Funzione di transizione estesa e ASFND

Created	@March 31, 2025 10:46 AM
Class	Fondamenti di informatica: linguaggi formali

Un modo alternativo ed ampiamente utilizzato di descrivere il comportamento di un automa a stati finiti deterministico e di definire il linguaggio da esso riconosciuto utilizza una estensione della funzione di transizione da caratteri a stringhe.

Definizione 3.5 *La* funzione di transizione estesa *di un automa a stati finiti deterministico $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$ è la funzione $\overline{\delta} : Q \times \Sigma^* \mapsto Q$, definita nel seguente modo:*

$$\begin{aligned} \overline{\delta}(q, \varepsilon) &= q \\ \overline{\delta}(q, xa) &= \delta\left(\overline{\delta}(q, x), a\right), \end{aligned}$$

dove $a \in \Sigma$, $x \in \Sigma^*$.

Dalla definizione precedente, avremo quindi che, dato uno stato q ed una stringa di input $x \in \Sigma^*$, lo stato q' è tale che $q' = \overline{\delta}(q, x)$ se e solo se la computazione eseguita dall'automa a partire da q ed in conseguenza della lettura della stringa x conduce l'automa stesso nello stato q'. Più formalmente, possiamo anche dire che $q' = \overline{\delta}(q, x)$ se e solo se esiste $y \in \Sigma^*$ tale che $(q, xy) \overset{*}{\longmapsto} (q', y)$. Di conseguenza, avremo che una stringa $x \in \Sigma^*$ è accettata da un ASFD $\mathcal{A} = \langle \Sigma, Q, \delta, q_0, F \rangle$ se e solo se $\overline{\delta}(q_0, x) \in F$.

È allora possibile introdurre la seguente definizione di linguaggio riconosciuto da un ASFD.

L'espressione nell'immagine descrive una proprietà della **funzione di transizione estesa** (δ^*) in un **automa a stati finiti.** Analizziamola passo per passo:

$$q' = \delta^*(q, x) \text{ se e solo se esiste } y \in \Sigma^* \text{ tale che } (q, xy) \overset{*}{\Longrightarrow} (q', y).$$

Interpretazione

- $\delta^*(q, x)$ rappresenta lo stato raggiunto dall'automa partendo da q e leggendo la stringa x.
- L'espressione $(q, xy) \overset{*}{\Longrightarrow} (q', y)$ afferma che, partendo da q e leggendo l'intera stringa xy, l'automa arriva in q' lasciando y come parte della stringa da leggere.

Perché è vera?

Questa proprietà è una conseguenza della definizione ricorsiva della funzione di transizione estesa δ^*, che è costruita nel seguente modo:

1. **Caso base:**

   $$\delta^*(q, \epsilon) = q$$

   Se non si legge alcun simbolo (stringa vuota ϵ), lo stato non cambia.

2. **Caso ricorsivo:**
   Per una stringa qualsiasi xy, possiamo decomporla come:

   $$\delta^*(q, xy) = \delta(\delta^*(q, x), y).$$

   Questo significa che leggendo prima x, l'automa arriva in $q' = \delta^*(q, x)$, e poi la parte restante y può essere processata a partire da q'.

Ora, la notazione $(q, xy) \overset{*}{\Longrightarrow} (q', y)$ esprime esattamente questo processo: il passaggio da q a q' mentre si legge x, lasciando y da leggere.

Conclusione

L'uguaglianza $q' = \delta^*(q, x)$ è equivalente alla condizione che esista una stringa y tale che, leggendo xy dall'inizio in q, si arrivi in q' con y ancora da leggere. Questo segue direttamente dalla definizione di δ^* ed è una proprietà chiave della funzione di transizione estesa negli automi.

La **funzione di transizione estesa** δ^* si definisce in modo ricorsivo come:

1. **Caso base**:

$$\delta^*(q, \epsilon) = q$$

(dove ϵ è la stringa vuota: se non si legge nulla, lo stato rimane invariato).

2. **Caso ricorsivo**:
Per ogni stringa $w \in \Sigma^*$ e ogni simbolo $a \in \Sigma$,

$$\delta^*(q, wa) = \delta(\delta^*(q, w), a)$$

Questo significa che, per leggere wa, l'automa prima elabora w, arrivando in uno stato intermedio, e poi applica δ per processare il simbolo finale a.

Esempio 3.2 Al fine di verificare che la stringa aab è accettata dall'ASFD di Figura 3.2, deriviamo il valore di $\delta(q_0, aab)$ nel modo seguente:

$$\overline{\delta}(q_0, aab) = \delta\left(\overline{\delta}(q_0, aa), b\right) = \delta\left(\delta\left(\overline{\delta}(q_0, a), a\right), b\right) = \delta\left(\delta\left(\delta\left(\overline{\delta}(q_0, \varepsilon), a\right), a\right), b\right) =$$

$$= \delta\left(\delta\left(\delta\left(q_0, a\right), a\right), b\right) = \delta\left(\delta\left(q_0, a\right), b\right) = \delta\left(q_0, b\right) = q_1.$$

Definizione 3.6 *Il* linguaggio riconosciuto *da un automa a stati finiti deterministico $\mathcal{A}$ è l'insieme*

$$L(\mathcal{A}) = \left\{x \in \Sigma^* \mid \overline{\delta}(q_0, x) \in F\right\}.$$

Linguaggio riconosciuto per ASFD se per ogni stringa, che è finita, in input ho un termine.

Linguaggio accettato, ossia se la computazione può anche non terminare.

R è un insieme di linguaggi per cui esiste per ognuno un ASF che lo riconosce, classe di linguaggi regolari, può essere definita da diversi “punti di vista”, tipo espressioni regolari, ASF, grammatiche, relazione di equivalenza, formule di sistemi formali ecc...

Il vantaggio è che certe proprietà possono essere dimostrate in maniera piu semplice analizzandola da un determinato punto di vista, sapendo che comunque essa varrà per tutti gli altri punti di vista.

3.2 Automi a stati finiti non deterministici

In questa sezione prenderemo in esame la classe più estesa degli automi a stati finiti, costituita dalla classe degli *automi a stati finiti non deterministici*, che risulterà particolarmente utile per studiare le proprietà dei linguaggi regolari. Tale estensione si ottiene, coerentemente con quanto esposto nella Sezione 2.5.2, definendo l'insieme delle transizioni non mediante una funzione, ma attraverso una relazione o, equivalentemente, come una funzione nell'insieme

delle parti dell'insieme degli stati. In altre parole tali automi sono caratterizzati dal fatto che, ad ogni passo, una applicazione della funzione di transizione definisce più stati anziché uno solo.

Definizione 3.7 *Un* automa a stati finiti non deterministico *(ASFND) è una quintupla* $\mathcal{A}_N = \langle \Sigma, Q, \delta_N, q_0, F \rangle$*, in cui* $\Sigma = \{a_1, \ldots, a_n\}$ *è l'*alfabeto di input*,* $Q = \{q_0, \ldots, q_m\}$ *è l'insieme finito e non vuoto degli* stati interni*,* $q_0 \in Q$ *è lo* stato iniziale*,* $F \subseteq Q$ *è l'insieme degli* stati finali*, e* $\delta_N : Q \times \Sigma \mapsto \mathcal{P}(Q)$ *è una funzione (parziale), detta* funzione di transizione*, che ad ogni coppia* $\langle stato, carattere \rangle$ *su cui è definita associa un sottoinsieme di* Q *(eventualmente vuoto).*[4]

Un automa a stati finiti non deterministico può essere anch'esso descritto, così come un ASFD, per mezzo di un grafo di transizione: in questo caso, per effetto del non determinismo, esisteranno archi uscenti dal medesimo nodo etichettati con il medesimo carattere.

Funzione di Transizione Estesa per Automata Non Deterministici (NFA)

Nel caso degli automi **non deterministici**, la funzione di transizione può essere definita come:

$$\delta : Q \times \Sigma \rightarrow P(Q)$$

Dove:

- Q è l'insieme degli stati dell'automa,
- Σ è l'alfabeto,
- $P(Q)$ è l'insieme delle parti di Q (cioè l'insieme di tutti i sottoinsiemi di Q).

In altre parole, per un dato stato q e simbolo a, la funzione $\delta(q, a)$ restituisce un **sottoinsieme di stati** di Q, il che implica che l'automa può passare a più stati contemporaneamente, invece di determinare un solo stato successivo come accade negli automi deterministici.

Funzione di Transizione Estesa per NFA

Analogamente alla funzione di transizione estesa nel deterministico, anche in un automa non deterministico si può definire una funzione di transizione estesa, che permette di determinare lo stato in cui l'automa si troverà dopo aver letto una stringa intera, non solo un singolo simbolo. La funzione di transizione estesa per un NFA è definita come:

$$\delta^* : Q \times \Sigma^* \rightarrow P(Q)$$

Dove Σ^* è l'insieme di tutte le stringhe possibili dell'alfabeto Σ (comprese le stringhe vuote). La transizione estesa per le stringhe di simboli si definisce ricorsivamente, esattamente come nel caso deterministico, ma ora, invece di restituire un singolo stato, essa restituisce un **insieme di stati**.

Definizione della Funzione di Transizione Estesa per NFA

La funzione di transizione estesa δ^* è definita nel seguente modo:

1. **Caso base (stringa vuota):**

   Se la stringa è vuota (ϵ), non c'è alcun simbolo da leggere, quindi la transizione non cambia lo stato, ma restituisce l'insieme contenente solo lo stato corrente:

$$\delta^*(q, \epsilon) = \{q\}$$

2. **Caso ricorsivo (stringa non vuota):**

   Se la stringa è composta da un simbolo a seguito da una sottostringa w, la funzione di transizione estesa si comporta nel seguente modo:

   $$\delta^*(q, aw) = \bigcup_{r \in \delta(q,a)} \delta^*(r, w)$$

   Dove:

   - $\delta(q, a)$ è l'insieme degli stati raggiungibili dal stato q leggendo il simbolo a,
   - $\delta^*(r, w)$ è la transizione estesa per la sottostringa w, partendo dallo stato r,
   - L'unione di tutti i risultati delle transizioni estese su ciascun stato in $\delta(q, a)$ restituisce l'insieme finale di stati raggiungibili dopo aver letto tutta la stringa aw.

Esempio

Supponiamo di avere un NFA con i seguenti stati $Q = \{q_0, q_1, q_2\}$, alfabeto $\Sigma = \{a, b\}$, e funzione di transizione δ definita come:

- $\delta(q_0, a) = \{q_0, q_1\}$ (dal stato q_0, leggendo "a", si può passare sia a q_0 che a q_1),
- $\delta(q_0, b) = \{q_2\}$,
- $\delta(q_1, a) = \{q_2\}$,
- $\delta(q_1, b) = \{q_0\}$,
- $\delta(q_2, a) = \{q_1\}$,
- $\delta(q_2, b) = \{q_2\}$.

Per una stringa come "ab", la transizione estesa partendo da q_0 sarebbe:

1. Iniziamo con q_0 e leggiamo il primo simbolo "a". La funzione $\delta(q_0, a)$ ci dà i due stati $\{q_0, q_1\}$.
2. Poi, leggiamo "b". Per ciascun stato in $\{q_0, q_1\}$, eseguiamo la transizione per "b":
   - Da q_0, $\delta(q_0, b) = \{q_2\}$,
   - Da q_1, $\delta(q_1, b) = \{q_0\}$.

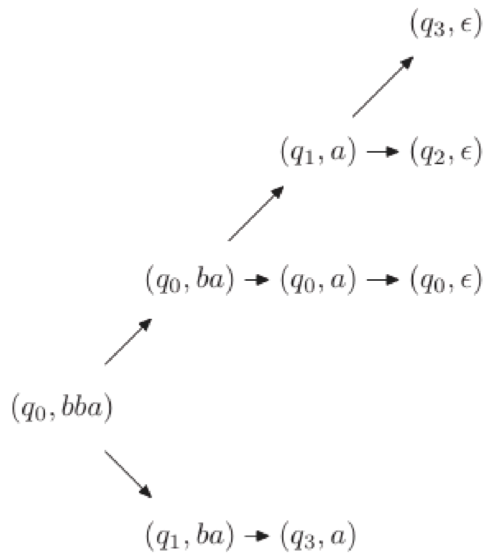

FIGURA 3.6 Albero di computazione relativo all'Esempio 3.7.

Esempio 3.8 Nel caso dell'Esempio 3.7, la stringa *bba* viene accettata in quanto una delle computazioni conduce alla configurazione di accettazione (q_3, ε). L'accettazione di *bba* deriva anche dal fatto che al termine della computazione non deterministica l'automa, finita di leggere la stringa, si trovi nell'insieme di stati $\{q_0, q_2, q_3\}$, e che $q_3 \in F$.

Possiamo allora definire il linguaggio $L(\mathcal{A})$ accettato da un ASFND $\mathcal{A}$ nel modo seguente:

$$L(\mathcal{A}) = \left\{ x \in \Sigma^* \mid (\{q_0\}, x) \overset{*}{\longmapsto} (\mathcal{Q}, \varepsilon), \mathcal{Q} \cap F \neq \emptyset \right\}.$$

Una funzione di transizione (normale) non può mai accettare Sigma*, poiché le stringhe di questo insieme sono infinite, si utilizza Sigma che è il solo insieme finito di caratteri dell'automa (alfabeto).

Domanda 3

Sia A_N Automa a Stati Finiti Non Deterministico sull'alfabeto di input $\Sigma = \{a, b\}$. Allora nel diagramma degli stati di A_N:

- ○ Tutti i nodi del diagramma degli stati di A_N, che corrispondono a stati finali di A_N, non hanno archi uscenti.
- ○ Il nodo del diagramma degli stati di A_N corrispondente a q_0, non ha nessun arco uscente.
- ○ Ogni nodo del diagramma degli stati di A_N ha esattamente un arco uscente etichettato "a" e esattamente un arco uscente etichettato "b".
- ◉ Potrebbe esistere un nodo del diagramma degli stati di A_N con esattamente due archi uscenti etichettati "a" e con esattamente un arco uscente etichettato "b".

Analisi delle affermazioni:

1. **"Tutti i nodi del diagramma degli stati di A_N che corrispondono a stati finali di A_N non hanno archi uscenti."**
   - Questa affermazione non è necessariamente vera. In un NFA, gli stati finali possono avere archi uscenti, quindi questa opzione è **falsa**.
2. **"Il nodo del diagramma degli stati di A_N corrispondente a q_0 non ha nessun arco uscente."**
   - In un NFA, lo stato iniziale q_0 può avere transizioni verso altri stati, quindi questa affermazione è **falsa**.
3. **"Ogni nodo del diagramma degli stati di A_N ha esattamente un arco uscente etichettato 'a' e esattamente un arco uscente etichettato 'b'."**
   - Questo vincolo è tipico di un **DFA**, ma in un **NFA** uno stato può avere più transizioni con la stessa etichetta o addirittura nessuna. Quindi questa affermazione è **falsa**.
4. **"Potrebbe esistere un nodo del diagramma degli stati di A_N con esattamente due archi uscenti etichettati 'a' e con esattamente un arco uscente etichettato 'b'."**
   - Questa affermazione è compatibile con il comportamento di un NFA, poiché può avere **più transizioni con la stessa etichetta.** Un singolo stato può avere più archi uscenti per lo stesso simbolo di input. Quindi questa affermazione è **vera.**

Risposta corretta:

L'ultima opzione selezionata nell'immagine è **corretta**, ovvero:
✅ **"Potrebbe esistere un nodo del diagramma degli stati di A_N con esattamente due archi uscenti etichettati 'a' e con esattamente un arco uscente etichettato 'b'."**

Domanda 4

Segnare, fra le definizioni che seguono, la definizione di linguaggio riconosciuto da un Automa a Sati Finiti Non Deterministico $A_N = \langle \Sigma, Q, \delta_N, q_0, F \rangle$.

- ○ $L(A_N) = \{x \in \Sigma^* \text{ tali che } \overline{\delta}_N(p, x) \cap F \neq \emptyset\}$
- ○ $L(A_N) = \{x \in Q^* \text{ tali che } \overline{\delta}_N(q_0, x) \cap F \neq \emptyset\}$
- ◉ $L(A_N) = \{x \in \Sigma^* \text{ tali che } \overline{\delta}_N(q_0, x) \cap F \neq \emptyset\}$
- ○ $L(A_N) = \{x \in \Sigma^* \text{ tali che } \delta_N(q_0, x) \cap F \neq \emptyset\}$

banale, linguaggio → stringhe → funzione di transizione estesa, e quindi

◉ $L(A_N) = \{x \in \Sigma^* \text{ tali che } \overline{\delta}_N(q_0, x) \cap F \neq \emptyset\}$

x = stringa, e quindi appartiene a Σ^* , $\overline{\delta}_n(q0, x)$

La n al pedice indica che l'automa è non deterministico.

Definizione di Linguaggio riconosciuto di un ASFND in termini di funzione di transizione

Un modo alternativo di definire il linguaggio accettato da un ASFND richiede, analogamente a quanto fatto nel caso degli ASFD, di estendere alle stringhe la definizione di funzione di transizione.

Definizione 3.8 *Dato un ASFND, la* funzione di transizione estesa *è la funzione* $\overline{\delta}_N : Q \times \Sigma^* \mapsto \mathcal{P}(Q)$, *definita nel seguente modo*

$$\overline{\delta}_N(q, \varepsilon) = \{q\}$$

$$\overline{\delta}_N(q, xa) = \bigcup_{p \in \overline{\delta}_N(q,x)} \delta_N(p, a)$$

dove $a \in \Sigma$, $x \in \Sigma^*$, $p \in Q$.

Dalla definizione precedente, avremo quindi che, dato uno stato q ed una stringa di input x, lo stato q' appartiene all'insieme $\overline{\delta}_N(q, x)$ se e solo se esiste una computazione dell'automa la quale, a partire da q ed in conseguenza della lettura della stringa x, conduce allo stato q'.

Cosa esprime la formula:

> Per calcolare l'insieme di stati raggiungibili leggendo una stringa xa a partire da uno stato q, si:
>
> 1. "calcola l'insieme di stati raggiungibili da q leggendo $x \to \overline{\delta}_N(q, x)$;"
> 2. "per ognuno di questi stati p, si applica la transizione con il simbolo a: $\delta_N(p, a)$;"
> 3. "si fa l'unione di tutti gli insiemi ottenuti → questo è $\overline{\delta}_N(q, xa)$."

Se ho un ASFND che riconosce L, allora partendo da questo, posso costruire un ASF che riconosce L e viceversa. quindi è una implicazione se e solo se.

Costruzione di sottoinsiemi

Nel processo di determinizzazione:

- Ogni stato dell'ASFND corrisponde a **un sottoinsieme di stati** dell'ASFN.
- L'insieme degli stati dell'ASFND è quindi, al massimo, l'insieme delle **parti** di Q, ovvero $2^{|Q|}$.

🔧 Formalmente

Se l'ASFN ha:

- Q: insieme degli stati, con $|Q| = n$

Allora l'ASFND risultante avrà al più:

- 2^n stati, uno per **ogni possibile sottoinsieme** di Q.

Ma come hai detto bene:

> ***"...non tutti gli stati potrebbero contribuire al cammino di accettazione della stringa."***

Quindi:

- **In pratica**, non tutti i 2^n stati vengono effettivamente generati. Solo quelli **raggiungibili** a partire dallo stato iniziale (e quindi dai suoi epsilon-chiusure, se presenti).
- L'automa deterministico che costruisci con la costruzione di potenza può avere **molti meno** stati del limite teorico.

🧠 Intuizione

- Ogni stato dell'ASFND "simula" un insieme di configurazioni dell'ASFN: tutti gli stati in cui l'automa non deterministico **potrebbe trovarsi** dopo aver letto una parte dell'input.
- Questo consente all'ASFND di "replicare" il comportamento dell'ASFN in modo deterministico.