biglie una alla volta, a ogni estrazione incremento il contatore di 1
- Quando il recipiente è vuoto, dico "sì" se il valore del contatore è dispari, "no" altrimenti

Cosa non va in questa soluzione

- La quantità di informazione da ricordare dipende dal numero di biglie nel recipiente
- Il contatore può diventare arbitrariamente grande (= non è "a stati finiti")

Un semplice esempio di riconoscimento

Problema

Sono in una stanza con un recipiente contenente un numero imprecisato (ma all'apparenza molto grande) di biglie. A parte il recipiente, nella stanza c'è solo un interruttore che accende e spegne una lampada. Ho poca memoria. Ho il compito di svuotare il recipiente e dire "sì" se il numero di biglie è dispari, "no" altrimenti.

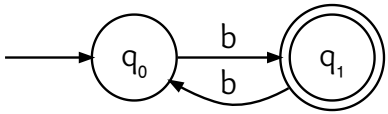
Osservazione

- I numeri pari e dispari si alternano

Soluzione

- Uso la lampada per ricordare se il numero di biglie estratte è pari (lampada spenta) o dispari (lampada accesa)
- Mi assicuro che la lampada sia inizialmente spenta (inizialmente ho estratto 0 biglie e 0 è un numero pari)
- Estraggo le biglie una alla volta, a ogni estrazione premo l'interruttore
- Quando il recipiente è vuoto, dico "sì" se la lampada è accesa e "no" altrimenti

La soluzione come automa a stati finiti



- ogni cerchio rappresenta uno **stato** dell'automa (q_0 = lampada spenta, q_1 = lampada accesa)
- gli archi etichettati "b" rappresentano le **transizioni di stato**, ovvero come cambia lo stato dell'automa (e della lampada) a ogni estrazione di biglia
- la freccia entrante in q_0 indica che q_0 è lo stato **iniziale** (all'inizio la lampada è spenta)
- il doppio cerchio attorno a q_1 indica che q_1 è lo stato **finale** o **di accettazione** (se l'automa si trova in questo stato quando le biglie sono finite, allora il loro numero era dispari)

Note

- in generale, possono esserci zero o più stati finali
- in generale, transizioni diverse possono essere etichettate con simboli diversi

Automi a stati finiti

Definizione

Un **automa a stati finiti** (detto anche **DFA**, da **D**eterministic **F**inite-state **A**utomaton) è una quintupla $A = (Q, \Sigma, \delta, q_0, F)$ dove:

- Q è un insieme **finito** di **stati**
- Σ è l'**alfabeto** riconosciuto dall'automa
- $\delta : Q \times \Sigma \to Q$ è la **funzione di transizione**
- $q_0 \in Q$ è lo **stato iniziale**
- $F \subseteq Q$ è l'insieme di **stati finali**

Esempio

Per l'automa della slide 8 abbiamo:

- $Q = \{q_0, q_1\}$
- $\Sigma = \{b\}$
- $\delta = \{((q_0, b), q_1), ((q_1, b), q_0)\}$
- $F = \{q_1\}$

Linguaggio riconosciuto da un DFA

Definizione

La **funzione di transizione estesa** dell'automa $A = (Q, \Sigma, \delta, q_0, F)$ è la funzione $\hat{\delta} : Q \times \Sigma^* \to Q$ definita per induzione sul suo secondo argomento come segue:

$$\hat{\delta}(q, \varepsilon) = q \qquad \hat{\delta}(q, wa) = \delta(\hat{\delta}(q, w), a)$$

Definizione

Il **linguaggio riconosciuto** (o **accettato**) dall'automa $A = (Q, \Sigma, \delta, q_0, F)$ è denotato da $L(A)$ e definito come segue:

$$L(A) \stackrel{\text{def}}{=} \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \in F\}$$

Esempio

Per l'automa in slide 8 abbiamo $L(A) = \{b^{2n+1} \mid n \in \mathbb{N}\}$

Definizione

Un linguaggio L si dice **regolare** se esiste un automa A tale che $L = L(A)$.

Tabelle di transizione

Un DFA si può rappresentare comodamente anche in forma tabellare. Per esempio, per l'automa visto in slide 8 avremo:

Stato	b
$\rightarrow q_0$	q_1
$* q_1$	q_0

- Le **righe** corrispondono agli **stati** dell'automa
- Le **colonne** corrispondono ai **simboli** dell'alfabeto dell'automa
- Lo **stato iniziale** è marcato con $\rightarrow$
- Ogni **stato finale** è marcato con *
- La **cella** corrispondente alla riga q e alla colonna a contiene $\delta(q, a)$

Esempio: numeri binari pari

Progettare un automa che riconosce le stringhe di bit che rappresentano **numeri pari**

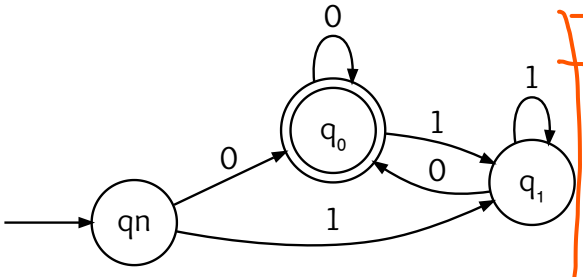
Osservazione

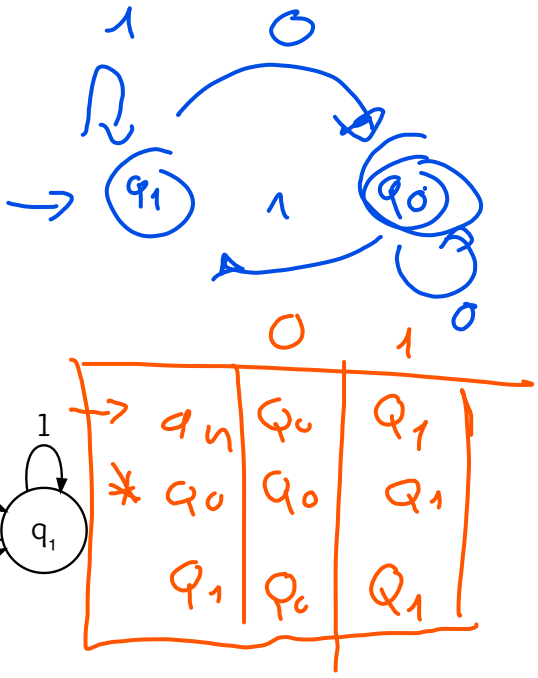
- le stringhe devono essere **non vuote** e **terminare con 0**

Stati

- q_n = non ho ancora riconosciuto alcun bit
- q_0 = ho riconosciuto almeno un bit, l'ultimo era 0
- q_1 = ho riconosciuto almeno un bit, l'ultimo era 1

Soluzione





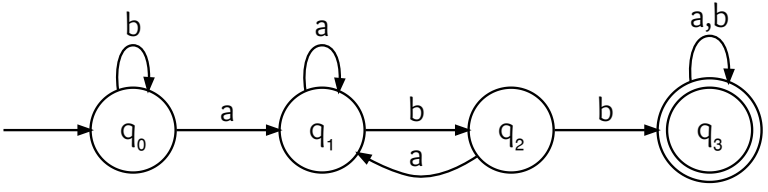
Esempio: esiste a seguita da bb

Definire un automa che riconosce le stringhe di a e b in cui **esiste** una a seguita da bb

Stati

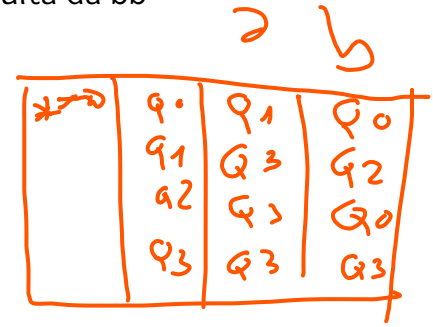
- q_0 = tutti i simboli riconosciuti fino ad ora erano b
- q_1 = l'ultimo simbolo riconosciuto era una a
- q_2 = gli ultimi due simboli riconosciuti erano ab
- q_3 = ho riconosciuto una a seguita da bb

Soluzione



Esempio: ogni a è seguita da bb

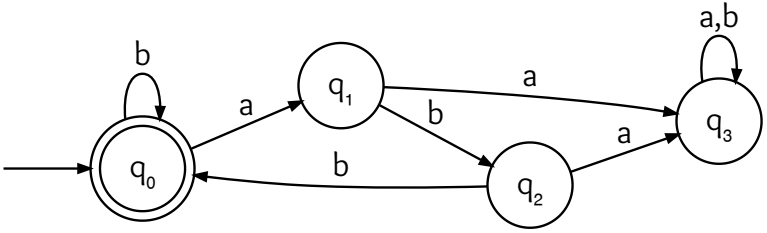
Definire un automa che riconosce le stringhe di a e b in cui **ogni** a è seguita da bb



Stati

- q_0 = ogni a riconosciuta era seguita da bb
- q_1 = l'ultimo simbolo riconosciuto era una a
- q_2 = gli ultimi due simboli riconosciuti erano ab
- q_3 = ho riconosciuto una a seguita da una sequenza diversa da bb

Soluzione



Nota

Una volta raggiunto lo stato q_3 l'automa non ha più speranze di riconoscere alcuna stringa. Per tale motivo, q_3 è detto **stato pozzo**

Esempio: numeri binari multipli di 5

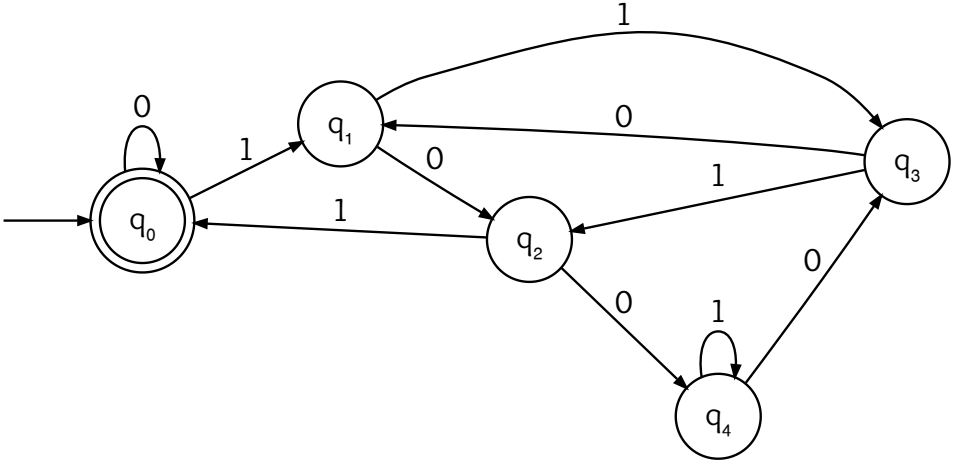
Definire un automa che riconosce le stringhe di bit che rappresentano **multipli di 5**

Osservazioni

Possiamo calcolare il resto della divisione per 5 del numero riconosciuto sapendo che:

- $(2n) \bmod 5 = 2(n \bmod 5) \bmod 5$
- $(2n+1) \bmod 5 = (2(n \bmod 5) + 1) \bmod 5$

Soluzione

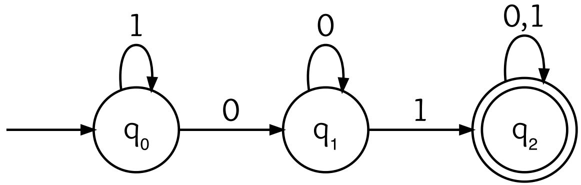


Esercizi sulla definizione di DFA

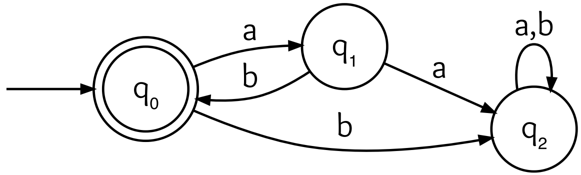
1. Determinare la rappresentazione tabellare di tutti gli automi visti a lezione.
2. Definire un DFA sull'alfabeto { a, c, i, o } che riconosca la sola stringa `ciao`.
3. Definire un DFA sull'alfabeto { a, c, i, o } che riconosca le stringhe che contengono al proprio interno la sottostringa `ciao`, eventualmente preceduta e/o seguita da altri simboli.
4. Definire un DFA sull'alfabeto { a, c, i, o } che riconosca le stringhe che contengono al proprio interno i simboli c, i, a e o in quest'ordine, ciascuno eventualmente preceduto e/o seguito da altri simboli.
5. Definire un DFA sull'alfabeto { `0`, 1 } che riconosca le stringhe di lunghezza arbitraria che iniziano con due `0` e terminano con due 1.
6. Definire un DFA sull'alfabeto { a, b, c } che riconosca le stringhe in cui sono presenti almeno due simboli uguali consecutivi.
7. Definire un DFA sull'alfabeto { a, b, c } che riconosca le stringhe in cui **non** sono presenti due simboli uguali consecutivi. Che relazione c'è tra questo automa e quello dell'esercizio precedente?
8. Definire un DFA sull'alfabeto { a, b, c } che riconosca le stringhe ordinate, assumendo $a \leq b \leq c$.
9. Definire un DFA sull'alfabeto { `/`, `*`, c } che riconosca le stringhe che iniziano con `/*`, che finiscono con `*/` e in cui non ci siano altre occorrenze di `*` seguite da `/`. vale /*/ ?
10. Trovare un DFA più semplice (con meno stati e transizioni) ma equivalente (che riconosce lo stesso linguaggio) di quello in slide 12.

Esercizi sulla comprensione di DFA

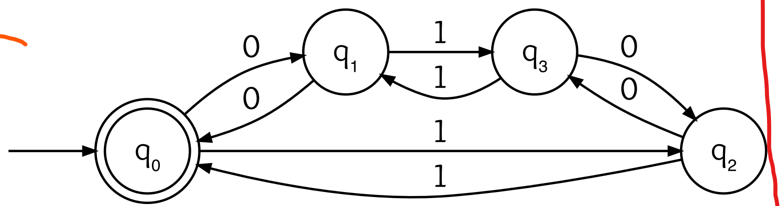
Descrivere a parole il linguaggio riconosciuto dai seguenti automi:



esista uno 0 seguito da un 1



$\{w \mid w = (ab)^n, n \in \mathbb{N}\}$



pari 0 e pari 1

ε 00 0110 0100110

11 0100001