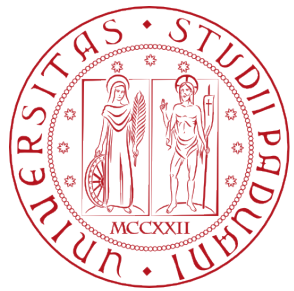
Automi e Linguaggi Formali

Introduzione a Linguaggi regolari e automi a stati finiti

01 Ottobre 2014



UNIVERSITÀ DEGLI STUDI DI PADOVA

A.A. 2014-2015
Enrico Mezzetti
emezzett@math.unipd.it

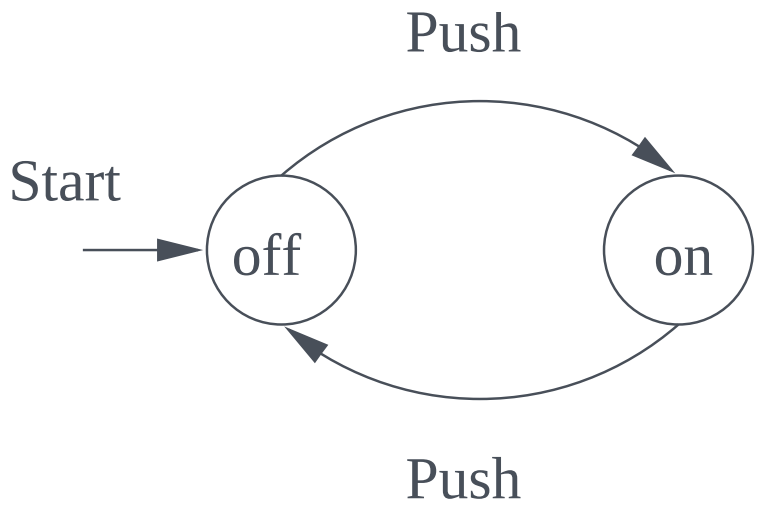
Automi a stati finiti

- Gli **automi a stati finiti** sono usati come modello per
  - Analizzatori lessicali di un compilatore
  - Software per la progettazione di circuiti digitali.
  - Ricerca di parole chiave in un file o sul web.
  - Software per verificare sistemi a stati finiti, come protocolli di comunicazione.

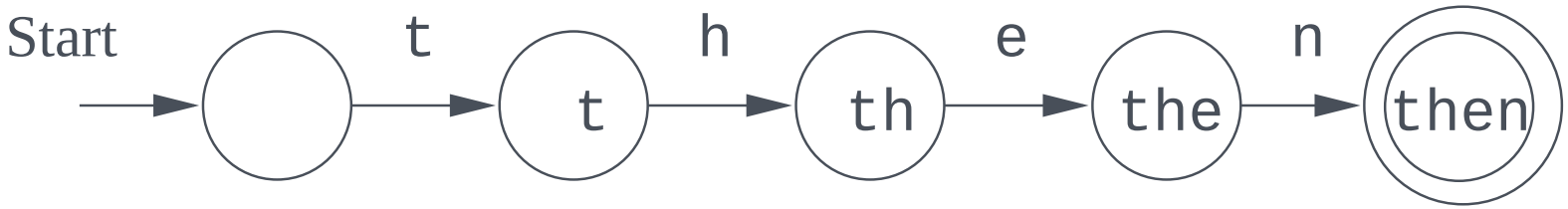
- Insieme finito di **stati**
  - Informazione (parziale) della storia del sistema

Esempi di automi

- Semplice interruttore on/off



- Riconoscitore della stringa `then`



Rappresentazioni strutturali

- **Grammatiche** (manipolazione strutture ricorsive)
  - Ad es, *parser*
  - Regole ricorsive:

$$E \Rightarrow E + E$$

$$Coda \Rightarrow Persona.Coda$$

- **Espressioni regolari** (strutture di dati, stringhe)
  - `'[A-Z][a-z]*[][A-Z][A-Z]'` ▶ `Ithaca NY`
    x `Palo Alto CA`
- Quale espressione e' compatibile con `Palo Alto CA`?

Rappresentazioni strutturali

- **Grammatiche** (manipolazione strutture ricorsive)
  - Ad es, *parser*
  - Regole ricorsive:

$$E \Rightarrow E + E$$

$$Coda \Rightarrow Persona.Coda$$

- **Espressioni regolari** (strutture di dati, stringhe)
  - `'[A-Z][a-z]*[][A-Z][A-Z]'` ▶ `Ithaca NY`
    x `Palo Alto CA`
- Quale espressione e' compatibile con `Palo Alto CA`?

```
'[A-Z][a-z]*([][A-Z][a-z]*)*[][A-Z][A-Z]'
```

Concetti di base della Teoria degli automi

- **Alfabeto** $\rightarrow$ insieme finito e non vuoto di simboli
  - Esempi: $\Sigma = \{0, 1\}$ alfabeto binario
    $\Sigma = \{a, b, c, \ldots, z\}$ lettere minuscole
    $\Sigma = \{0, 1, 2, \ldots, 7\mathtt{F}\}$ caratteri ASCII in base hex
- **Stringa** (parola) $\rightarrow$ sequenza finita di simboli da un alfabeto
  - Esempi: $0011001, 111 \in \Sigma = \{0, 1\}$
    $0011002, 01a \notin \Sigma = \{0, 1\}$
- Caso speciale **stringa vuota** (notazione ϵ)
  - Stringa con zero occorrenze di simboli da Σ
  - Appartiene a qualsiasi alfabeto
- **Lunghezza di una stringa**
  - Numero di *posizioni* per i simboli nella stringa (non # simboli!)
  - $|w| \rightarrow$ lunghezza di w
    Quindi: $|0110| = 4, |\epsilon| = 0$

Alfabeti e stringhe

- **Potenze di un alfabeto**
  - Σ^k = insieme delle stringhe di lunghezza k con simboli da Σ
  - Esempio per $\Sigma = \{0, 1\}$

    $\Sigma^0 = \{\epsilon\}$

    $\Sigma^1 = \{0, 1\}$

    $\Sigma^2 = \{00, 01, 10, 11\}$

    $\Sigma^3 = ?$
- Insiemi notevoli
  - $\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \Sigma^2 \cup \cdots$
  - $\Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \cdots$
  - Quindi vale anche $\Sigma^* = \Sigma^+ \cup \epsilon$
- **Concatenazione**
  - Se x e y sono stringhe $\Rightarrow$ xy e' stringa ottenuta facendo seguire ad una istanza di x un'istanza di y
  - $x = a_1 a_2 \ldots a_i, y = b_1 b_2 \ldots b_j \Rightarrow xy = a_1 a_2 \ldots a_i b_1 b_2 \ldots b_j$
  - Esempio: $x = 01101, y = 110, xy = 01101110$
  - ϵ e' identita' per la concatenazione $(x\epsilon = \epsilon x = x)$

Linguaggi

- **Linguaggio** $\rightarrow$ insieme di stringhe scelte da Σ^*
  - Se Σ e' alfabeto, e $L \subseteq \Sigma^*$, allora L e' un *linguaggio su* Σ
- Esempi di linguaggi:
  - Insieme delle parole italiane
  - Insieme dei programmi C legali
  - Insieme delle stringhe che consistono di n 0 seguiti da n 1
    $$\{\epsilon, 01, 0011, 000111, \ldots\}$$
  - Insieme delle stringhe con un numero uguale di 0 e 1
    $$\{\epsilon, 01, 10, 0011, 0101, 1001, \ldots\}$$
  - Insieme dei numeri binari il cui valore e' primo
    $$L_P = \{10, 11, 101, 111, 1011, \ldots\}$$
  - Il *linguaggio vuoto* $\emptyset$ (per ogni Σ)
  - Il linguaggio $\{\epsilon\} \neq \emptyset$
- Nota: alfabeto Σ e' sempre **finito**
  (linguaggio puo' comprendere un numero infinito di stringhe)

Problemi in teoria degli automi

- **Problema** $\rightarrow$ stringa w e' un elemento di un linguaggio L?
  - Es: decidere se un numero binario e' primo
    $11101 \in L_P$?
    Che risorse computazionali sono necessarie per rispondere a questa domanda?
- Problemi solitamente oltre la risposta si/no
  - ad es. il processo di elaborazione di un output a partire da un input come in un parser per un programma C
  - Non solo "il programma e' corretto?"
  - In caso positivo, deve anche produrre un albero di parsing
- Problema di appartenenza a linguaggio comunque cruciale in teoria della complessita'
  - Se possiamo mostrare che determinare l'appartenenza a L_X e' un problema complesso $\Rightarrow$ allora lo sara' anche fare il parsing di programmi scritti in X