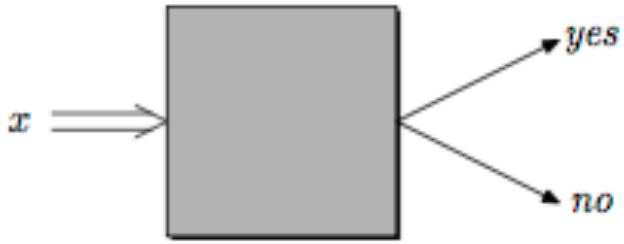
Fondamenti d'Informatica: Accettazione e riconoscimento di linguaggi

Barbara Re, Phd

Che cosa significa?

- Le grammatiche formali costituiscono uno strumento generativo per la rappresentazione e la classificazione di linguaggi
- Le grammatiche formali non consentono di impostare in modo **algoritmico** il problema del riconoscimento di un linguaggio, e cioè il problema di decidere, dato un linguaggio L e una stringa x, se $x \in L$
- Il concetto fondamentale per quanto riguarda il **riconoscimento di linguaggi** è il concetto di **automa**, inteso come dispositivo astratto che, data un stringa x fornitagli in input, può eseguire una computazione ed eventualmente, se la computazione termina, restituisce, secondo una qualche modalità, un valore che, nel caso del problema del riconoscimento, sarà **booleano**

Schema generale di automa

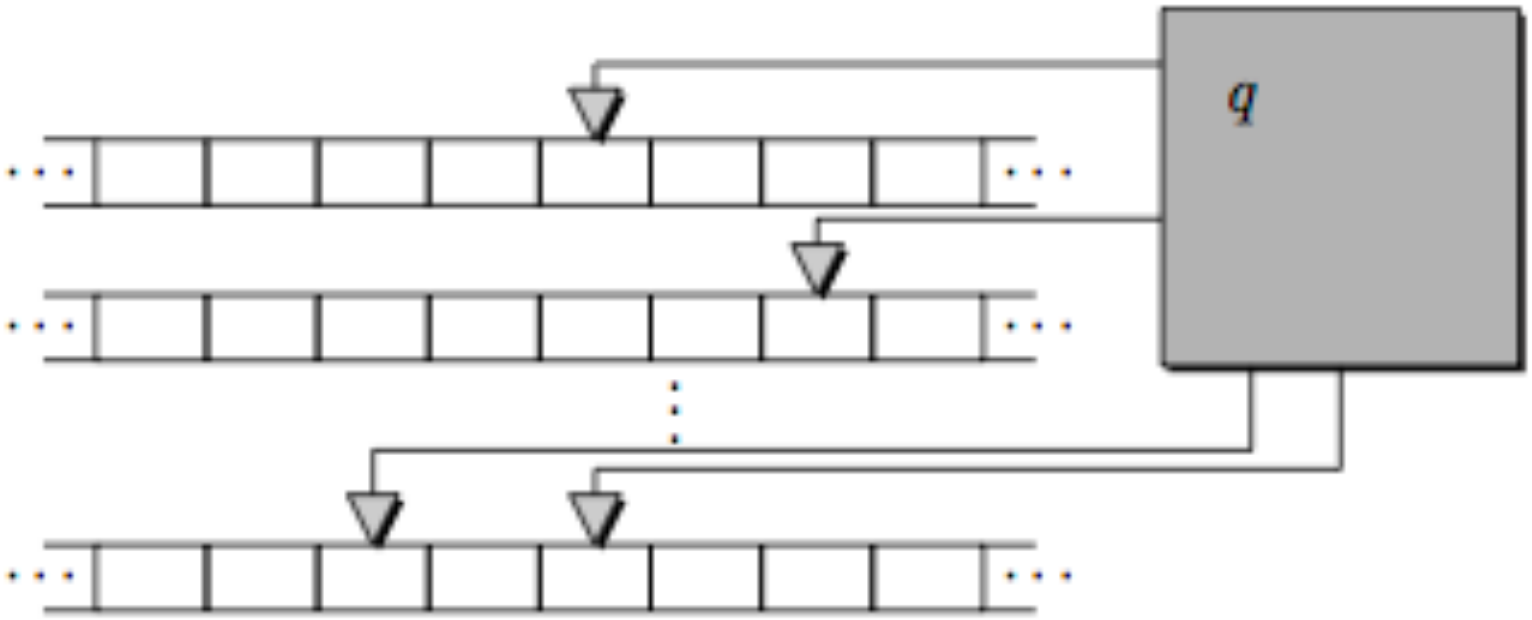


La struttura di un automa comprende un **dispositivo interno**, che ad ogni istante assume uno **stato** in un possibile insieme predefinito: lo stato rappresenta essenzialmente l'informazione associata al funzionamento interno

Schema generale di automa

- Un automa può utilizzare **uno o più dispositivi di memoria**, sui quali è possibile memorizzare delle informazioni, sotto forma di stringhe di caratteri da alfabeti anch'essi predefiniti
- In genere si assume che tali dispositivi di memoria siano **nastri**, sequenze cioè di celle, ognuna delle quali può contenere un carattere: **uno tra tali nastri contiene inizialmente l'input fornito all'automa**
- I caratteri vengono letti o scritti **per mezzo di testine** che possono muoversi lungo i nastri, posizionandosi sulle diverse celle

Schema particolareggiato di automa



Diversi tipi di automi possono essere definiti facendo ipotesi diverse sulle capacità di movimento, lettura e scrittura delle testine sui vari nastri

Funzionamento automa

- Il funzionamento di un automa è definito rispetto a due concetti di
  - **Configurazione**
  - **Funzione di transizione**

La configurazione

- Una configurazione rappresenta l'insieme delle informazioni che determinano, in un certo istante, il comportamento futuro dell'automa

- Una configurazione è composta dalle seguenti informazioni

1. Stato interno dell'automa
2. Contenuto di tutti i nastri di memoria
3. Posizione di tutte le testine sui nastri

- Esiste una sola configurazione, detta **configurazione iniziale**, che rappresenta la situazione complessiva in cui si trova l'automa all'inizio, nel momento in cui la stringa di input gli viene sottoposta

La funzione di transizione

- La funzione di transizione, che è parte della definizione della struttura dell'automa, induce una relazione di transizione tra configurazioni, che associa ad una configurazione un'altra (o più di una) configurazione successiva
- L'applicazione della funzione di transizione ad una configurazione si dice **transizione** o **mossa** o **passo computazionale** dell'automa
- Per ogni automa A, date due configurazioni c_i, c_j di A, useremo nel seguito la notazione c_i |-- A-- c_j per indicare che c_i e c_j sono correlate dalla relazione di transizione, vale a dire che c_j deriva da c_i per effetto dell'applicazione della funzione di transizione di A

Accettazione o Rifiuto

- Alcune configurazioni, aventi strutture particolari e dipendenti dalla struttura dell'automa, sono inoltre considerate come **configurazioni di accettazione**
- Tutte le altre configurazioni sono definite come **Configurazioni di non accettazione**, o di rifiuto

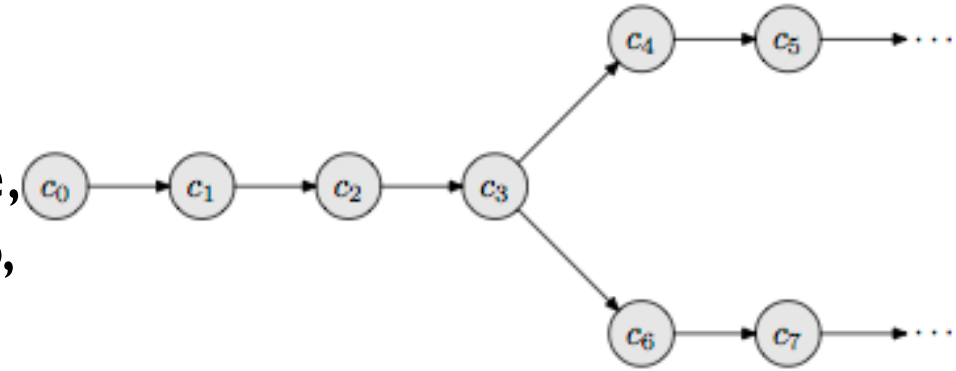
Computazione di accettazione e rifiuto

- Possiamo quindi vedere una computazione eseguita da un automa A a partire da una configurazione iniziale c_0 come una sequenza di configurazioni $c_0, c_1, c_2, \ldots$ tale che, per $i = 0, 1, \ldots$, si ha c_i |-- A-- c_{i+1}.
- Indicheremo anche nel seguito con la notazione |-- *A*-- la **chiusura transitiva e riflessiva della relazione** |-- A--
- È facile allora rendersi conto che, dato un automa A e due configurazioni c_i |-- A-- c_j di A, si ha c_i |-- *A*-- c_j se e solo se esiste una computazione che porta A da c_i a c_j
- Se la sequenza di configurazioni $c_0, c_1, c_2, \ldots, c_n$ ha lunghezza finita e se è massimale, nel senso che non esiste nessuna configurazione c tale che c_n |-- A-- c, allora diciamo che la computazione termina
  - È una **computazione di accettazione** se termina in una configurazione di accettazione,
  - È una **computazione di rifiuto** nel caso contrario in cui termini in una configurazione non di accettazione

Automi deterministici e non deterministici

- Un automa è detto deterministico se ad ogni stringa di input associa una sola computazione, e quindi una singola sequenza di configurazioni

- Un automa è detto non deterministico se esso associa ad ogni stringa di input un numero qualunque, in generale maggiore di uno, di computazioni



Automi e classi di linguaggi

- Esistono infiniti linguaggi per cui non esistono algoritmi (né automi) di riconoscimento
- Una considerazione analoga è stata fatta in relazione alle grammatiche
- Un problema decisionale (o un linguaggio) viene detto decidibile se esiste un algoritmo che risolve ogni istanza del problema
- E' ammissibile che non esistono algoritmi di decisione, ma solo algoritmi in grado di riconoscere correttamente un'istanza positiva del problema
  - In queste situazioni si parla di semi-decidibilità

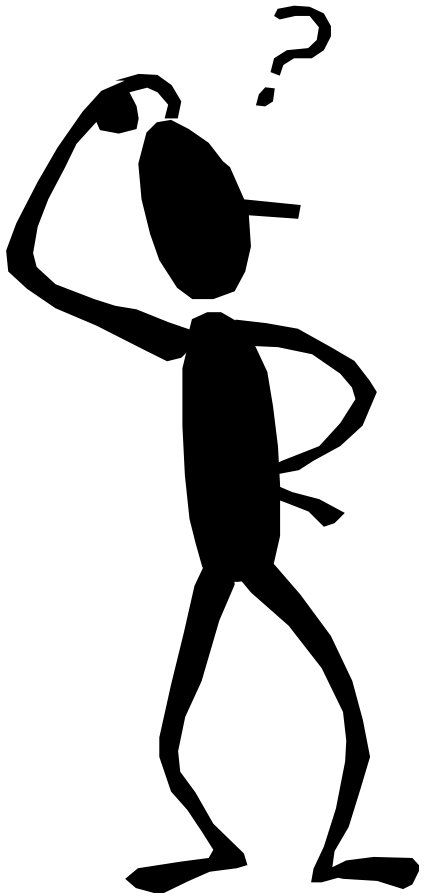
Decidibile e Semi-decidibile

- Un problema decisionale (o un linguaggio) è detto **decidibile** se esiste un algoritmo (in particolare, un automa) che per ogni istanza del problema risponde correttamente vero oppure falso
  - In caso contrario il problema è detto **indecidibile**
- Un problema decisionale (o un linguaggio) è detto **semi-decidibile** se esiste un algoritmo (in particolare, un automa) che per tutte e sole le istanze positive del problema risponde correttamente vero.

I linguaggi di tipo 1, 2 e 3 sono decidibili, mentre quelli di tipo 0 sono semi-decidibili

Anticipiamo qualcosa!

- Per i linguaggi di tipo 3 esistono dispositivi di riconoscimento che operano con memoria costante e tempo lineare
  - Automi a stati finiti
- Per i linguaggi strettamente di tipo 2, il riconoscimento può avvenire sempre in tempo lineare ma con dispositivi di tipo non deterministico dotati di una memoria a pila
  - Automi a pila non deterministici
- Per i linguaggi strettamente di tipo 1, l'esigenza di tempo e di memoria è ancora maggiore. Per essi si può mostrare che il riconoscimento può essere effettuato con una macchina non deterministica che fa uso di una quantità di memoria che cresce linearmente con la lunghezza della stringa da esaminare
  - Macchina di Turing non deterministica "linear bounded", o Automa lineare
- Per i linguaggi strettamente di tipo 0, si farà riferimento a un dispositivo di calcolo costituito da un automa che opera con quantità di tempo e di memoria potenzialmente illimitate
  - Macchina di Turing



Questions?