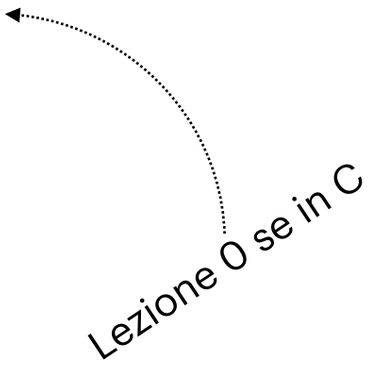
Computabilità, Complessità e Logica

Lezione 1



Informazioni sul corso

Docenti e tutor

Luca Manzoni

Computabilità

Complessità

Logica

Laura Nenzi

Giulia Pietrosanti

Tutorato

Informazioni sul corso

Lezioni e tutorati

	Lunedì	Martedì	Mercoledì	Giovedì	Venerdì
9:00	Lezione				
11:00					
13:00					
14:00			Tutorato	Lezione	
16:00		Lezione			
18:00					

*situazione soggetta a modifiche per la parte di logica del corso

Informazioni sul corso

Seguire il corso

- Le registrazioni del corso saranno disponibili online dopo ogni lezione
- Le slide e il materiale del corso saranno anche loro disponibili online su moodle
- Per i libri di testo saranno fornite indicazioni di capitoli di libri liberamente disponibili
- Per chi vorrà approfondire:
  "Introduction to the theory of computation"
  di Michael Sipser

Informazioni sul corso

Modalità d’esame

- L’esame consiste di due parti:
  - Una prova scritta
  - Un esame orale
- È necessario il superamento della prova scritta per accedere all’orale

La domanda a cui tenteremo di rispondere:

Quali sono le capacità e i limiti fondamentali dei computer?

Computabilità

Quali problemi non sono risolubili da nessun computer?

Complessità

Quali problemi per essere risolti richiedono troppo tempo indipendentemente da quanto veloce sia il nostro computer?

Esprimere un problema

Formalizzare i problemi di decisione

- Se vogliamo studiare un problema dobbiamo essere in grado di esprimerlo
- Vogliamo una formulazione che sia abbastanza generale e flessibile, ma anche formale
- In particolare, esprimeremo problemi chiedendoci se un certo elemento w appartiene a un insieme L
- Per fare questo abbiamo bisogno delle nozioni di *alfabeti*, *parole*, e *linguaggi*.

Alfabeti, parole e linguaggi

Alfabeti

- Un **alfabeto** (solitamente indicato con Σ) è un insieme finito di simboli
  - Alfabeto binario $\Sigma = \{0,1\}$
  - Alfabeto latino $\Sigma = \{a, b, c, d, \ldots, z\}$
- Non è importante la natura dei simboli
- È importante che l'alfabeto contenga un numero *finito* di simboli

Alfabeti, parole e linguaggi

Parole

- Una parola su un alfabeto Σ è una sequenza finita di simboli dell'alfabeto:
- $w = w_1 w_2 \cdots w_n$ dove $w_i \in \Sigma$ per $1 \leq i \leq n$
- e.g., 0100010 è una parola sull'alfabeto $\Sigma = \{0,1\}$
- e.g., *casa* è una parola sull'alfabeto $\Sigma = \{a, b, c, \ldots, z\}$
- e.g., "", o la parola/stringa vuota, spesso indicata con ε o λ, è anch'essa una parola

Alfabeti, parole e linguaggi

Parole

- Data una parola $w = w_1 w_2 \cdots w_n$ sull'alfabeto Σ:
  - $|w|$ denota la **lunghezza** della parola w, ovvero il numero di simboli di cui è composta
    - e.g., $|01001| = 5$
- Diciamo che $x = x_1 x_2 \cdots x_k$ è un prefisso di w se esiste una parola $y = y_1 y_2 \cdots y_h$ tale per cui $x_1 \cdots x_k y_1 \cdots y_h = w$
  - e.g., 010 è un prefisso di 01001

Alfabeti, parole e linguaggi

Linguaggi

- L'insieme di tutte le parole su un alfabeto Σ è indicato con Σ^* o $\Sigma^\star$, pronunciato "*sigma star*".

- Se $\Sigma = \{0,1\}$ allora
  $\Sigma^\star = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, \ \ldots\}$

- Se $\Sigma = \{a, b, c, \ldots, z\}$ allora $\Sigma^\star$ conterrà tutte le possibili sequenze di lettere dell'alfabeto latino. Quindi "*casa*", ma anche "*fdhalcnchuw*".

Alfabeti, parole e linguaggi

Linguaggi

- Dato un alfabeto Σ, un sottoinsieme $L \subseteq \Sigma^\star$ è detto un **linguaggio** su Σ

- Casi particolari sono $L = \emptyset$, il linguaggio vuoto, o $L = \Sigma^\star$, il linguaggio di tutte le parole

- E.g., dato $\Sigma = \{0,1\}$ alcuni linguaggi possibili sono:

  - $L = \{w \in \Sigma^\star : w \text{ contiene almeno uno zero}\}$

  - $L = \{00, 01\}$

Alfabeti, parole e linguaggi

Riassunto

- Un **alfabeto** Σ è un insieme finito di simboli
- Una **parola** $w = w_1 w_2 \cdots w_n$ su Σ è una sequenza finita di simboli di Σ
- L'insieme di tutte le parole possibili su un alfabeto Σ è denotato da $\Sigma^\star$ (o $\Sigma*$)
- Un **linguaggio** L su Σ è un sottoinsieme di $\Sigma^\star$ (i.e., $L \subseteq \Sigma^\star$)

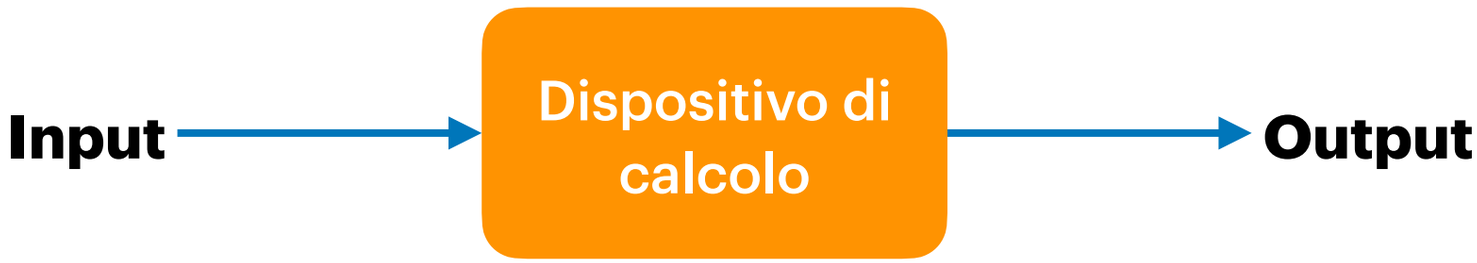
Ma a cosa ci serve tutto questo?

Come si lega con il nostro obiettivo di trovare i limiti di un computer?

Problemi di decisione

E risposte sì/no

Come possiamo astrarre i problemi
che un computer è in grado di risolvere?



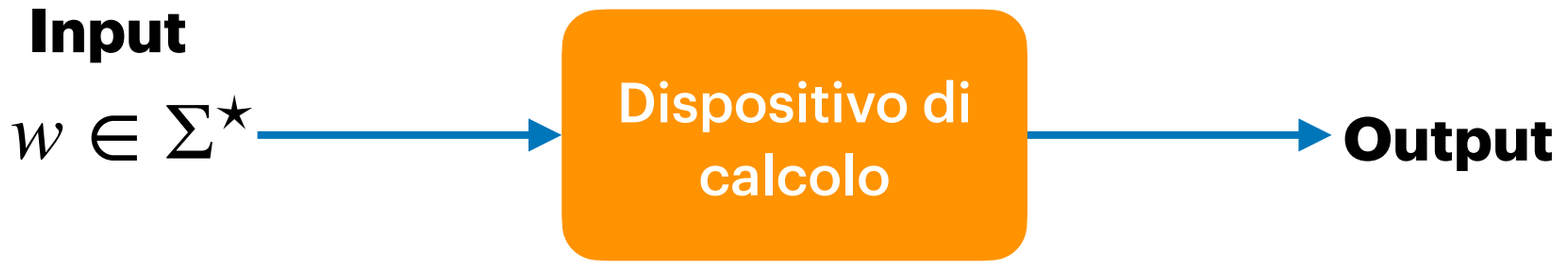
L’input è una parola

- L’input può essere codificato in molti modi
- Potremmo semplificare dicendo che è una sequenza di simboli
- I simboli sono in numero finito
- E l’input è di lunghezza finita

Problemi di decisione

E risposte sì/no

Come possiamo astrarre i problemi
che un computer è in grado di risolvere?



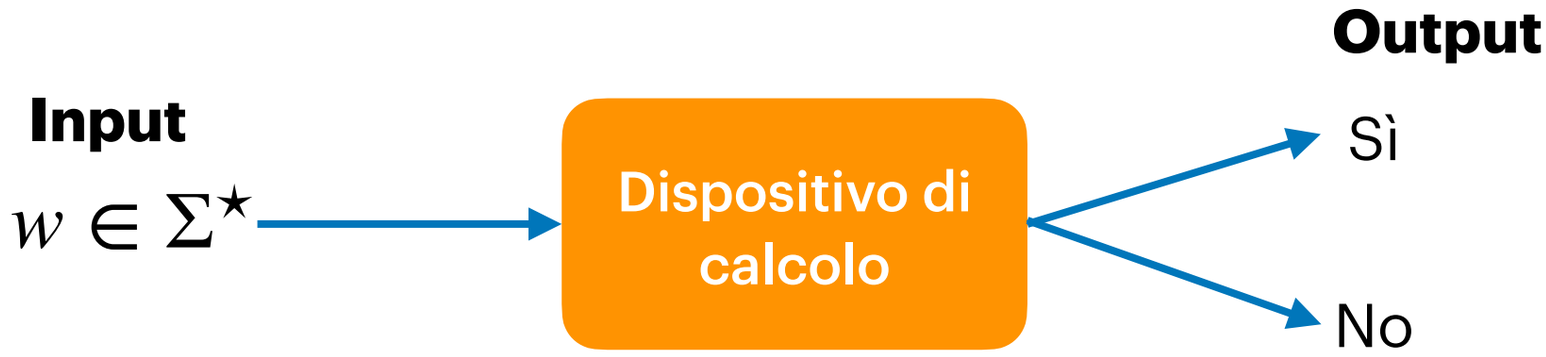
L'output ci dice se l'input appartiene o no a un insieme

- Quale è il modello di output più semplice che possiamo avere?
- Dobbiamo avere almeno **due** output possibili
- Se così non fosse l'output non potrebbe dipendere dall'input
- L'output può essere **sì**/**no**

Problemi di decisione

E risposte sì/no

Cosa abbiamo costruito:



- Un linguaggio va quindi a corrispondere a un **problema di decisione**
- Dato un linguaggio $L \subseteq \Sigma^\star$ e una parola $w \in \Sigma^\star$ possiamo chiedere se $w \in L$
  - Data una sequenza di lettere dell'alfabeto latino, questa rappresenta una parola della lingua italiana?
  - Data questa sequenza di bit questi sono la codifica di una immagine che contiene un gatto?

Che domande ci possiamo porre?

- Dato un linguaggio L:
  - Esiste un dispositivo di calcolo che risponde “sì” per tutte e sole le parole in L?
  - **Computabilità**
  - Quanto tempo o spazio (quante *risorse*) sono necessari per rispondere alla domanda $w \in L$ al crescere della lunghezza di w?
  - **Complessità**

Dispositivi di calcolo

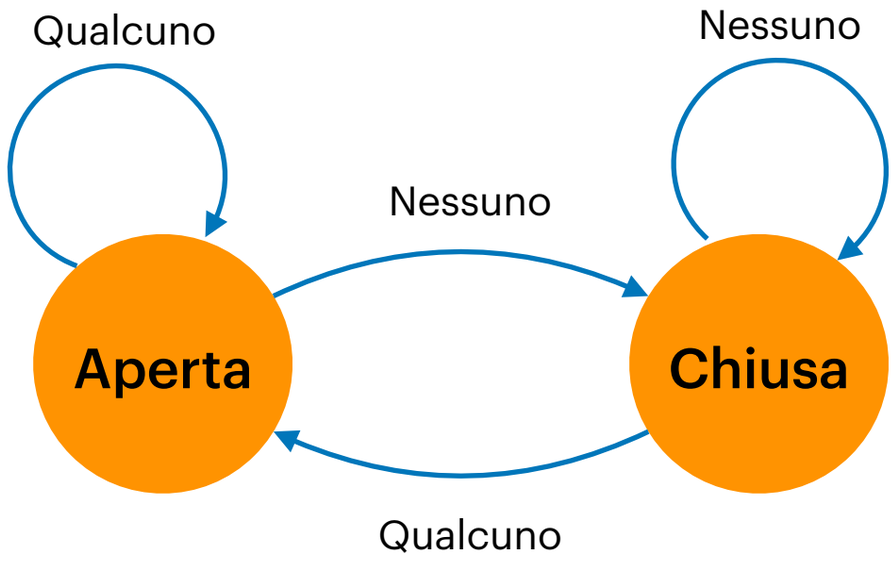
Automi a stati finiti e macchine di Turing

- Il dispositivo di calcolo più “potente” è la **macchina di Turing**
- Però per semplicità inizieremo con gli **automi a stati finiti** prima di passare alle macchine di Turing
- Dobbiamo definire cosa è una computazione (i.e., come viene “elaborato” l’input)
- E come possiamo accettare o rifiutare un input

Un esempio

Porta automatica

- Abbiamo una porta automatica con un sensore
- La porta può essere aperta o chiusa
- Il sensore ci riporta ogni secondo se c'è "qualcuno" o "nessuno" di fronte alla porta
- Dobbiamo far aprire la porta se c'è qualcuno e chiuderla se non c'è nessuno
- Possiamo modellare gli stati in cui la porta può trovarsi e come fare la transizione da uno stato all'altro

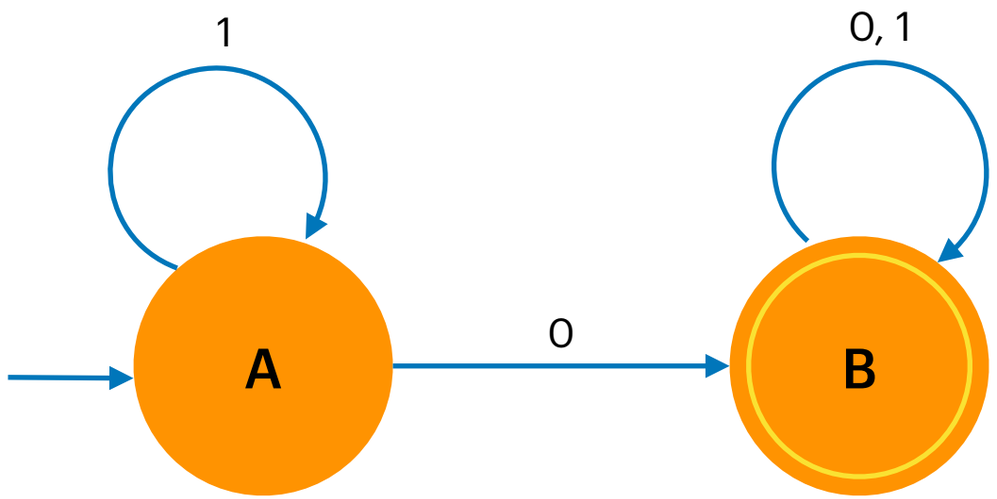


Un esempio

Riconoscere il linguaggio con almeno uno zero

- Come possiamo adattare un automa a stati finiti per riconoscere un linguaggio?
- Gli input del sensore possono essere i simboli di cui è composta una parola
- Possiamo partire da uno stato iniziale...
- ...ricevere i simboli della parola uno alla volta...
- ...e quando abbiamo esaurito la parola verificare in che stato ci troviamo

Un automa che riconosce le stringhe di bit in cui compare zero almeno una volta.
Ma come funziona?



Un esempio

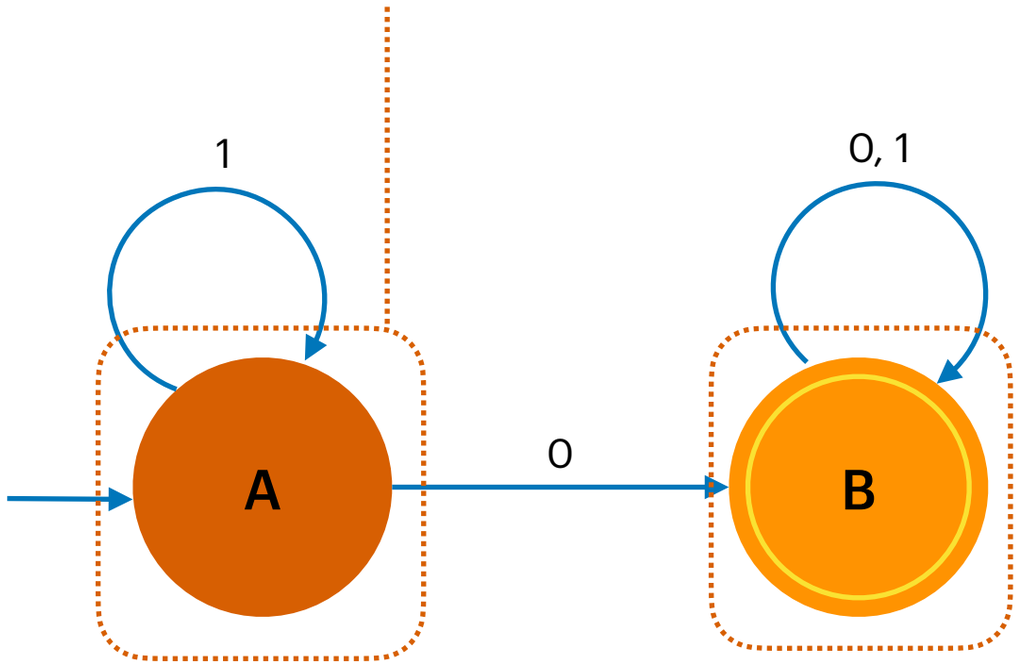
Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101

Parte di parola che ci rimane da leggere

Indichiamo con una freccia entrante il nostro stato iniziale.

Prima di iniziare a leggere la parola in input siamo in quello stato



Indichiamo con un cerchio uno stato accettante.

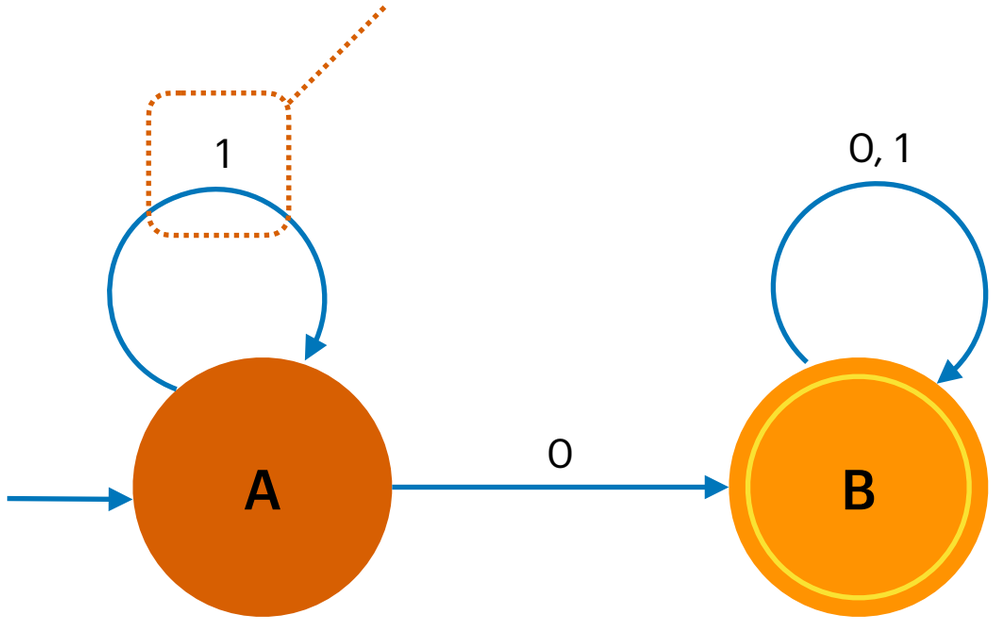
Se terminiamo la lettura su di esso allora accettiamo

Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 1 10101 Parte di parola che ci rimane da leggere

Abbiamo letto 1, quindi seguiamo l'arco etichettato con il valore 1: rimaniamo nello stesso stato

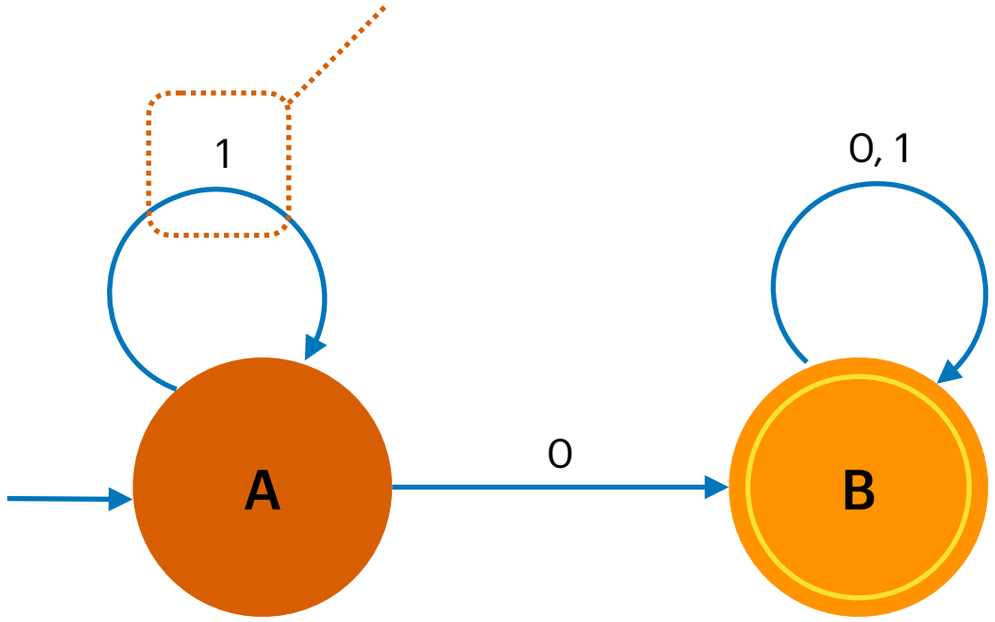


Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101 Parte di parola che ci rimane da leggere

Abbiamo letto nuovamente 1:
rimaniamo nello stesso stato

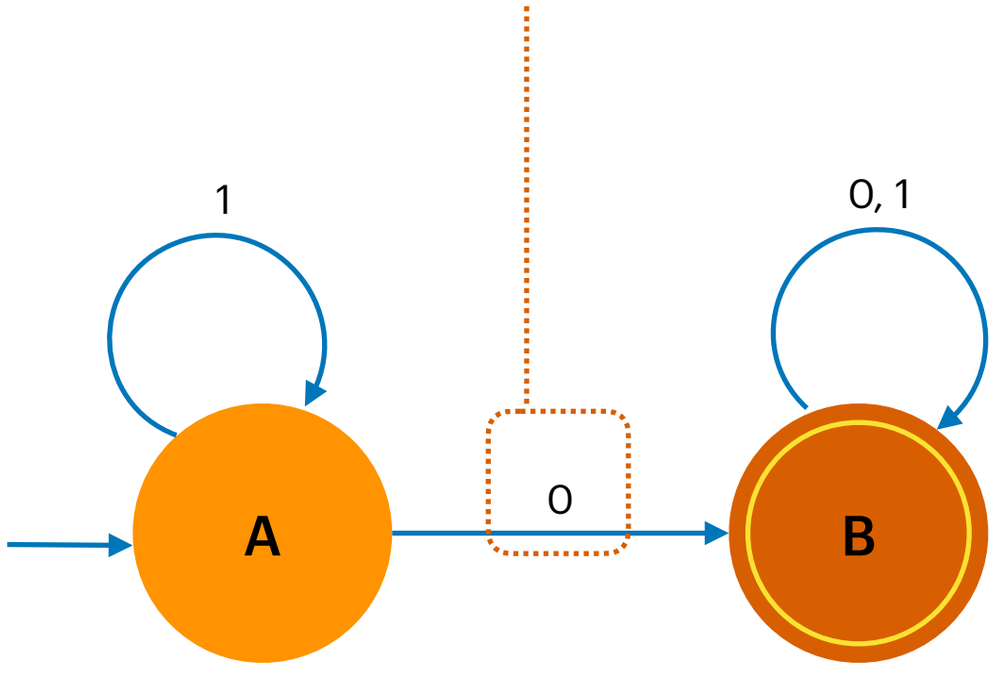


Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101 Parte di parola che ci rimane da leggere

Leggiamo 0, quindi seguiamo l'arco etichettato con 0, cambiando lo stato in cui siamo

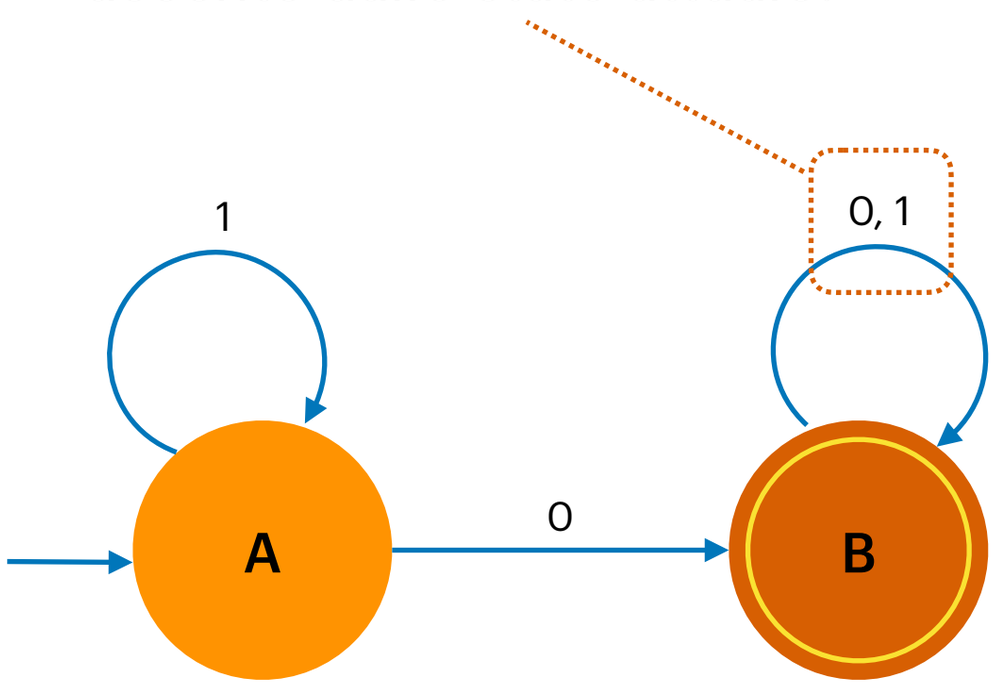


Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101 Parte di parola che ci rimane da leggere

Leggiamo 1, quindi seguiamo l'arco etichettato con 1 *uscente dallo stato attuale*.

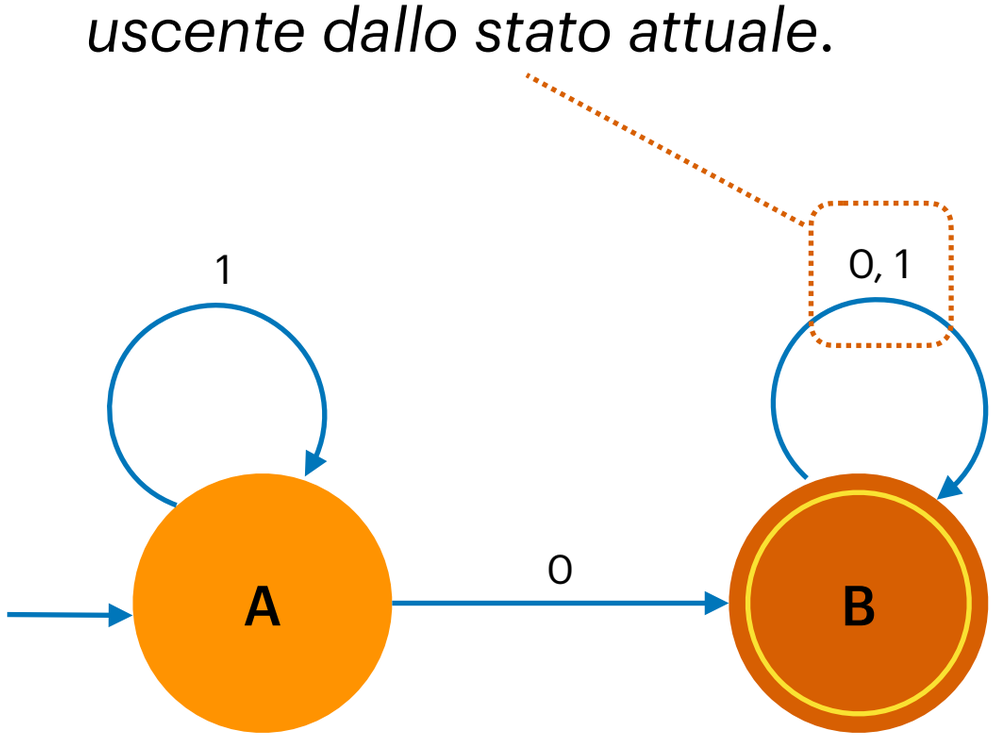


Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101 Parte di parola che ci rimane da leggere

Leggiamo 0, quindi seguiamo l'arco etichettato con 0 *uscente dallo stato attuale*.

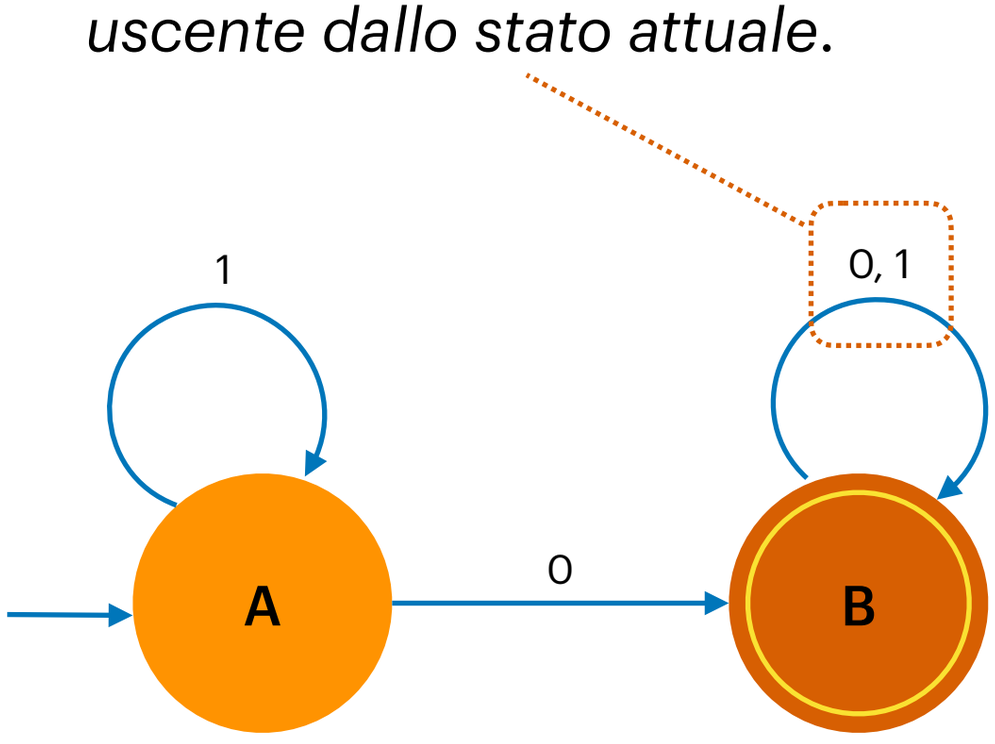


Un esempio

Riconoscere il linguaggio con almeno uno zero

Parola in input: 110101 Lettura completata

Leggiamo 1, quindi seguiamo l'arco etichettato con 1
uscente dallo stato attuale.



Abbiamo completato la lettura in uno stato accettante,
quindi 110101 viene accettata

I prossimi passi

Linguaggi regolari

- Formalizzare cosa è un automa a stati finiti
- Formalizzare la nozione di “segui l’arco etichettato con il simbolo che leggi” (passo di computazione)
- Formalizzare come si arriva ad accettare/rifiutare una parola (computazione)
- Per quali linguaggi esiste un automa a stati finiti che li riconosce? Esistono linguaggi non riconosciuti da automi a stati finiti?