6 - Automi a stati finiti (deterministici e non deterministici)

(Si ringrazia Giada per una parte di appunti scritti a mano)

Automa deterministico

Un automa a stati finiti deterministico, noto anche come accettore a stati finiti deterministico, è un modello matematico utilizzato nell'informatica teorica per riconoscere linguaggi formali. Un automa a stati finiti è vantaggioso per avere una memoria limitata più facile da gestire, ma comunque con vincoli molto importanti.

Definizione (stati deterministici)

Sia X un alfabeto, un automa a stati finiti (**FSA**) è una quadrupla definita con:

$$M = (Q, \delta, q_0, F)$$

dove:

- Q è un insieme finito e non vuoto di **stati**, chiamati anche memoria dell'automa e, come i simboli NT, permettono di effettuare transizioni nella fase di riconoscimento
- δ è una funzione di Q, detta **funzione di transizione**:

$$\delta : Q \times X \to Q$$

- q_0 è lo stato iniziale (dove comincia il processo di riconoscimento)
- $F \subseteq Q$ è l'insieme degli stati di accettazione o finali, dove, se l'automa termina le transizioni in quello stato, la parola viene accettata.

Talora i valori della funzione di transizione δ non sono definiti per tutte le coppie (stato-simbolo di ingresso) (q, x), si dice che δ è una **funzione parziale** o definita parzialmente. Questo significa che la lettura di x dà luogo in q ad un comportamento dell'automa che non si ritiene utile descrivere ai fini del riconoscimento (nel senso che produrrebbe stringhe non accettate).
Evidentemente questo fatto può essere descritto in modo equivalente, seguendo la definizione data di automa a stati finiti, passando da q, per effetto di x, in uno stato dal quale non si possa mai raggiungere uno stato finale (chiamato anche **stato pozza**).
Lo stato pozza non è altro che uno stato generato **a seguito di input su uno stato non definito nel dominio**

Rappresentazione di un FSA

Grafo degli stati/Diagramma di Transizione/ Diagramma di stato

È una rappresentazione grafica in cui:

- Ogni stato $q \in Q$ è rappresentato da un cerchio, o nodo, con etichetta q
- Lo stato iniziale (nodo q_0) ha un arco orientato entrante libero (ossia che non proviene da nessun altro nodo)
- per ogni stato $q \in Q$ e per ogni simbolo x dell'alfabeto di ingresso, $x \in X$, se $\delta(q, x) = q'$, esiste un arco orientato etichettato con x uscente dal nodo q ed entrante nel nodo

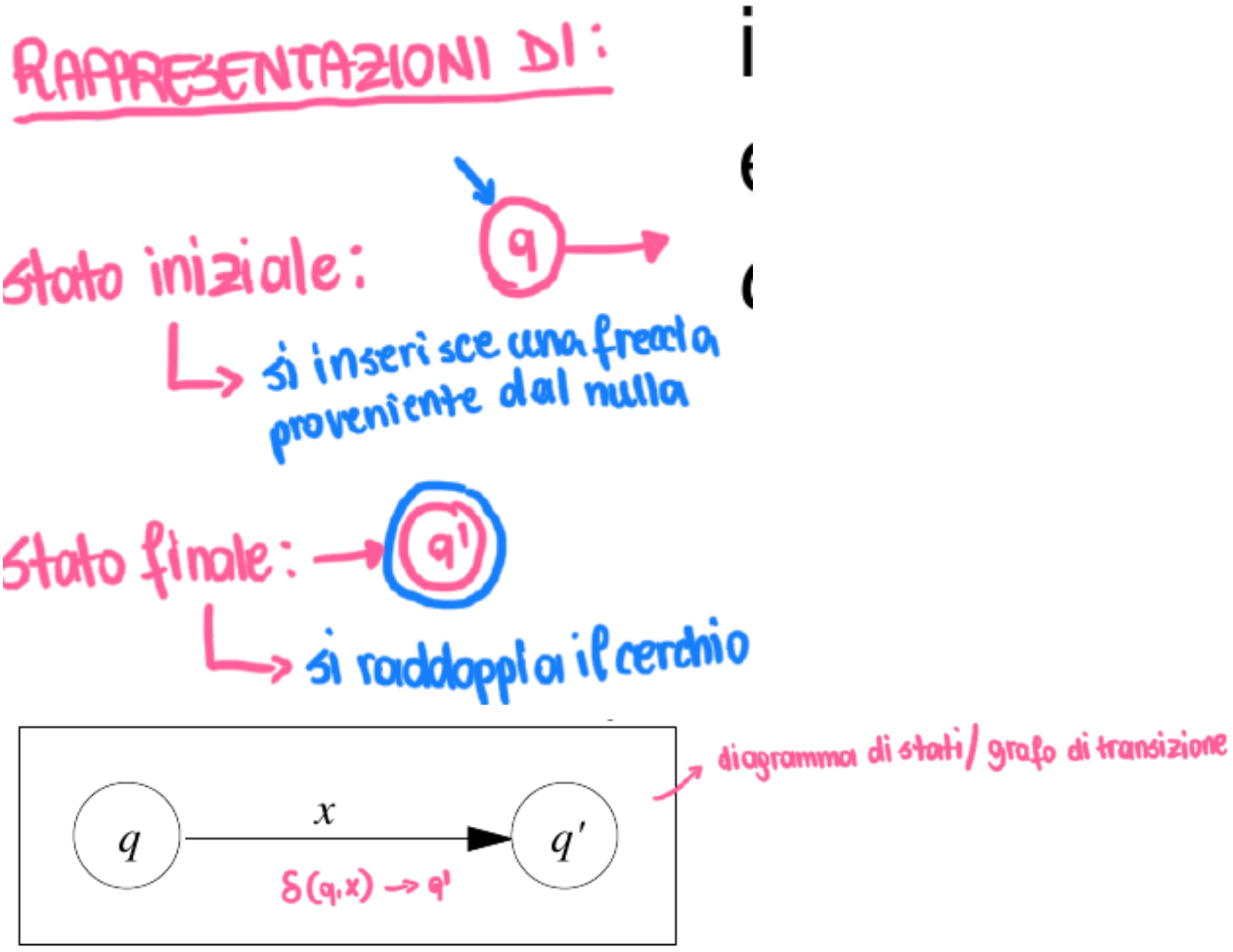
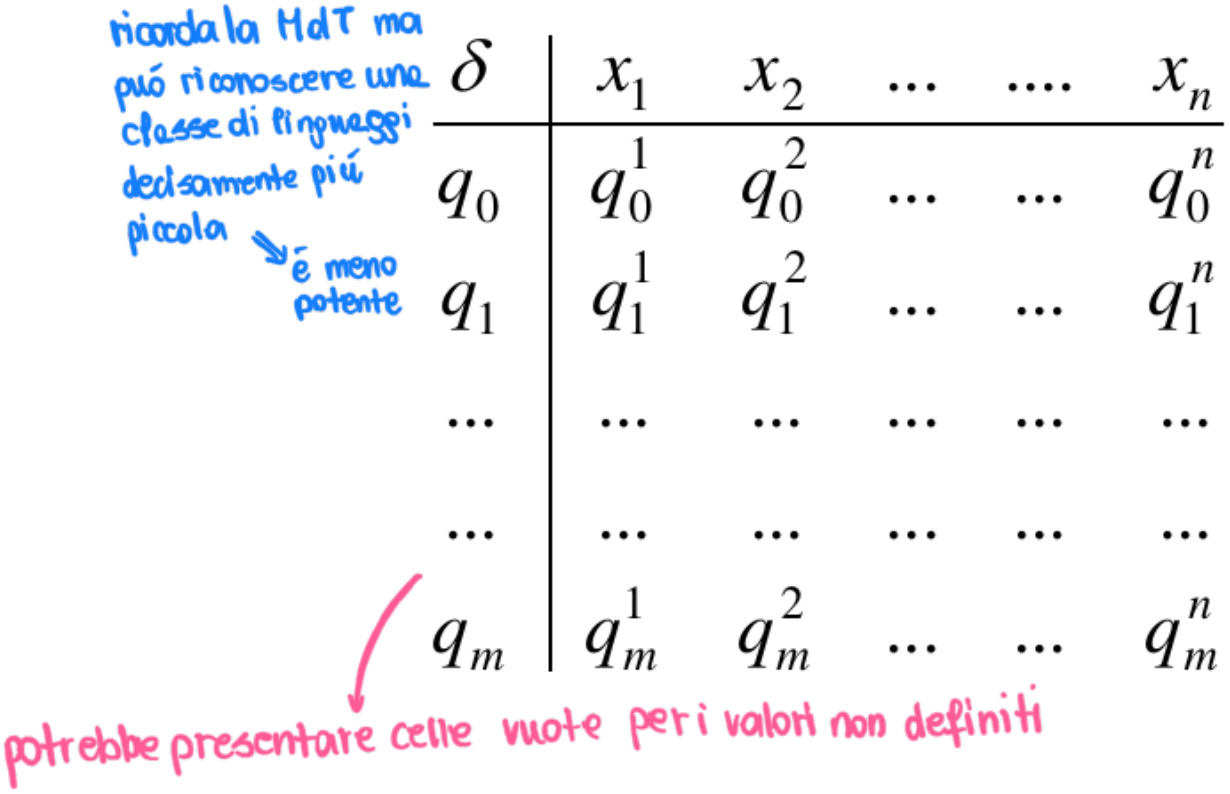


Tavola di Transizione

È una tabella in cui sono riportati gli stati sulle righe e i simboli dell'alfabeto di ingresso sulle colonne.
Per ogni coppia (stato-ingresso) si legge nella tavola lo stato successivo:



δ	x_1	x_2	...	....	x_n
q_0	q_0^1	q_0^2	...	...	q_0^n
q_1	q_1^1	q_1^2	...	...	q_1^n
...	...	...	...	...	...
...	...	...	...	...	...
q_m	q_m^1	q_m^2	...	...	q_m^n

dove l'alfabeto di ingresso e l'insieme degli stati sono rispettivamente:$X = \{x_1, x_2, \ldots, x_n\}$ e $Q = \{q_0, q_1, \ldots, q_m\}$

Quindi si ha che:
$\delta(q_i, x_j) = q_i^j \quad \text{con} \quad q_i, q_i^j \in Q, x_j \in X,$
Ovvero **Se l'automa è nello stato q^i e legge il simbolo x^j, allora passa nello stato q_j^i.**

In altre parole:

- Ogni cella della tavola di transizione contiene q_i^j, che è lo stato raggiunto quando l'automa è in q_i e legge x_j.
- Poiché è un **FSA**, per ogni coppia (q_i, x_j) esiste **esattamente un unico stato** q_i^j (determinismo).

Estensione della funzione di transizione

L'**estensione della funzione di transizione** δ^* è un concetto fondamentale negli automi a stati finiti (FSA e NDA) che generalizza la funzione di transizione base δ per lavorare con **intere stringhe** (sequenze di simboli) invece di singoli caratteri.

Definizione:
Dato un automa a stati finiti(FSA) $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X definiamo la funzione:

$$\delta^* : Q \times X^* \to Q$$

tale che $\delta^*(q, w)$, per $q \in Q$ e $w \in X^*$, sia lo stato in cui M si porta avendo in ingresso la parola w su X e partendo dallo stato q
Definizione ricorsiva

partendo dallo stato q.

assioma ← $\delta^*(q,\lambda) = q$ (parola + piccola)

$\delta^*(q,wx) = \delta(\delta^*(q,w),x)$ (ultima transizione che accetta l'ultimo simbolo)

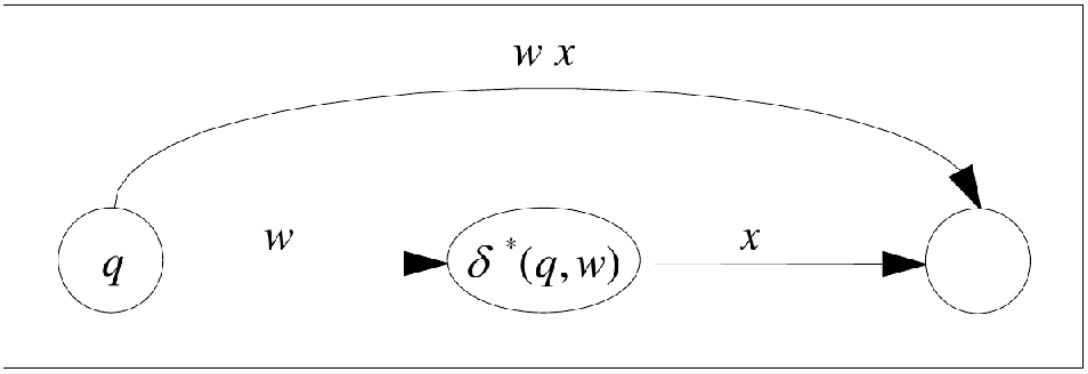
per ogni $q \in Q,\ x \in X,\ w \in X^*$

parola, ultimo simbolo; definita nei termini di δ (funzione nota e conosciuta)

definita ricorsivamente dove la parola viene diminuita ogni volta del suo ultimo simbolo fino ad arrivare all'assioma

Linguaggi di Programmazione 6/27

La prima parte è il caso base, dove con una stringa vuota si rimane nello stato corrente, nel passo induttivo si calcola prima lo stato processando w e applicando la transizione per l'ultimo simbolo w



Parola accettata o riconosciuta da un FSA

Sia $M = (Q, \delta, q_0, F)$ un FSA con alfabeto di ingresso X. Una parola $w \in X^*$ è **accettata** (o **riconosciuta**) da M se, partendo dallo stato iniziale q_0, lo stato q in cui l'automa si porta alla fine della sequenza di ingresso w è uno stato finale.

$$w \text{ accetata} \overset{def}{\iff} \delta(q_0, w) \in F$$

Questa definizione coincide con il concetto di parola generata da una grammatica

Linguaggio accettato o riconosciuto da un FSA

Sia $M = (Q, \delta, q_0, F)$ un FSA con alfabeto di ingresso X, **il linguaggio accettato o riconosciuto** da M è il seguente sottoinsieme di X^*:

$$T(M) = \{w \in X^* | \delta(q_0, w \in F)\}$$

$T(M)$ è il linguaggio generato dall'automa M,ovvero l'insieme di tutte le stringhe che, quando processate da M, portano a uno stato finale.
Questa definizione coincide con il concetto di grammatiche equivalenti

FSA equivalenti

Sia $M_1 = (Q_1, \delta_1, q_1, F_1)$ ed $M_2 = (Q_2, \delta_2, q_2, F_2)$ due FSA di alfabeto di ingresso X, M_1 e M_2 si dicono equivalenti se:

$$T(M_1) = T(M_2)$$

Linguaggi a stati finiti

Definizione di una nuova classe di linguaggi
Dato un alfabeto X, un linguaggio L su X è un **linguaggio a stati finiti** (o FSL - Finite State Language) se esiste un automa che lo accetta FSA M con alfabeto di ingresso X tale che $L = T(M)$

Classe dei linguaggi a stati finiti

La **classe dei linguaggi a stati finiti** (indicata come $\mathcal{L}_{FSL}$) è l'insieme di tutti i linguaggi formali che possono essere **riconosciuti da un automa a stati finiti deterministico (FSA)** o non deterministico (NDA).
Questi linguaggi sono strettamente legati alle **grammatiche regolari** e costituiscono la classe più semplice nella gerarchia di Chomsky.
Definizione:
Dato un alfabeto X, la classe $\mathcal{L}_{FSL}$ è definita come:

$$\mathcal{L}_{FSL} = \{L \subseteq X^* \mid \exists \text{ FSA } M \text{ t.c } L = T(M)\}$$

Automa non deterministico

Dato in ingresso un alfabeto X,un automa a stati finiti non deterministico(**FSA**) è una quadrupla definita con:

$$M = (Q, \delta, q_0, F)$$

dove:

- Per Q, q_0, F valgono le definizioni date per gli FSA
- $\delta : Q \times X \to 2^Q$ è la funzione di transizione che assegna ad ogni coppia (stato-simbolo di ingresso) (q, x) un insieme $\delta(q, x) \subseteq Q$ di possibili stati successivi

Un NDA è un FSA con l'unica eccezione che, in corrispondenza di una coppia (stato simbolo di ingresso) (q, x), vi è un insieme di stati in cui l'automa può transitare (stati successivi possibili)

Estensione della funzione di transizione per un NDA

Come per gli FSA si può definire un'estensione della funzione di transizione δ come segue:
Definizione δ^* per NDA
Dato un NDA $M = (Q, \delta, q_0, F)$ con alfabeto di ingresso X, definiamo per induzione la funzione:

$$\delta^* : 2^Q \times X^* \to 2^Q$$

La definizione è induttiva:

Si parte dal **caso base** (con parola vuota λ):

$$M = (Q, \delta, q_0, F)$$

Per arrivare al passo **induttivo**:

$$\delta^*(q, xw') = \bigcup_{p \in \delta(q,x)} \delta^*(p, w')$$

Se leggo una parola composta da un simbolo iniziale x seguito da una parola w, allora:

- prima vedo dove posso andare da q leggendo x: $\delta(q, x)$ è un insieme di stati.
- poi, per ciascuno di questi stati p, calcolo $\delta^*(p, w)$, ossia dove posso arrivare leggendo w da p.
- l'unione di tutti questi insiemi è il risultato finale.

Analogamente a quanto fatto per gli FSA, si dovrebbero riformulare le definizioni di parola accettata e di linguaggio accettato da un NDA. La complicazione, rinveniente dalla computazione non deterministica dello stato successivo in cui un NDA transita, comporta che una stessa parola può indurre cammini multipli attraverso un NDA, alcuni che terminano

in stati di accettazione, altri che terminano in stati di non accettazione.

Esempio 6.1

Sia dato il seguente NDA $M = (Q, \delta, q_0, F)$ (Figura 6.3):

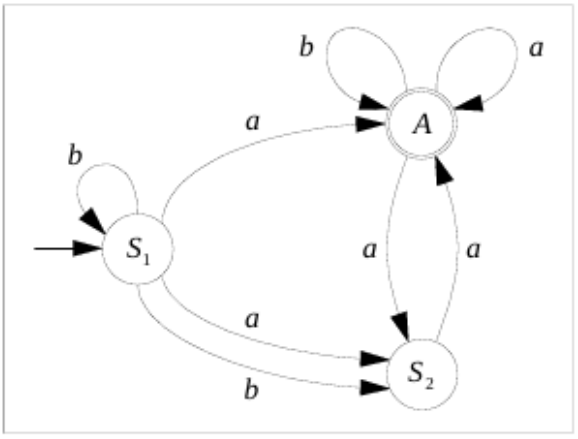


Figura 6.3

$\delta(\{S_1\}, a) = \{A, S_2\}$ $\quad$ $\delta(\{S_1\}, b) = \{S_1, S_2\}$

... $\quad$ ...

... $\quad$ ...

Supponiamo di avere la parola $w = aba$. Vogliamo stabilire se w è riconosciuta dall'automa (Figura 6.4).

Partiamo da S_1:

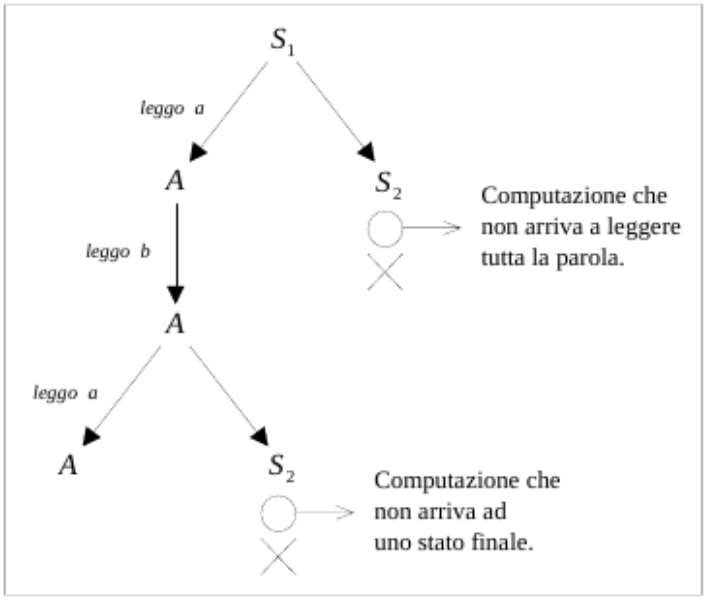


Figura 6.4

Parola accettata o riconosciuta da un NDA

Sia $M = (Q, \delta, q_0, F)$ un NDA con alfabeto di ingresso X. Una parola $w \in X^*$ è **accettata** (o **riconosciuta**) da M se, partendo dallo stato iniziale q_0, se esiste almeno un modo per M di portarsi in uno stato di accettazione alla fine della sequenza di ingresso w

$$w \text{ accettata} \iff \exists p \in \delta^*(\{q_0\}, w) \cap F \iff \delta^*(\{q_0\}, w) \cap F \neq \emptyset$$

Linguaggi accettati o riconosciuti da un NDA

Sia $M = (Q, \delta, q_0, F)$ un NDA con alfabeto di ingresso X, **il linguaggio accettato o riconosciuto** da M è l'insieme delle parole su X accettate da M

$$T(M) = \{w \in X^* \mid \delta^*(\{q_0\}, w) \cap F \neq \emptyset\}$$

(è l'insieme delle parole w per le quali esiste almeno un cammino, etichettato con lettere di w nell'ordine da sinistra a destra, attraverso il diagramma degli stati che porta M dallo stato iniziale ad uno degli stati di accettazione).

NDA equivalenti

Sia $M_1 = (Q_1, \delta_1, q_1, F_1)$ ed $M_2 = (Q_2, \delta_2, q_2, F_2)$ due NDA di alfabeto di ingresso X, M_1 e M_2
si dicono equivalenti se:

$$T(M_1) = T(M_2)$$

Classe dei linguaggi riconosciuti da automi a stati finiti non deterministici

La classe dei linguaggi riconosciuti da automi a stati finiti non deterministici è definita come l'insieme di tutti i linguaggi formali che possono essere accettati da almeno un automa a stati finiti non deterministico.

Data la notazione nel documento:

$\mathcal{L}_{NDL} = \{L \in 2^{X^*} \mid \exists M,\ M \text{ è un NDA} : L = T(M)\}$

- (2^{X^*}) rappresenta l'insieme di tutti i linguaggi possibili sull'alfabeto X

Caratteristiche Principali

1. **Equivalenza con gli automi deterministici**: Un risultato fondamentale è che questa classe è esattamente equivalente alla classe dei linguaggi riconosciuti da automi a stati finiti deterministici (FSA). Questo significa che per ogni NDA esiste un FSA equivalente che riconosce lo stesso linguaggio, e viceversa.
2. **Metodo di riconoscimento**: Un NDA accetta una parola se esiste almeno un cammino computazionale (tra i molti possibili a causa del non determinismo) che porta da lo stato iniziale a uno stato finale.
3. **Potere espressivo**: Nonostante il non determinismo, gli NDA non possono riconoscere linguaggi più complessi di quelli riconoscibili da FSA. Il non determinismo offre spesso una rappresentazione più compatta o intuitiva degli stessi linguaggi.

Equivalenza e trasformazione dei linguaggi riconosciuti da automi a stati finiti non deterministici

Gli automi a stati finiti deterministici (FSA) e non deterministici (NDA) riconoscono la **stessa classe di linguaggi**, ovvero i **linguaggi regolari**. Questo significa che:

- **Ogni NDA può essere convertito in un FSA equivalente** (che riconosce lo stesso linguaggio).
- **Ogni FSA è già un caso particolare di NDA** (dove ogni transizione porta a un solo stato).

Teorema

- 1a formulazione): Le classi dei linguaggi $(\mathcal{L}_{FSL})$ e $(\mathcal{L}_{NDL})$ sono equivalenti.
- 2a formulazione):Sia L un linguaggio su X, L è un linguaggio a stati finiti se e solo se $L = T(M)$ per qualche NDA M.

Dimostrazione

Parte 1: Da FSA a NDA (⇒)

- **Ipotesi**: $L \in \mathcal{L}_{FSL}$ (esiste un FSA M_1 che riconosce L)
- **Costruzione**: Dato $M_1 = (Q_1, \delta_1, q_1, F_1)$, definiamo un NDA M_2 tale che:
  $M_2 = (Q_2, \delta_2, q_2, F_2)$
  dove:
  - $Q_2 = Q_1$
  - $\delta_2(q, x) = \{\delta_1(q, x)\}$ (transizioni come insiemi singoli)
  - $q_2 = q_1$
  - $F_2 = F_1$
- **Risultato**: $T(M_2) = T(M_1)$
  Ogni FSA è un caso particolare di NDA con transizioni deterministiche "impacchettate" in insiemi.
  Parte 2: Da NDA a FSA (⇐)
- **Ipotesi**: $L \in \mathcal{L}_{NDL}$ (esiste un NDA M che riconosce L)
- **Costruzione**: Convertiamo $M = (Q, \delta, q_0, F)$ in un FSA $M' = (Q', \delta', q_0', F')$ tramite la **costruzione dei sottoinsiemi**:
  - $Q' = 2^Q$ (tutti i sottoinsiemi di Q)
  - $q_0' = \{q_0\}$
  - $F' = \{p \subseteq Q \mid p \cap F \neq \emptyset\}$
  - $\delta'(q, x) = \bigcup_{q \in p} \delta(q, x)$ per ogni $p \in Q'$, $x \in X$
- **Funzionamento**:
  - L'FSA simula **tutti i possibili percorsi** dell’NDA in parallelo.
  - Uno stato p dell' FSA rappresenta l’insieme degli stati in cui l’NDA potrebbe trovarsi.
- **Risultato**: $T(M') = T(M)$