Esercizi Fondamenti dell'Informatica

federico.serafini@studenti.unipr.it

2021/2022

Sommario

Lo scopo principale di questo testo è di fornire alcune possibili soluzioni a una parte degli esercizi presenti nel libro di testo[1] del corso di Fondamenti dell'Informatica. L'autore di questo documento è uno studente che ha cercato di seguire quanto più possibile *the KISS principle* con l'intenzione di rivolgersi ad un pubblico di studenti.

Attenzione: vista l'inesperienza dell'autore è possibile che ci siano alcuni errori e/o imprecisioni, per uno studio adeguato della materia è fondamentale riferirsi principalmente al materiale fornito dal docente e ad i libri di testo consigliati, quindi consultare con cautela questo elaborato.

[1] A. Dovier, R. Giacobazzi. Fondamenti dell'informatica: Linguaggi formali, calcolabilità e complessità. Bollati Boringhieri, ISBN9788833933795, 2020.

Indice

1 Nozioni Preliminari

1.1 Definizioni, Teoremi e Lemmi

Definizione 1.1 (Cardinalità di un insieme)**.** *Sia S un insieme, la cardinalità di S si indica con $|S|$ e corrisponde al numero di elementi in S.*

Esempio. Sia $S = \{\, 0, 1 \,\}$, $|S| = 2$.

Definizione 1.2 (Insieme enumerabile)**.** *Sia S un insieme, S è enumerabile se è possibile stabilire una corrispondenza biunivoca tra gli elementi di S e gli elementi dell'insieme dei numeri naturali $\mathbb{N}$.*

Enumerare un insieme significa quindi stabilire un primo, secondo, terzo, etc. elemento. Notare che, essendo $\mathbb{N}$ un insieme infinito, se un insieme S è enumerabile allora S è anche un insieme infinito.

Definizione 1.3 (Cardinalità insiemi enumerabili)**.** *La cardinalità degli insiemi enumerabili si indica con $\aleph_0$ (aleph zero).*

Sia S un insieme, allora:

- S è finito se $|S| < \aleph_0$,
- S è enumerabile se $|S| = |\mathbb{N}| = \aleph_0$.

In quanto informatici, siamo interessati agli insiemi enumerabili perché i dati in informatica, essendo rappresentati da numeri, sono enumerabili.

Definizione 1.4 (Insieme delle parti)**.** *Sia S un insieme, l'*insieme delle parti *o* insieme potenza *di S si indica con $\mathcal{P}(S)$ ed è l'insieme formato da tutti i sottoinsiemi di S.*

Esempio. Sia $S = \{\, 0, 1 \,\}$, allora $\mathcal{P}(S) = \big\{\, \varnothing, \{\, 0 \,\}, \{\, 1 \,\}, \{\, 0, 1 \,\} \,\big\}$.

Teorema 1.1 (Cantor)**.** *Sia S un insieme, allora*

$$|S| < |\mathcal{P}(S)|,$$

$$|\mathcal{P}(S)| = 2^{|S|},$$

$$|\mathbb{N}| = \aleph_0 < |\mathcal{P}(\mathbb{N})| = |\mathbb{R}|.$$

Lemma 1.1. *Sia I un insieme qualsiasi di indici, e sia $\{ S_i \}_{i \in I}$ una famiglia di insiemi indicizzata su I. Sia T un altro insieme. Se per ogni $i \in I$ abbiamo $S_i \subseteq T$, allora $\bigcup_{i \in I} S_i \subseteq T$.*

Dimostrazione. Dobbiamo mostrare che, preso un qualsiasi $x \in \bigcup_{i \in I} S_i$, abbiamo $x \in T$. Infatti, se $x \in \bigcup_{i \in I} S_i$, deve necessariamente esistere (almeno un) $j \in I$ tale che $x \in S_j$. Ma allora, poiché $S_j \subseteq T$, abbiamo $x \in T$. □

1.2 Principio di Induzione Matematica

Definizione induttiva. La definizione per induzione viene utilizzata quando si vuole definire un "qualcosa" Q che è parametrico sui naturali. La definizione è composta da:

1. *caso base*, $Q(0) = k_0$. Definisce il valore k_0 che assume Q nel caso base che spesso si indica con 0;
2. *passo induttivo*, $Q(n) = k_n \implies Q(n+1) = f(k_n)$. Definisce $Q(n+1)$ in termini di $Q(n)$ che è ipotizzato vero.

Esempio. Cos'è un numero naturale? Definiamo l'insieme $\mathbb{N}$ dei numeri naturali definendo in maniera induttiva gli elementi che vi fanno parte.

Caso base: $0 \in \mathbb{N}$.

Passo induttivo: $\forall n \in U : n \in \mathbb{N} \implies n + 1 \in \mathbb{N}$.

Nel caso base della definizione induttiva di $\mathbb{N}$ diciamo che è sempre vero che 0 è un numero naturale, il caso base non fa riferimento a nessun'altra nozione. Il passo induttivo è tipicamente in forma di implicazione: se un elemento n appartiene ad $\mathbb{N}$, allora anche il suo successore appartiene ad $\mathbb{N}$. Se conosciamo il valore per n possiamo ricavare il valore anche di $n + 1$, in altre parole, $n + 1$ è definito in termini di n.

Per definire le implicazioni nei linguaggi di programmazione si utilizza la *notazione semantica* in cui si utilizza una linea orizzontale per separare le *precondizioni* o *premesse* dalle *conclusioni*:

$$\text{Caso base: } \frac{}{0 \in \mathbb{N}}$$

$$\text{Passo induttivo: } \frac{n \in \mathbb{N}}{n + 1 \in \mathbb{N}}$$

Dimostrazione per induzione. Una dimostrazione per induzione si utilizza quando si vuole mostrare che una certa proprietà P vale per tutti i numeri naturali. Si procede mostrando che la proprietà è vera per il caso base 0. Poi si mostra che se la proprietà è vera per n allora è vera anche per $n + 1$.

$$\text{Caso base: } \frac{}{P(0)}$$

$$\text{Passo induttivo: } \frac{P(n)}{P(n+1)}$$

1. *Caso base*: $P(0)$, mostriamo che P vale nel caso base indicato con 0;
2. *Passo induttivo*:

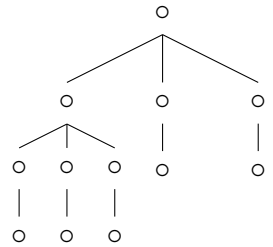
$$\underbrace{P(n)}_{antecedente} \implies \underbrace{P(n+1)}_{conseguente}.$$

   Il passo induttivo è tipicamente in forma di implicazione e la proposizione antecedente prende il nome di *ipotesi induttiva* proprio perché viene ipotizzata vera. A questo punto si deve mostrare che vale l'implicazione; ipotizzando vero l'antecedente è necessario mostrare che è vero anche il conseguente.

Quella vista fino ad ora viene detta *induzione matematica semplice*. Esistono anche *induzione matematica completa* ed *induzione strutturale*.

Induzione Matematica completa. L'induzione matematica completa ha una ipotesi induttiva più forte: dice che una certa proprietà $P(m)$ vale per m e anche per tutti i suoi predecessori, quindi da 0 a m compreso. Non è più potente dell'induzione semplice ma a volte è più conveniente da utilizzare.

Esempio. Dato il seguente albero, per dimostrare che la profondità dei suoi sottoalberi è ≤ 3 è necessaria l'induzione completa perché alcuni sottoalberi hanno profondità < 3.



Induzione strutturale. Il principio su cui si base l'induzione strutturale è il seguente: *per mostrare che una proprietà P è vera per tutte le espressioni, è sufficiente mostrare che è vera per tutte le espressioni atomiche ed è preservata da tutti i metodi utilizzati per generare nuove espressioni a partire da quelle esistenti.*

Proseguire nella lettura per esempi di induzione semplice, completa e strutturale.

2 Alfabeti e Linguaggi

2.1 Definizioni, Teoremi e Lemmi

Definizione 2.1 (Simbolo)**.** *Un simbolo è un'entità primitiva astratta non meglio definita.*

Esempio. Tra i simboli troviamo ad esempio: lettere, caratteri numerici, ideogrammi, bandiere per segnalazioni nautiche.

Notazione. Utilizzeremo a, b, c, d (prime lettere del nostro alfabeto) per indicare simboli.

Definizione 2.2 (Alfabeto)**.** *Un alfabeto (spesso indicato con Σ) è un insieme* finito *di simboli.*

La finitezza dell'alfabeto è necessaria per poter distinguere i simboli gli uni con gli altri in tempo finito e con una quantità finita di memoria cosa che non sarebbe possibile se l'alfabeto preso in considerazione fosse infinito.

Esempio. $\Sigma_{bin} = \{ 0, 1 \}, \Sigma_{\text{hexa}} = \{ 1, 2, \ldots, 9, A, B, \ldots, F \}$ sono alfabeti.

Definizione 2.3 (Definizione informale di stringa)**.** *Una stringa è una sequenza finita di simboli giustapposti (posti uno dietro l'altro).*

Esempio. Se a, b, c sono simboli, *abcba* è una stringa.

Osservazione 2.1 (Definizione formale di stringa)**.** *E' possibile definire formalmente una stringa di lunghezza $n \in \mathbb{N}$ per mezzo di una funzione* parziale $x : \{ 0, \ldots, n-1 \} \rightarrowtail \Sigma$ *definita come*

$$x(i) := \begin{cases} a_i, & \text{se } 0 \leq i < n, x = a_0 \cdots a_{n-1}; \\ \uparrow, & \text{altrimenti.} \end{cases}$$

Definizione 2.4 (Definizione informale di lunghezza di una stringa)**.** *Sia x una generica stringa, la sua lunghezza si indica con $|x|$ e corrisponde al numero di occorrenze dei simboli che la compongono. C'è una sola stringa di lunghezza 0, la stringa vuota ϵ.*

Esempio. Sia la stringa $x = aba$. $|x| = |aba| = 3$.

Osservazione 2.2 (Definizione formale di lunghezza di una stringa)**.** *E' possibile definire formalmente la lunghezza di una stringa per mezzo di una funzione totale. Sia x una stringa. $|x| : \Sigma^* \mapsto \mathbb{N}$ è definita come*

$$|x| := \begin{cases} 0, & \text{se } \forall i \in \mathbb{N} : x(i) = \uparrow; \\ 1 + \max\{\, i \in \mathbb{N} \mid x(i) = \downarrow \,\} & \text{altrimenti;} \end{cases}$$

Notare che come dominio è stato messo Σ^*. Dal momento che l'operazione "lunghezza di una stringa" prende come parametri stinghe, possiamo già intuire che Σ^* è un insieme di stringhe. Proseguire nella lettura per ulteriori approfondimenti.

Osservazione 2.3 (Definizione ricorsiva della lunghezza di una stringa)**.** *Sia $x \in \Sigma^*$, un'altra definizione formale (equivalente) per la funzione "lunghezza di una stringa" $|x| : \Sigma^* \mapsto \mathbb{N}$ è la seguente:*

$$|x| := \begin{cases} 0, & \text{se } x = \epsilon; \\ 1 + |y|, & \text{se } x = ay. \end{cases}$$

Definizione 2.5 (Prefisso, suffisso, sottostringa)**.** *Sia $x = a_1 \cdots a_n$ una stringa di lunghezza n:*

- *prefisso: $a_1 \cdots a_i$, con $1 \leq i \leq n$;*
- *suffisso: $a_i \cdots a_n$, con $1 \leq i \leq n$;*
- *sottostringa: $a_i \cdots a_j$, con $1 \leq i \leq j \leq n$;*
- *ϵ è prefisso, suffisso e sottostringa di ogni stringa, compresa di se stessa.*

Quando i prefissi o suffissi di una stringa non *sono uguali alla stringa stessa, vengono detti* propri*.*

Esempio. Sia *abc* una stringa, i suoi prefissi sono ϵ, *a*, *ab*, *abc*, mentre i suffissi sono ϵ, *c*, *bc*, *abc*,

Definizione 2.6 (Definizione informale di concatenazione)**.** *Siano x, y stringhe, la loro concatenazione si indica con $x \cdot y$ ed è la stringa ottenuta facendo seguire alla prima la seconda.*

Come accade per la moltiplicazione nell'artimetica, spesso il '$\cdot$' viene omesso e la concatenazione tra due stringhe x e y viene semplicemente indicata con xy.

Osservazione 2.4 (Definizione formale di concatenazione)**.** *Siano $x, y \in \Sigma^*$. E' possibile definire formalmente la concatenazione tra stringhe xy per mezzo di una funzione parziale $(xy) : \{\ 0, \ldots, |x+y|-1\} \rightarrowtail \mathbb{N}$ definita come*

$$(xy)(i) := \begin{cases} x(i), & \text{se } 0 \leq i < |x|; \\ y(i-|x|), & \text{se } |x| \leq i < |xy|; \\ \uparrow, & \text{altrimenti.} \end{cases}$$

Osservazione 2.5 (Definizione induttiva di concatenazione)**.** *Siano $x, y \in \Sigma^*$, è possibile definire la concatenazione xy per induzione strutturale su y.*

$$\textit{Caso base: } \frac{y = \epsilon}{x\epsilon = x}$$

$$\textit{Passo induttivo: } \frac{y = wa}{x(wa) = (xw)a}$$

Lemma 2.1 (Proprietà associativa della concatenazione)**.** *Siano x, y, z tre stringhe, allora*

$$(xy)z = x(yz).$$

Dimostrazione. Siano $x, y, z \in \Sigma^*$. Mostreremo che $(xy)z = x(yz)$ per induzione sulla strutturale su z.

Caso base: se $z = \epsilon$ abbiamo che

$$\begin{aligned} (xy)z &= (xy)\epsilon && [z = \epsilon] \\ &= xy && [\text{def. di concat.}] \\ &= x(y\epsilon) && [\text{def. di concat.}]. \end{aligned}$$

Passo induttivo: se $z = wa$ abbiamo che

$$\begin{aligned} (xy)z &= (xy)(wa) && [z = wa] \\ &= \big((xy)w\big)a && [\text{def. di concat.}] \\ &= \big(x(yw)\big)a && [\text{ip. ind.}] \\ &= x\big((yw)a\big) && [\text{def. di concat.}] \\ &= x\big(y(wa)\big) && [\text{def. di concat.}] \\ &= x(yz) && [z = wa]. \end{aligned}$$

□

Dalla validità del lemma segue che

$$(xy)z = x(yz) = xyz.$$

Lemma 2.2 (Identità della concatenazione)**.** *Sia x una stringa, allora*

$$\epsilon x = x = x\epsilon.$$

Dimostrazione. Sia x una generica stringa e $i \in \mathbb{N}$. Mostreremo che $(\epsilon x)(i) = x(i) = (x\epsilon)(i)$.

$$\begin{aligned}
(\epsilon x)(i) &= \begin{cases} \epsilon(i), & \text{se } 0 \leq i < |\epsilon|; \\ x(i - |\epsilon|), & \text{se } |\epsilon| \leq i < |\epsilon| + |x|; \\ \uparrow, & \text{altrimenti.} \end{cases} && \text{[def. di concat.]} \\
&= \begin{cases} \epsilon(i), & \text{se } 0 \leq i < 0; \\ x(i - 0), & \text{se } 0 \leq i < 0 + |x|; \\ \uparrow, & \text{altrimenti.} \end{cases} && [|\epsilon| = 0] \\
&= \begin{cases} x(i), & \text{se } 0 \leq i < |x|; \\ \uparrow, & \text{altrimenti.} \end{cases} && [\nexists i \in \mathbb{N} \,.\, 0 \leq i < 0] \\
&= x(i). && \text{[def. di stringa].}
\end{aligned}$$

$$\begin{aligned}
(x\epsilon)(i) &= \begin{cases} x(i), & \text{se } 0 \leq i < |x|; \\ \epsilon(i - |x|), & \text{se } |x| \leq i < |x| + |\epsilon|; \\ \uparrow, & \text{altrimenti.} \end{cases} && \text{[def. di concat.]} \\
&= \begin{cases} x(i), & \text{se } 0 \leq i < |x|; \\ \uparrow, & \text{altrimenti.} \end{cases} && [\forall i \in \mathbb{N} : \epsilon[i] = \uparrow] \\
&= x(i) && \text{[def. di stringa].}
\end{aligned}$$

□

Definizione 2.7 (Stringhe su un alfabeto)**.** *Sia Σ un alfabeto, le stringhe su Σ sono le stringhe di lunghezza $n \in \mathbb{N}$ (arbitraria ma finita) formate dalla giustapposizione dei simboli che compaiono in Σ, dunque una generica stringa x su Σ è definita come:*

$$x = a_1 a_2 \cdots a_n, \ \textit{con } a_i \in \Sigma, n \geq 0.$$

Definizione 2.8 (Insieme di tutte le stringhe su un alfabeto)**.** *Sia Σ un alfabeto, l'insieme di tutte le stringhe su Σ, denotato con Σ^* è dunque:*

$$\Sigma^* := \{\, a_1 a_2 \cdots a_n \mid a_i \in \Sigma, n \geq 0 \,\} = \bigcup_{n \in \mathbb{N}} \{\, 0, \ldots, n-1 \,\} \rightarrowtail \Sigma.$$

Siano Σ un alfabeto e $n \in \mathbb{N}$, è possibile dare una definizione di forma induttiva per Σ^n:

$$\Sigma^n := \begin{cases} \{\, \epsilon \,\}, & \text{se } n = 0; \\ \{\, a \cdot x \mid a \in \Sigma, x \in \Sigma^{n-1} \,\}, & \text{altrimenti.} \end{cases}$$

Osservazione 2.6. *Notare che se $\Sigma \neq \varnothing$, allora Σ^n è l'insieme formato da tutte le stringhe di lunghezza n su Σ.*

Dimostrazione. Vogliamo mostrare che:

$$\forall n \in \mathbb{N}, \forall x \in \Sigma^n : |x| = n.$$

Procediamo quindi per induzione sull'esponente n di Σ e di conseguenza sulla lunghezza delle stringhe, con caso base $n = 0$ e passo induttivo $n = m + 1$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: Se $n = 0$ dobbiamo mostrare che $|x| = 0$.

$$\begin{aligned} \Sigma^n &= \Sigma^0 && [n = 0] \\ &= \{\, \epsilon \,\} && [\text{def. di } \cdot^0]. \end{aligned}$$

Sia $x \in \Sigma^n$, allora $x = \epsilon, |x| = 0 = n$ ✓.

Passso induttivo:

$$\forall m \in \mathbb{N}, \forall x \in \Sigma^m : |x| = m \overset{?}{\Longrightarrow} \forall y \in \Sigma^{m+1} : |y| = m + 1.$$

Se $n = m + 1$ abbiamo che

$$\begin{aligned} \Sigma^n &= \Sigma^{m+1} && [n = m + 1] \\ &= \{\, ax \mid a \in \Sigma, x \in \Sigma^m \,\} && [\text{def. di } \cdot^n]. \end{aligned}$$

Sia $y \in \Sigma^{m+1}$, dunque $y = aw$ con $a \in \Sigma, w \in \Sigma^m$.

$$\begin{aligned} |y| &= |a| + |w| && [y = aw] \\ &= 1 + |w| && [a \in \Sigma] \\ &= 1 + m \checkmark && [w \in \Sigma^m, \text{ip. ind.}]. \end{aligned}$$

□

Secondo quanto appena visto, è possibile dare un'altra definizione per Σ^*:

$$\Sigma^* := \bigcup_{n \in \mathbb{N}} \Sigma^n.$$

Esempio. Siano $\Sigma = \{ 0 \}, n = 3$, allora

$$\Sigma^n = \{ 000 \}, \Sigma^* = \{ \epsilon, 0, 00, 000, 0000, \dots \}.$$

Siano $\Sigma = \{ 0, 1 \}, n = 2$, allora

$$\Sigma^n = \{ 00, 01, 10, 11 \}, \Sigma^* = \{ \epsilon, 0, 1, 00, 01, 11, 000, \dots \}.$$

Definizione 2.9 (Linguaggio formale)**.** *Sia Σ un alfabeto, un linguaggio formale (spesso indicato con L), è un* insieme *formato dalle stringhe su Σ, quindi vale che:*

$$L \subseteq \Sigma^*.$$

Notare che:

- Σ^* è esso stesso un linguaggio, il linguaggio formato da *tutte* le stringhe su Σ;
- essendo $\varnothing$ sottoinsieme di qualsiasi altro insieme, $\varnothing \subseteq \Sigma^*$ dunque $\varnothing$ è un linguaggio su qualsiasi alfabeto;
- essendo ϵ la stringa nulla su qualsiasi alfabeto Σ, $\{ \epsilon \} \subseteq \Sigma^*$ dunque $\{ \epsilon \}$ è un linguaggio su qualsiasi alfabeto.

Osservazione 2.7. *Sia Σ un alfabeto. Sia $L \subseteq \Sigma^*$ il linguaggio formato dall'insieme di tutte le stringhe di lunghezza 1 su Σ, allora*

$$L \cong \Sigma, \ \textit{(L e Σ sono} \text{ isomorfi}\textit{)}$$

ed è possibile dare una definizione induttiva di Σ^n equivalente a quella già vista in precedenza:

$$\textit{Caso base: } \frac{}{\Sigma^0 = \{ \epsilon \}}$$

$$\textit{Passo induttivo: } \frac{\Sigma^n = L}{\Sigma^{n+1} = \Sigma \cdot L}$$

da cui segue da definizione di concatenazione di due linguaggi L, M

$$L \cdot M = \{ x \cdot y \mid x \in L, y \in M \}.$$

2.2 Esercizi

Esercizio 2.1

Definire i predicati $\text{prefix}(x, y), \text{suffix}(x, y), \text{substr}(x, y)$ su stringhe.

Soluzione.

$$\text{prefix}(x,y) \begin{cases} \textbf{true}, & \text{se } x = \epsilon; \\ \textbf{true}, & \text{se } \forall i, 0 \leq i < |x| : x(i) = y(i); \\ \textbf{false}, & \text{altrimenti.} \end{cases}$$

$$\text{suffix}(x,y) \begin{cases} \textbf{true}, & \text{se } x = \epsilon; \\ \textbf{true}, & \text{se } \forall i, 0 \leq i < |x| : x(i) = y(i+k), k = |y| - |x|; \\ \textbf{false}, & \text{altrimenti.} \end{cases}$$

$$\text{substr}(x,y) \begin{cases} \textbf{true}, & \text{se } x = \epsilon; \\ \textbf{true}, & \text{se } \exists k, 0 \leq k < |y| \;.\; \forall i, 0 \leq i < |x| : x(i) = y(i+k); \\ \textbf{false}, & \text{altrimenti.} \end{cases}$$

Esercizio 2.2

Sia x una stringa con $|x| = n$. Qual'è il numero di prefissi, suffissi e sottostringhe di x? Quanti sono i prefissi dei suoi suffissi? E i suffissi dei suoi prefissi?

Soluzione. Scegliamo una stringa di esempio $x = abc$ ed elenchiamo i prefissi:

$$\epsilon, a, ab, abc.$$

Con $|x| = 3$ otteniamo 4 prefissi, con una generica stringa s di lunghezza n otteniamo $|s| + 1$ prefissi. Ragionamento analogo si applica ai suffissi. Per quanto riguarda le sottostringhe dobbiamo ragionare un po' di più, elenchiamo le sottostringhe:

- ϵ (1 sottostringa di lunghezza 0);
- abc (1 sottostringa di lunghezza 3);

- ab, bc (2 sottostringhe di lunghezza 2);
- a, b, c (3 sottostringhe di lunghezza 1).

Notiamo che, fatta eccezione per la sottostringa ϵ, il numero di sottostringhe trovate diminuisce quando la lunghezza delle sottostringhe cercate aumenta. Il numero di sottostringhe di una stringa di lunghezza 3 é $1 + 1 + 2 + 3$. Generalizzando ad una stringa x di lunghezza n, il numero delle sue sottostringhe è la somma dei numeri naturali da 1 a n, incrementata di 1, ovvero, se chiamiamo substr(x) l'insieme formato da tutte e sole le sottostringhe di x abbiamo:

$$|\text{substr}(x)| = 1 + \sum_{i=1}^{|x|} i = 1 + \frac{|x|(|x|+1)}{2} \qquad \text{[Gauss]}.$$

Questa è la nostra ipotesi, proviamo a dimostrarne la validità con una dimostrazione formale per induzione.

Dimostrazione. Vogliamo mostrare che

$$\forall n \in \mathbb{N}, \forall x \in \Sigma^n : |\text{substr}(x)| = 1 + \sum_{i=1}^{n} i$$

Dimostreremo per induzione sulla lunghezza della stringa n con caso base $n = 0$ e passo induttivo $n = m + 1$ con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 0 = |x|$ abbiamo $x = \epsilon$. La stringa ϵ ha solo se stessa come sottostringa quindi

$$\begin{aligned} |\text{substr}(x)| &= |\text{substr}(\epsilon)| && [n = 0 = |x|] \\ &= 1 \\ &= 1 + \sum_{i=1}^{|x|} i && [|x| = n = 0]. \end{aligned}$$

Passo induttivo:

$$\forall m \in \mathbb{N}, \forall x \in \Sigma^m : |\text{subs}(x)| = 1 + \sum_{i=1}^{m} i \overset{?}{\Longrightarrow} \forall y \in \Sigma^{m+1} : |\text{subs}(y)| = 1 + \sum_{i=1}^{m+1} i.$$

Se $n = m + 1 = |y|$ abbiamo $y = wa$, con $|w| = m$ e vale che

$$\begin{aligned}
|\text{substr}(y)| &= |\text{substr}(wa)| && [y = wa] \\
&= |\text{substr}(w)| + |\text{suffix}(wa)| - 1 && [-1 \text{ rimuove } \epsilon \text{ duplicato}] \\
&= 1 + \sum_{i=1}^{m} i + |\text{suffix}(wa)| - 1 && [|w| = m, \text{ ip. ind.}] \\
&= 1 + \frac{m(m+1)}{2} + |\text{suffix}(wa)| - 1 && [\text{Gauss}] \\
&= 1 + \frac{m(m+1)}{2} + \big((m+1)+1\big) - 1 && [|wa| = m + 1] \\
&= 1 + \frac{m(m+1)}{2} + m + 1 \\
&= 1 + \frac{m(m+1) + 2m + 2}{2} && [m.c.m] \\
&= 1 + \frac{m^2 + m + 2m + 2}{2} \\
&= 1 + \frac{(m+1)(m+2)}{2} \\
&= 1 + \sum_{i=1}^{m+1} i && [\text{Gauss}].
\end{aligned}$$

□

Quanti sono i suffissi dei prefissi e i prefissi dei suffissi? Prendiamo una stringa di esempio $x = abc$ con lunghezza $n = 3$. Elenchiamo i suoi prefissi ϵ, a, ab, abc con il relativo numero di suffissi (che sappiamo essere $|x| + 1$):

- $abc : 3 + 1$ suffissi;
- $ab : 2 + 1$ suffissi;
- $a : 1 + 1$ suffissi;
- $\epsilon : 0 + 1$ suffissi.

Sommando i "+1" di ogni riga otteniamo $|x| + 1$ perchè il numero di righe è proprio il numero di prefissi di x che è $|x| + 1$. Ciò che rimane è una somma

da 0 a $|x|$. Quindi, il numero di suffissi dei prefissi di una stringa x é

$$|x| + 1 + \sum_{i=1}^{|x|} i = \sum_{i=1}^{|x|+1} i.$$

Dimostrare formalmente per esercizio.

Esercizio 2.3

Sia $\Sigma \neq \varnothing$ un alfabeto, qual'è la cardinalità di Σ^n? Dimostrare la propria affermazione.

Soluzione. Lavoriamo ad esempio con le stringhe binarie, quindi sia $\Sigma = \{\ 0, 1,\ \}$ il nostro alfabeto di riferimento, quindi $|\Sigma| = 2$. Con n bit disponibili (quindi stringhe lunghe n) possiamo contare da 0 fino a $2^n - 1$, un totale di 2^n elementi.

Preso un generico $\Sigma \neq \varnothing$, vediamo la lunghezza n delle stringhe come i bit disponibili, quindi con $|\Sigma|$ simboli possiamo ottenere $|\Sigma|^n$ stringhe di lunghezza n. Quindi ipotizziamo che:

$$|\Sigma^n| = |\Sigma|^n,$$

proviamo a dimostrarne la validità con una dimostrazione formale per induzione.

Dimostrazione. Vogliamo mostrare che $\forall n \in \mathbb{N} : |\Sigma^n| = |\Sigma|^n$. Procederemo per induzione su n con caso base $n = 0$ e passo induttivo $n = m + 1$ con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 0$ abbiamo che

$$\begin{aligned}
|\Sigma^n| &= |\Sigma^0| && [n = 0] \\
&= |\{\ \epsilon\ \}| && [\text{def. di } \cdot^0] \\
&= 1 \\
&= |\Sigma|^0 && [\forall i \in \mathbb{N}, i^0 = 1] \\
&= |\Sigma|^n && [n = 0].
\end{aligned}$$

Passo induttivo:

$$\forall m \in \mathbb{N} : |\Sigma^m| = |\Sigma|^m \overset{?}{\Longrightarrow} |\Sigma^{m+1}| = |\Sigma|^{m+1}.$$

se $n = m + 1$ abbiamo che

$$\begin{aligned}
|\Sigma^n| &= |\Sigma^{m+1}| && [n = m + 1] \\
&= |\Sigma \cdot \Sigma^m| && [\text{def. di } \cdot^n] \\
&= |\Sigma||\Sigma^m| && [\text{def. di concat.}] \\
&= |\Sigma||\Sigma|^m && [\text{ip. ind.}] \\
&= |\Sigma|^{m+1} && [\text{prop. potenze}].
\end{aligned}$$

□

Esercizio 2.4

Trovare gli alfabeti Σ per i quali l'insieme Σ^* è esso stesso un alfabeto.

Soluzione. Per definizione, un alfabeto è un insieme finito di simboli. Σ^* è l'insieme formato da tutte le stringhe di lunghezza arbitraria ma finita su Σ. Per ogni $\Sigma \neq \varnothing$, l'insieme Σ^* essendo infinito non può essere un alfabeto, l'unica eccezione è proprio $\Sigma = \varnothing \implies \Sigma^* = \{ \epsilon \}$.

Dimostrazione. Utilizziamo le definizioni viste in questo capitolo per costruire una catena di relazioni (uguaglianze in questo caso) che parte da Σ^* ed arriva a $\{ \epsilon \}$, ricordandoci che abbiamo posto $\Sigma = \varnothing$ perché ad un certo punto tornerà utile:

$$\begin{aligned}
\Sigma^* &= \bigcup_{i \in \mathbb{N}} \Sigma^i && [\text{def. di } \cdot^*] \\
&= \Sigma^0 \cup \bigcup_{i \geq 1} \Sigma^i && [0 \in \mathbb{N}] \\
&= \{ \epsilon \} \cup \bigcup_{i \geq 1} \Sigma^i && [\text{def. di } \cdot^0] \\
&= \{ \epsilon \} \cup \bigcup_{i \geq 1} \{ ax \mid a \in \Sigma, x \in \Sigma^{i-1} \} && [\text{def. di } \cdot^i] \\
&= \{ \epsilon \} \cup \bigcup_{i \geq 1} \{ ax \mid a \in \varnothing, x \in \Sigma^{i-1} \} && [\Sigma = \varnothing] \\
&= \{ \epsilon \} \cup \varnothing && [\nexists a \in \varnothing] \\
&= \{ \epsilon \} && [\text{def. di } \cup].
\end{aligned}$$

□

Esercizio 2.5

Una definizione non induttiva di concatenazione è la seguente: siano $i, j \in \mathbb{N}$, siano $x = a_1 \cdots a_i$ e $y = b_1 ... b_j$ due stringhe su Σ^* allora $xy = a_1 ... a_i b_1 ... b_j$. Mostrare l'equivalenza con la definizione induttiva.

Dimostrazione. Mostriamo per induzione sulla lunghezza n della stringa y applicando la definizione induttiva, con caso base $n = 0$ e passo induttivo $n = m + 1$ con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 0$ abbiamo

$$\begin{aligned} xy &= x\epsilon && [|y| = n = 0] \\ &= x && [\text{def. ind. concat.}] \\ &= xb_1 \cdots b_n && [|y| = n = 0]. \end{aligned}$$

Passo induttivo:

$$\forall x, y \in \Sigma^*, |y| = m : xy = xb_1 \cdots b_m \overset{?}{\Longrightarrow} x(yb) = xb_1 \cdots b_m b.$$

Se $n = m + 1$ abbiamo $y = wb$ con $|w| = m$ e vale che

$$\begin{aligned} xy &= x(wb) && [|y| = n = m + 1] \\ &= (xw)b && [\text{def. ind. concat.}] \\ &= xb_1 \cdots b_m b && [\text{ip. ind.}] \\ &= xb_1 \cdots b_{m+1}. \end{aligned}$$

□

Esercizio 2.6

Verificare che se $\Sigma \neq \varnothing$, allora Σ^* è numerabile.

Soluzione. Un insieme è enumerabile quando è possibile stabilire una corrispondenza biunivoca tra gli elementi dell'insieme e l'insieme dei numeri naturali. Scegliamo un alfabeto di esempio su cui lavorare, sia $\Sigma = \{ a, b, c \}$. Σ^* è l'insieme formato da tutte le stringhe su Σ, ovvero $\Sigma^* = \{ \epsilon, a, b, c, aa, ab, ac, ba, bb, \cdots \}$. Ciò che vorremo fare è associare ad ogni elemento di Σ^* un numero naturale. Iniziamo enumerando le stringhe di lunghezza 0 (soltanto ϵ), procediamo poi con le stringhe di lunghezza 1, 2, etc. (se le stringhe lunghezza n sono più di una, occorre ordinarle secondo un qualche criterio, noi utilizzeremo l'ordine alfanumerico):

- $\epsilon \rightsquigarrow 0$;
- $a \rightsquigarrow 1, b \rightsquigarrow 2, c \rightsquigarrow 3$;
- $aa \rightsquigarrow 4, ab \rightsquigarrow 5, ac \rightsquigarrow 6, \cdots, cc \rightsquigarrow 12$;
- $\cdots$

Cerchiamo di formalizzare questo processo: $\forall i \in \mathbb{N}$, ordiniamo le stringhe $x \in \Sigma^i$ per poi associare alla stringa in posizione j-esima dell'ordinamento il seguente numero naturale:

$$\sum_{k \in \mathbb{N}, k=0}^{i-1} |\Sigma|^k + j - 1.$$

$\sum_{k=0}^{i-1} |\Sigma|^k$ corrisponde all'ultimo numero naturale associato al "passo" $i-1$ perché $|\Sigma|^k$ corrisponde al numero di elementi nell'insieme Σ^k, facendo la somma dei $|\Sigma|^k$, con $0 \leq k \leq i-1$, otteniamo il numero totale delle stringhe di lunghezza compresa tra 0 e $i-1$. Il "-1" è perché vogliamo iniziare a contare da 0.

Esempio. Consideriamo $\Sigma = \{ a, b, c \}, k \in \mathbb{N}, \forall i \in \mathbb{N}$:

- $i = 0$,
  $\Sigma^i = \Sigma^0 = \{ \epsilon \}$:
  - $j = 1, \epsilon \rightsquigarrow \sum_{k=0}^{i-1} |\Sigma|^k + j - 1 = \sum_{k=0}^{-1} |\Sigma|^k + j - 1 = j - 1 = 0.$
- $i = 1$,
  $\Sigma^i = \Sigma = \{ a, b, c \}$:

- $j = 1, a \rightsquigarrow \sum_{k=0}^{i-1} |\Sigma|^k + j - 1 = |\Sigma|^0 + j - 1 = 1$;
- $j = 2, b \rightsquigarrow |\Sigma|^0 + j - 1 = 2$;
- $j = 3, c \rightsquigarrow |\Sigma|^0 + j - 1 = 3$.

- $i = 2$,
  $\Sigma^i = \Sigma^2 = \{ aa, ab, ac, ba, bb, bc, ca, cb, cc \}$:
  - $j = 1, aa \rightsquigarrow \sum_{k=0}^{i-1} |\Sigma|^k + j - 1 = |\Sigma|^0 + |\Sigma| + j - 1 = 4$;
  - $\ldots$
  - $j = 9, cc \rightsquigarrow |\Sigma^0| + |\Sigma| + j - 1 = 12$;

- $i = 3$
  $\ldots$

3 Automi

3.1 Rappresentazione dei Linguaggi

La rappresentazione finita di linguaggi infiniti è afrontabile da tre punti di vista:

1. *riconoscitivo-analitico*, in cui un linguaggio è visto come un insieme di stringhe accettate da strutture finite come gli automi;

2. *algebrico*, in cui il linguaggio è rappresentato da un'espressione algebrica o è la soluzione di un sistema di relazioni algebriche;

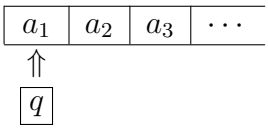
3. *generativo-sintetico*, in cui il linguaggio è visto come l'insieme delle stringhe *generate* da strutture finite dette *grammatiche*.

Gli *automi*, che ora vedremo, ricadono nell'approccio di tipo riconoscitivo-analitico.

Un automa è un oggetto composto da:

- una *t*estina $\Uparrow$ che legge, da sinistra verso destra, simboli posti su un nastro di lunghezza illimitata. A seguito della lettura di un simbolo, la testina si sposta di una posizione verso destra per leggere il simbolo successivo;

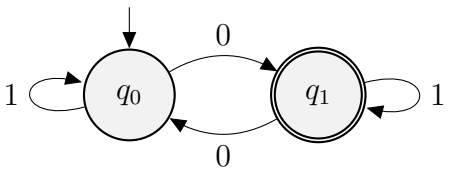
- uno *stato* interno $\boldsymbol{q}$ che a seguito della lettura di un simbolo può modificarsi o rimanere lo stesso, a seconda di come è stato definito il "comportamento" dell'automa.



La lettura termina quando sul nastro non rimangono più simboli da leggere, ovvero, sul nastro giace la stringa vuota ϵ. A questo punto, l'automa si può trovare in uno stato di accettazione o di rifiuto della stringa che, un simbolo alla volta, è stata letta per intero.

3.2 DFA: automi a stati finiti deterministici

Descrizione informale di un automa a stati finiti deterministico M tramite *grafo di transizione*:



Descrizione dell'automa a stati finiti deterministico M tramite *matrice di transizione*:

	0	**1**
$\boldsymbol{q_0}$	q_1	q_0
$\boldsymbol{\dot{q}_1}$	q_0	q_1

- nella prima colonna troviamo i diversi stati q_i dell'automa: per definizione q_0 è lo stato iniziale, mentre lo stato finale (che può cambiare di automa in automa) viene identificato (in questo documento) ponendo un puntino sopra il nome dello stato ($\dot{q}_1$);
- nella prima riga troviamo i simboli in ingresso $a \in \Sigma$.

In questo caso particolare quindi, se lo stato attuale dell'automa è q e il prossimo simbolo da leggere sul nastro è a, lo stato in cui si troverà l'automa dopo la lettura del simbolo a è indicato nella cella $[q, a]$. La matrice di transizione (ed equivalentemente il grafo di transizione corrispondente), possono essere visti come la definizione di una funzione, che prende il nome di *funzione di transizione* e si indica con δ.

La funzione di transizione δ quindi ha:

- due argomenti in ingresso:
  - q, lo stato attuale del DFA;
  - a, il prossimo simbolo da leggere,
- uno ed un solo stato in uscita q'.

Il fatto che la funzione di transizione δ restituisce un solo ed unico stato in uscita è la motivazione dell'aggettivo *deterministico* nel nome dell'automa che descrive; infatti, per ogni coppia $\langle q, a \rangle$ in ingresso alla funzione di transizione δ, è sempre possibile determinare quale sarà il q' che si otterrà in uscita.

Notazione. Utilizzeremo la seguente notazione:

- p, q, r, s, t per indicare stati degli automi;
- P, Q, R, S, T per indicare insiemi di stati.
- $\overbrace{aa\cdots a}^{n} = a^n$ indica la stringa composta dalla giustapposizione del simbolo a con se stesso n volte;
- $a_1 a_2 \cdots a_n$ indica la stringa composta dalla giustapposizione di simboli a_i, anche diversi tra loro;
- $\overbrace{xx\cdots x}^{n} = x^n$ indica la stringa composta dalla concatenazione della stringa x con se stessa n volte;
- $x_1 x_2 \cdots x_n$ indica la stringa composta dalla concatenazione delle stringhe x_i, anche diverse tra loro.

3.2.1 Definizioni, Teoremi e Lemmi

Definizione 3.1 (DFA: automa a stati finiti deterministico)**.** *Un DFA è una quintupla $\langle Q, \Sigma, \delta, q_0, F \rangle$, dove:*

- $\boldsymbol{Q}$ *è un insieme finito di stati;*
- $\boldsymbol{\Sigma}$ *è l'alfabeto di input;*
- $\boldsymbol{\delta} : Q \times \Sigma \mapsto Q$ *è la funzione di transizione;*
- $\boldsymbol{q_0} \in Q$ *è lo stato iniziale;*
- $\boldsymbol{F} \subseteq Q$ *è l'insieme degli stati finali (o stati accettanti).*

Per definizione, una stringa è una giustapposizione di simboli, ciò che vedremo ora è una nuova funzione che applica più volte la δ per leggere intere stringhe dal nastro di input.

Definizione 3.2 (Funzione di transizione su stringhe)**.** *Sia M il DFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La funzione di transizione su stringhe $\hat{\delta} : Q \times \Sigma^* \rightarrowtail Q$ è comunemente definita come:*

$$\hat{\delta}(q, x) := \begin{cases} q, & \text{se } x = \epsilon; \\ \delta\big(\hat{\delta}(q, y), a\big), & \text{se } x = ya. \end{cases}$$

Osservazione 3.1. *Nella definizione di $\hat{\delta}$ vi è un elemento di arbitrarietà. Si può infatti dimostrare che, $\hat{\delta}$ si può sostituire con la funzione $\check{\delta}$ così definita:*

$$\check{\delta}(q, x) := \begin{cases} q, & \text{se } x = \epsilon; \\ \check{\delta}\big(\delta(q, a), y\big), & \text{se } x = ay. \end{cases}$$

Dimostrazione. Sia M il DFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Dimostriamo che, per ogni $x \in \Sigma^*$ e ogni $q \in Q$, abbiamo $\hat{\delta}(q, x) = \check{\delta}(q, x)$. Sia q arbitrario. Per il caso $x = \epsilon$ abbiamo, banalmente

$$\hat{\delta}(q, \epsilon) = q = \check{\delta}(q, \epsilon).$$

Sia ora $x \neq \epsilon$, $x = a_1 \dots a_n$. Dimostreremo per induzione matematica su lunghezza $n \geq 1$ della stringa, con caso base $n = 1$ e passo induttivo $n = m + 1$ con $m \geq 1$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva:

$$\hat{\delta}(q, a_1 \cdots a_m) = \underbrace{\delta\big(\cdots \delta(}_{m} q \underbrace{, a_1) \cdots a_m\big)}_{m} = \check{\delta}(q, a_1 \cdots a_m). \tag{1}$$

Caso base: con $n = 1$ abbiamo

$$\hat{\delta}(q, a_1) = \delta\big(\hat{\delta}(q, \epsilon), a_1\big) = \delta(q, a_1) = \check{\delta}\big(\delta(q, a_1), \epsilon\big) = \check{\delta}(q, a_1).$$

Passo inuttivo: con $n = m + 1$ abbiamo

$$\begin{aligned} \hat{\delta}(q, a_1 \cdots a_m a_{m+1}) &= \delta\big(\hat{\delta}(q, a_1 \cdots a_m), a_{m+1}\big) && \text{[def. di } \hat{\delta}\text{]} \\ &= \delta\big(\underbrace{\delta\big(\cdots \delta(}_{m} q \underbrace{, a_1) \cdots a_m\big)}_{m}, a_{m+1}\big) && \text{[ip. ind. (1)]} \\ &= \underbrace{\delta\big(\cdots \delta(}_{m+1} q \underbrace{, a_1) \cdots a_{m+1}\big)}_{m+1} && \text{[raggruppando].} \end{aligned}$$

Analogamente:

$$\begin{aligned}
\check{\delta}(q, a_1 a_2 \cdots a_{m+1}) &= \check{\delta}\big(\delta(q, a_1), a_2 \cdots a_{m+1}\big) && \text{[def. di } \check{\delta}\text{]} \\
&= \underbrace{\delta\big(\cdots \delta(}_{m}\, \delta(q, a_1)\, \underbrace{, a_2) \cdots a_{m+1}\big)}_{m} && \text{[ip. ind. (1)]} \\
&= \underbrace{\delta\big(\cdots \delta(}_{m+1}\, q\, \underbrace{, a_1) \cdots a_{m+1}\big)}_{m+1} && \text{[raggruppando].}
\end{aligned}$$

□

Definizione 3.3 (Accettazione di una stringa)**.** *Sia M il DFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La stringa x è* accettata *da M se e solo se:*

$$\hat{\delta}(q_0, x) \in F.$$

Una stringa x quindi è accettata da un DFA M se, partendo dallo stato iniziale q_0 e leggendo x, il DFA termina in uno degli stati accettanti.

Se durante il processo di lettura viene letto un simbolo che non sta in Σ, la lettura termina immediatamente con il DFA in uno stato di "errore" che ha come conseguenza il rifiuto della stringa.

Definizione 3.4 (Linguaggio accettato da un DFA)**.** *Sia M il DFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$, il linguaggio* accettato *da M si indica con $L(M)$ ed è l'insieme di tutte le stringhe accettate da M, ovvero:*

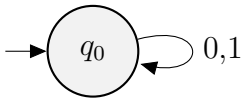
$$L(M) := \{\, x \in \Sigma^* \mid \hat{\delta}(q_0, x) \in F \,\}.$$

Teorema 3.1 (Linguaggio regolare)**.** *Un linguaggio L è regolare se esiste almeno un DFA M che accetta L, ovvero:*

$$L = L(M).$$

Esempio. DFA $M = \langle Q = \{\, q_0 \,\}, \Sigma = \{\, 0, 1 \,\}, \delta, q_0, F = \varnothing\rangle$ che accetta il linguaggio (regolare) $\varnothing$ ovvero "nessuna stringa è accettata":

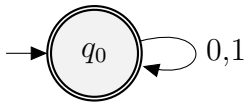
$$L(M) = \varnothing$$



	0	**1**
$\boldsymbol{q_0}$	q_0	q_0

DFA $M' = \langle Q = \{ q_0 \}, \Sigma = \{ 0, 1 \}, \delta, q_0, F = \{ q_0 \} \rangle$ che accetta il linguaggio (regolare) Σ^* ovvero "tutte le stringhe su Σ sono accettate":

$$L(M') = \Sigma^*$$



	0	**1**
$\dot{\boldsymbol{q}}_{\boldsymbol{0}}$	q_0	q_0

Lemma 3.1 (Proprietà funzione di transizione su stringhe)**.** *Sia M il DFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F \rangle$. Siano x, y due stringhe e $q \in Q$ uno stato, allora*

$$\hat{\delta}(q, xy) = \hat{\delta}\big(\hat{\delta}(q, x), y\big)$$

Dimostrazione. Vogliamo mostrare che $\forall x, y \in \Sigma^*, \forall q \in Q$:

$$\hat{\delta}(q, xy) = \hat{\delta}\big(\hat{\delta}(q, x), y\big).$$

Dimostriamo per induzione sulla lunghezza n della stringa y, con caso base $n = 0$ e passo induttivo $n = m + 1$, con $m \in \mathbb{N}$; pertanto nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 0$ abbiamo

$$\begin{aligned}
\hat{\delta}(q, xy) &= \hat{\delta}(q, x\epsilon) && [|y| = 0] \\
&= \hat{\delta}(q, x) && [\text{Lemma 2.2}] \\
&= \hat{\delta}\big(\hat{\delta}(q, x), \epsilon\big) && [\text{def. di } \hat{\delta}] \\
&= \hat{\delta}\big(\hat{\delta}(q, x), y\big) && [|y| = 0].
\end{aligned}$$

Passo induttivo: se $n = m + 1, y = wa$ abbiamo

$$\begin{aligned}
\hat{\delta}(q, xy) &= \hat{\delta}(q, xwa) && [y = wa] \\
&= \delta\big(\hat{\delta}(q, xw), a\big) && [\text{def. di } \hat{\delta}] \\
&= \delta\big(\hat{\delta}\big(\hat{\delta}(q, x), w\big), a\big) && [\text{ip. ind.}] \\
&= \hat{\delta}\big(\hat{\delta}(q, x), wa\big) && [\text{def. di } \hat{\delta}] \\
&= \hat{\delta}\big(\hat{\delta}(q, x), y\big) && [y = wa].
\end{aligned}$$

□

3.2.2 Esercizi

Esercizio 3.1

Siano xz, yz due stringhe e $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ un DFA, dimostrare che vale:

$$\hat{\delta}(q_0, x) = \hat{\delta}(q_0, y) \Longrightarrow \hat{\delta}(q_0, xz) = \hat{\delta}(q_0, yz).$$

Soluzione.

$$\begin{aligned}
\hat{\delta}(q_0, xz) &= \hat{\delta}\big(\hat{\delta}(q_0, x), z\big) && [\text{Lemma 3.1}] \\
&= \hat{\delta}\big(\hat{\delta}(q_0, y), z\big) && [\hat{\delta}(q_0, x) = \hat{\delta}(q_0, y)] \\
&= \hat{\delta}(q_0, yz) && [\text{Lemma 3.1}].
\end{aligned}$$

Esercizio 3.2

Siano x, y stringhe e q uno stato, dimostrare che vale:

$$\hat{\delta}(q, x) = q_1, \hat{\delta}(q, xy) = q_2 \Longrightarrow \hat{\delta}(q_1, y) = q_2.$$

Soluzione.

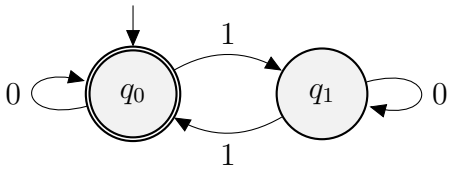
$$\begin{aligned}
\hat{\delta}(q_1, y) &= \hat{\delta}\big(\hat{\delta}(q, x), y\big) && [q_1 = \hat{\delta}(q, x)] \\
&= \hat{\delta}(q, xy) && [\text{Lemma 3.1}] \\
&= q_2.
\end{aligned}$$

Esercizio 3.3

Determinare il DFA che accetta il seguente linguaggio: $L = \{ x \in \{ 0, 1 \}^* \mid$ x contiene un numero pari di '1' $\}$.

Soluzione. Costruiamo un DFA $M = \langle Q = \{ q_0, q_1 \}, \Sigma = \{ 0, 1 \}, \delta, q_0, F = \{ q_0 \} \rangle$ che tiene traccia della lettura del simbolo '1'. L'idea è di passare dallo stato iniziale accettante q_0 al non accettante q_1 quando viene letto '1' e viceversa, così da tenere sempre taccia se il numero di '1' che sono stati letti è pari (M in q_0) o dispari (M in q_1).

Descrizione del DFA tramite grafo di transizione:



Descrizione del DFA tramite matrice di transizione:

	0	**1**
$\dot{\boldsymbol{q_0}}$	q_0	q_1
$\boldsymbol{q_1}$	q_1	q_0

Dimostrazione. Dobbiamo dimostrare che il comportamento del DFA è stato definito correttamente e fa quello che deve. Per fare questo, dimostriamo che:

1. il DFA accetta le stringhe di interesse;
2. il DFA rifiuta tutte le stringhe non di interesse.

Notiamo che, in qualunque stato il DFA si trova, leggere '0' non comporta modifiche dello stato, ovvero

$$\forall q \in Q : \delta(q, 0) = q,$$

per semplificare il nostro lavoro possiamo considerare le sole stringhe della forma 1^n, con $n \in \mathbb{N}$, tenendo a mente che potranno apparire degli '0' ovunque senza modificare lo stato dell'automa.

Accettazione. Una generica stringa accettata dal DFA è della forma:

$$1^{2n}, n \in \mathbb{N}.$$

Vogliamo mostrare che, applicando la funzione di transizione su stringhe $\hat{\delta}$ sullo stato iniziale q_0 e una stringa della forma 1^{2n}, arriviamo in uno stato $q \in F$ che è accettante, ovvero

$$\forall n \in \mathbb{N} : \hat{\delta}(q_0, 1^{2n}) = q_0 \in F.$$

Dimostriamo la per induzione su n, con caso base $n = 0$ e passo induttivo $n = m + 1$, con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 0$ abbiamo che

$$\begin{aligned}
\hat{\delta}(q_0, 1^{2n}) &= \hat{\delta}(q_0, 1^0) && [n = 0] \\
&= \hat{\delta}(q_0, \epsilon) && [\text{def. di } \cdot^0] \\
&= q_0 \in F && [\text{def. di } \hat{\delta}].
\end{aligned}$$

Passo induttivo:

$$\forall m \in \mathbb{N} : \hat{\delta}(q_0, 1^{2m}) = q_0 \overset{?}{\Longrightarrow} \hat{\delta}(q_0, 1^{2(m+1)}) = q_0.$$

Se $n = m + 1$ abbiamo che

$$\begin{aligned}
\hat{\delta}(q_0, 1^{2n}) = \hat{\delta}(q_0, 1^{2(m+1)}) &= \hat{\delta}(q_0, 111^{2m}) && [n = m + 1] \\
&= \hat{\delta}\big(\delta(q_0, 1), 11^{2m}\big) && [\text{def. di } \hat{\delta}] \\
&= \hat{\delta}(q_1, 11^{2m}) && [\text{def. di } \delta] \\
&= \hat{\delta}\big(\delta(q_1, 1), 1^{2m}\big) && [\text{def. di } \hat{\delta}] \\
&= \hat{\delta}(q_0, 1^{2m}) && [\text{def. di } \delta] \\
&= q_0 \in F && [\text{ip. ind.}].
\end{aligned}$$

Rifiuto. Una generica stringa rifiutata dal DFA è della forma:

$$1^{2n+1}, n \in \mathbb{N}. \tag{2}$$

Vogliamo mostrare che, applicando la funzione di transizione su stringhe $\hat{\delta}$ sullo stato iniziale q_0 e una stringa della forma (2), arriviamo in uno stato $q \notin F$ di rifiuto, ovvero

$$\forall n \in \mathbb{N} : \hat{\delta}(q_0, 1^{2n+1}) = q_1 \notin F,$$

Dimostriamo per induzione su n, con caso base $n = 0$ e passo induttivo $n = m + 1$, con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva. In breve:

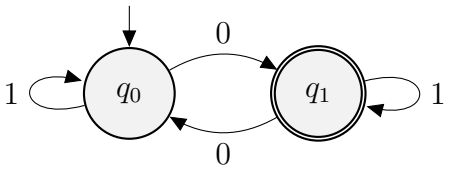
$$\hat{\delta}(q_0, 1^{2(m+1)+1}) = \hat{\delta}(q_0, 111^{2m}1) = \hat{\delta}(q_0, 1) = q_1 \notin F.$$

(Dimostrare con passaggi nel dettaglio per esercizio). □

Esercizio 3.4

Determinare il DFA che accetta il seguente linguaggio: $L = \{\, x \in \{\, 0, 1 \,\}^* \mid x$ contiene un numero dispari di '0' $\}$.

Soluzione. Descrizione del DFA $M = \langle Q = \{\, q_0, q_1 \,\}, \Sigma = \{\, 0, 1 \,\}, \delta, q_0, F = \{\, q_1 \,\}\rangle$ tramite grafo di transizione:



Descrizione del DFA tramite matrice di transizione:

	0	**1**
$\boldsymbol{q_0}$	q_1	q_0
$\boldsymbol{\dot{q}_1}$	q_0	q_1

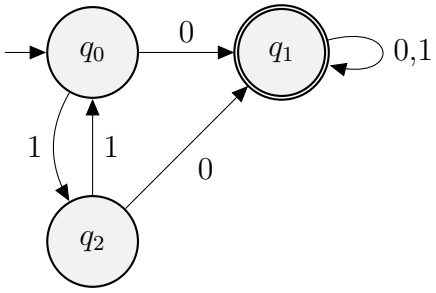
(Spiegare il ragionamento per la costruzione del DFA e dimostrare il corretto funzionamento per esercizio, notare che è tutto estremamente simile all'esercizio precedente).

Esercizio 3.5

Determinare il linguaggio accettato dal DFA descritto tramite la seguente matrice di transizione:

	0	**1**
$\boldsymbol{q_0}$	q_1	q_2
$\boldsymbol{\dot{q}_1}$	q_1	q_1
$\boldsymbol{q_2}$	q_1	q_0

Soluzione. Grafo di transizione del DFA:



Osservando il DFA notiamo che: a partire dagli stati q_0, q_2, la lettura del simbolo '1' fa ciclare tra gli stati q_0, q_2 stessi, mentre la lettura del simbolo '0' porta dentro allo stato $q_1 \in F$. Lo stato q_1 non avendo archi uscenti è detto stato *assorbente*, quindi:

$$L(M) = \{\, x \in \Sigma^* \mid x \text{ contiene almeno uno '0'} \,\}.$$

Dimostrazione. Constatiamo che a partire dagli stati q_0 e q_2, la lettura del simbolo '1' fa ciclare tra gli stati q_0, q_2 stessi:

$$\forall q \in \{\, q_0, q_2 \,\} : \delta(q, 1) = q' \in \{\, q_0, q_2 \,\} \qquad \text{[def. di } \delta\text{]},$$

da cui deriva che

$$\forall q \in \{\, q_0, q_2 \,\}, \forall n \in \mathbb{N} : \hat{\delta}(q, 1^n) = q' \in \{\, q_0, q_2 \,\}. \tag{3}$$

Constatiamo che q_1 è uno stato assorbente:

$$\forall a \in \Sigma : \delta(q_1, a) = q_1 \qquad \text{[def. di } \delta\text{]}.$$

da cui deriva che

$$\forall x \in \Sigma^*, \forall n \in \mathbb{N} : \hat{\delta}(q_1, x) = q_1. \tag{4}$$

Rifiuto. Una generica stringa rifiutata dal DFA è della forma:

$$1^n, n \in \mathbb{N}.$$

$$\forall q \in \{ q_0, q_2 \}, \forall n \in \mathbb{N} : \hat{\delta}(q, 1^n) \notin F \qquad [(3)].$$

Accettazione. Una generica stringa accettata dal DFA è della forma:

$$1^n 0y, n \in \mathbb{N}, y \in \Sigma^*.$$

Dal momento che $q_0 \in \{ q_0, q_2 \}$, abbiamo appena dimostrato che

$$\forall n \in \mathbb{N} : \hat{\delta}(q_0, 1^n 0y) = \hat{\delta}(q', 0y), \text{ con } q' \in \{ q_0, q_2 \},$$

proseguiamo

$$\begin{aligned} \hat{\delta}(q', 0y) &= \hat{\delta}\big(\delta(q', 0), y\big) && [\text{def. di } \hat{\delta}] \\ &= \hat{\delta}(q_1, y) && [\text{def. di } \delta] \\ &= q_1 \in F && [(4)]. \end{aligned}$$

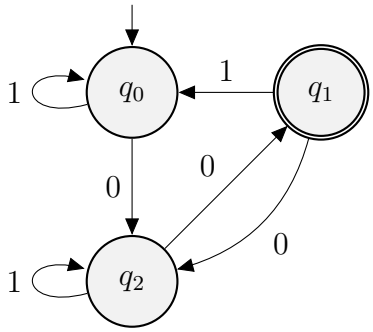
□

Esercizio 3.6

Determinare il linguaggio accettato dal DFA descritto tramite la seguente matrice di transizione:

	0	**1**
$\boldsymbol{q_0}$	q_2	q_0
$\boldsymbol{\dot{q}_1}$	q_2	q_0
$\boldsymbol{q_2}$	q_1	q_2

Soluzione. Disegnamo il grafo di transizione del DFA:



Notiamo che l'unico modo per entrare in q_2 è con la lettura di uno '0' e nel momento in cui un'altro viene letto, si passa da q_2 a $q_1 \in F$. Da q_1 si esce e si torna in q_2 con la lettura di '0'. Le osservazioni fatte fino ad ora fanno pensare che le stringhe devono contenere un numero pari di '0'. Da q_1 però con la lettura di '1' si va in q_0, quindi le stringhe devono anche terminare con '0'.

$L(M) = \{\, x \in \{\, 0, 1 \,\}^* \mid x$ contiene un numero pari di '0' e termina con '0' $\,\}$.

Dimostrazione. Mostriamo che il DFA accetta tutte e sole le stringhe.

Accettazione. Una generica stringa accettata dal DFA è della forma:

$$x_1 x_2 \cdots x_n \text{ con } n \geq 1, x_i = 1^j 0 1^k 0, j, k \in \mathbb{N}.$$

E' facile dimostrare che dagli stati q_0, q_1 la lettura di una generica x porta in $q_1 \in F$, ovvero

$$\forall q \in \{\, q_0, q_1 \,\} : \hat{\delta}(q, x) = q_1 \in F. \tag{5}$$

Vogliamo mostrare che dallo stato iniziale q_0, leggendo una concatenazione di generiche x_i si arriva in $q_1 \in F$, ovvero

$$\forall n \in \mathbb{N}, n \geq 1 : \hat{\delta}\big(q_0, x_1 x_2 \cdots x_n\big) = q_1 \in F,$$

Dimostreremo per induzione su $n \in \mathbb{N}$, con caso base $n = 1$, passo induttivo $n = m + 1$, con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: se $n = 1$ abbiamo che

$$\begin{aligned} \hat{\delta}(q_0, x_1 \cdots x_n) &= \hat{\delta}(q_0, x) && [n = 1] \\ &= q_1 \in F && [(5)]. \end{aligned}$$

Passo induttivo:

$$\forall m \geq 1 : \hat{\delta}(q_0, x_1 \cdots x_m) = q_1 \overset{?}{\Longrightarrow} \hat{\delta}(q_0, x_1 \cdots x_{m+1}) = q_1.$$

Se $n = m + 1$ abbiamo che

$$\begin{aligned} \hat{\delta}(q_0, x_1 \cdots x_n) &= \hat{\delta}(q_0, x_1 \cdots x_{m+1}) && [n = m + 1] \\ &= \hat{\delta}\big(\hat{\delta}(q_0, x_1 \cdots x_m), x_{m+1}\big) && [\text{Lemma (3.1)}] \\ &= \hat{\delta}(q_1, x_{m+1}) && [\text{ip. ind.}] \\ &= q_1 \in F && [(5)]. \end{aligned}$$

Rifiuto. Una generica stringa rifiutata dal DFA ha tutte o alcune delle seguenti caratteristiche:

- stringhe che non terminano con '0', ovvero

  $$\forall x \in \Sigma^* : \hat{\delta}(q_0, \epsilon) \notin F, \hat{\delta}(q_0, x1) \notin F.$$

  Dimostrazione.

  $$\hat{\delta}(q_0, \epsilon) = q_0 \notin F \qquad [\text{def. di } \hat{\delta}].$$

  $$\hat{\delta}(q_0, x1) = \hat{\delta}(q_1, 1) \notin F \qquad [\nexists q \in Q \,.\, \delta(q, 1) \in F].$$

  □

- stringhe che contengono un numero dispari di '0', ovvero

  $$\forall n \in \mathbb{N} : \hat{\delta}(q_0, 0^{2n+1}) \notin F,$$

  considerando il fatto che potrebbero esserci degli '1' ovunque ma non sono rilevanti per questo particolare caso perché chiaramente non entrano in gioco nel conteggio degli '0' e il caso in cui la stringa termini con '1' è gia stato coperto da punto precedente.

  $$\hat{\delta}(q_0, 0^{2n+1}) = \hat{\delta}(q_0, 0^{2n}0) = \hat{\delta}(q_1, 0) = q_2 \notin F.$$

□

Esercizio 3.7

Sia Q un insieme di stati. Sia Σ un alfabeto. Quanti sono i DFA che si possono costruire fissati Q e Σ?

Soluzione. Un DFA è una quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$, se Q possiamo contare i possibili DFA nel seguente modo:

- numero di modi in cui possiamo scegliere lo stato iniziale q_0 è

$$|Q|;$$

- numero di modi in cui possiamo scegliere l'insieme degli stati finali $F \subseteq Q$ è

$$|\mathcal{P}(Q)| = 2^{|Q|};$$

- numero di modi in cui possiamo definire la funzione di transizione δ è

$$|Q|^{|Q||\Sigma|},$$

  perché ogni in ogni cella della matrice di transizione possiamo inserire $|Q|$ valori differenti, e le dimensioni della matrice sono $|Q||\Sigma|$.

Quindi, fissati Q e Σ, è possibile costruire

$$|Q| \cdot 2^{|Q|} \cdot |Q|^{|Q||\Sigma|}$$

differenti DFA.

Esercizio 3.8

Sia L il linguaggio accettato dal DFA $M = \langle Q, \Sigma, \delta, q_0, F\rangle$, dimostrare che esiste un DFA $M' = \langle Q', \Sigma', \delta', q_0', F'\rangle$ tale che $\Sigma = \Sigma'$ e $L(M) = L(M')$.

Soluzione. Sia $M' = \langle Q', \Sigma', \delta', q_0', F' \rangle$ il DFA definito come segue:

- $Q' = Q \cup \{ q_\varnothing \}$ (stiamo aggiungendo uno stato "spazzatura" $q_\varnothing$ che renderemo irraggiungibile tramite la definzione che daremo a δ'.)
- $\Sigma' = \Sigma$;
- $\delta'(q, a) = \begin{cases} q_\varnothing & \text{se } q = q_\varnothing; \\ \delta(q, a) & \text{altrimenti.} \end{cases}$

  (abbiamo definito la $\delta' = \delta$ per ogni stato, fatta eccezione per q_t).
- $q_0' = q_0$;
- $F' = F$.

Mostriamo l'equivalenza tra M' ed M, ovvero:

$$\forall x \in \Sigma^* : \hat{\delta}'(q_0', x) = \hat{\delta}(q_0, x).$$

Procediamo per induzione sulla lunghezza n della stringa accettata, con caso base $n = 0$ e passo induttivo $n = m + 1$, con $m \in \mathbb{N}$; pertanto, nel passo induttivo potremo sfruttare l'ipotesi induttiva.

Caso base: con $|x| = 0$ abbiamo che

$$\begin{aligned} \hat{\delta}'(q_0', x) &= \hat{\delta}'(q_0', \epsilon) && [|x| = 0] \\ &= q_0' && [\text{def. di } \hat{\delta}] \\ &= q_0 && [q_0' = q_0] \\ &= \hat{\delta}(q_0, \epsilon) && [\text{def. di } \hat{\delta}] \\ &= \hat{\delta}(q_0, x) && [x = \epsilon]. \end{aligned}$$

Passo induttivo:

$$\hat{\delta}'(q_0, x) = \hat{\delta}(q_0, x) \overset{?}{\Longrightarrow} \hat{\delta}'(q_0, xa) = \hat{\delta}(q_0, xa).$$

con $n = m + 1$ abbiamo che $x = wa$, con $|w| = m$ e vale che

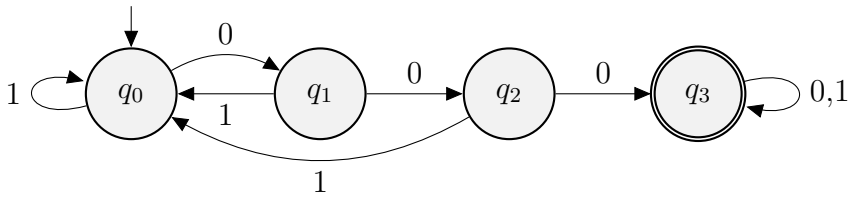
$$\begin{aligned}
\hat{\delta}'(q_0', x) &= \hat{\delta}'(q_0', wa) && [x = wa] \\
&= \delta'\big(\hat{\delta}'(q_0', w), a\big) && [\text{def. di } \hat{\delta}'] \\
&= \delta'\big(\hat{\delta}(q_0, w), a\big) && [\text{ip. ind.}] \\
&= \delta\big(\hat{\delta}(q_0, w), a\big) && [\text{def. di } \delta'] \\
&= \hat{\delta}(q_0, wa) && [\text{def. di } \hat{\delta}] \\
&= \hat{\delta}(q_0, x) && [x = wa].
\end{aligned}$$

Esercizio 3.9

Verificare che il seguente linguaggio su $\Sigma = \{\, 0, 1 \,\}$ è regolare: insieme delle stringhe binarie aventi tre '0' consecutivi.

Soluzione. Per verificare che un linguaggio è regolare, dobbiamo trovare un DFA che lo accetta e dimostrarne il corretto funzionamento (le dimostrazioni del corretto funzionamento sono lasciate per esercizio).

Costruiamo un DFA in cui ogni stato rappresenta il numero di '0' consecutivi che sono stati letti. Lo stato iniziale è q_0. Per ogni stato $q_i, 0 \leq i \leq 2$, quando viene letto '1' si va nello stato q_{i+1}, mentre se viene letto '0' si torna nello stato q_0. Nel caso si leggano tre '0' consecutivi si arriva dunque in q_3 che è stato accettante ed assorbente.



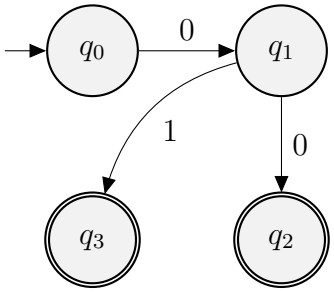
Esercizio 3.10

Verificare che il seguente linguaggio su $\Sigma = \{\, 0, 1 \,\}$ è regolare: insieme delle stringhe binarie aventi '0' come penultimo simbolo.

Soluzione. Sia $\Sigma = \{0, 1\}$. Sia x una generica string su Σ. Le stringhe accettate sono del tipo:

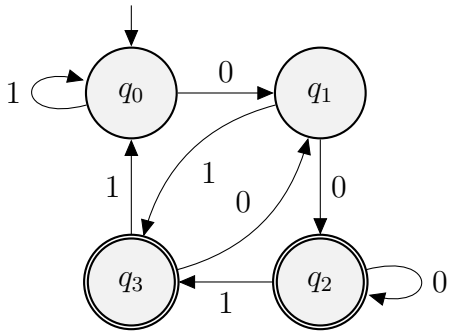
1. $x00$;
2. $x01$.

Le transizioni $q_0 \xrightarrow{0} q_1 \xrightarrow{0} \dot{q}_2$ "rilevano" che gli ultimi due simboli letti sono stati "00"; questa informazione viene poi "memorizzata" avendo $\dot{q}_2$ come stato finale. Le transazioni $q_0 \xrightarrow{0} q_1 \xrightarrow{1} \dot{q}_3$ rilevano che gli ultimi due simboli letti sono stati "01" e questa informazione viene memorizzata con $\dot{q}_3$ come stato finale.



A questo punto, $\forall q \in Q$ andiamo ad aggiungere le transizioni mancanti rispettando le proprietà degli stati $\dot{q}_2, \dot{q}_3$, ovvero:

1. partendo da q arriviamo in $\dot{q}_2 \iff$ viene letto "00";
2. partendo da q arriviamo in $\dot{q}_3 \iff$ viene letto "01".

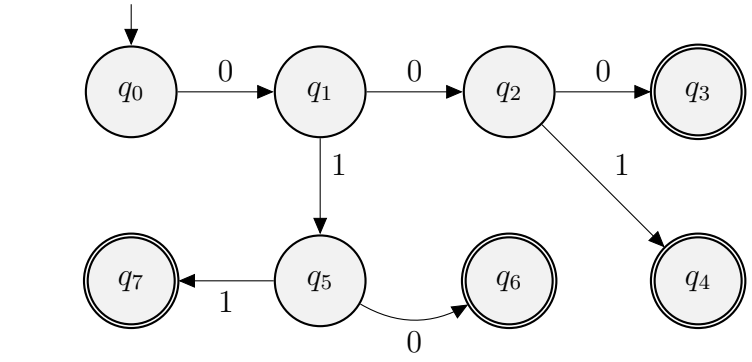


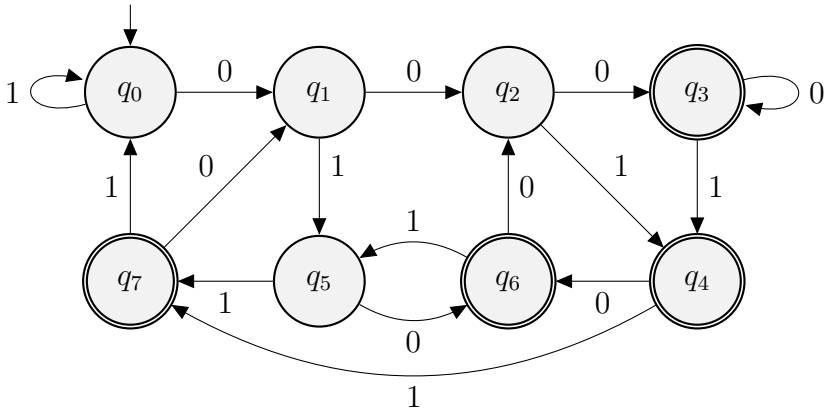
Esercizio 3.11

Verificare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe binarie aventi '0' come terzultimo simbolo.

Soluzione. Sia $\Sigma = \{0, 1\}$. Sia x una generica string su Σ. La logica è identica al DFA precedente, solo che le stringhe accettate e quindi anche gli stati aumentano (esponenzialmente):

1. $x000$, $(q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_2 \xrightarrow{0} \dot{q}_3)$;
2. $x001$, $(q_0 \xrightarrow{0} q_1 \xrightarrow{0} q_2 \xrightarrow{1} \dot{q}_4)$;
3. $x010$, $(q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_5 \xrightarrow{0} \dot{q}_6)$;
4. $x011$, $(q_0 \xrightarrow{0} q_1 \xrightarrow{1} q_5 \xrightarrow{1} \dot{q}_7)$;





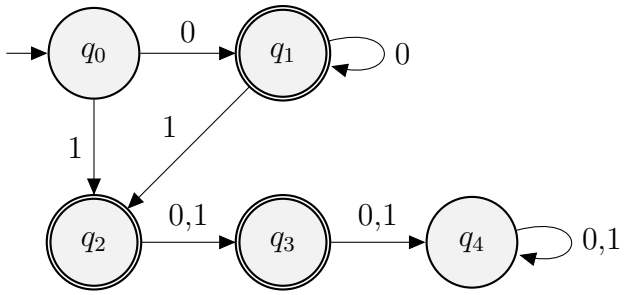
Esercizio 3.12

Verificare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe binarie tali che, se interpretate come numero decimale, hanno un valore minore o uguale a 3.

Soluzione. Stringhe accettate:

- $0_2 \mapsto 0_{10} : q_0 \xrightarrow{0} \dot{q}_1$;
- $1_2 \mapsto 1_{10} : q_0 \xrightarrow{1} \dot{q}_2$;
- $10_2 \mapsto 2_{10} : q_0 \xrightarrow{1} q_2 \xrightarrow{0} \dot{q}_3$;
- $11_2 \mapsto 3_{10} : q_0 \xrightarrow{1} q_2 \xrightarrow{1} \dot{q}_3$.

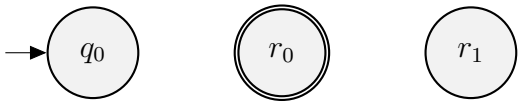
Utilizziamo q_4 come stato assorbente non finale, in modo da rifiutare tutte le stringhe con valore maggiore di 3.



Esercizio 3.13

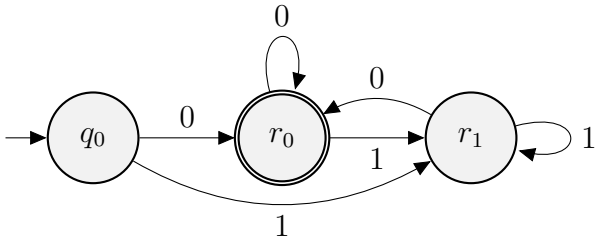
Verificare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe binarie tali che, se interpretate come numero decimale, sono divisibili per 2.

Soluzione. Avremo lo stato iniziale q_0 non accettante perché non vogliamo accettare la stringa ϵ visto che non ha interpretazione nei numeri interi senza segno. Da q_0 ci spostiamo in r_0 o r_1 a seconda se viene letto '0' o '1' rispettivamente. La r nel nome degli stati sta per "resto"; negli stati r_0, r_1 infatti, vi si arriva quando, interpretando come numero intero la stringa letta e facendone il modulo 2, si ottiene resto 0 e resto 1 rispettivamente.



Facciamo un "mapping" dei numeri da 0 a n, in accordo con quanto detto sopra, fino a che tutte le tansizioni non sono state definite (raggiunto un *punto fisso*):

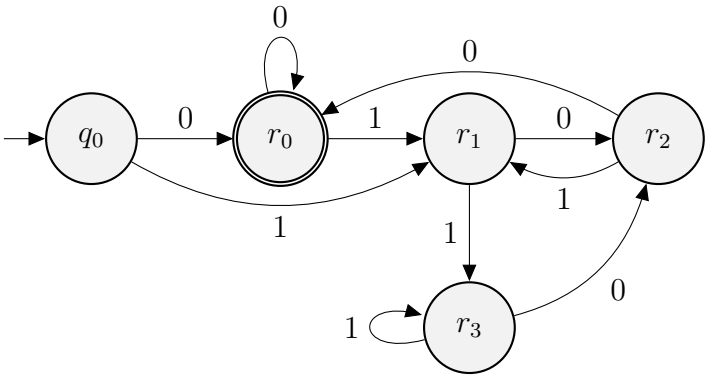
- $0_2 \mapsto 0_{10} \mod 2 = 0 : q_0 \xrightarrow{0} \dot{r}_0$;
- $1_2 \mapsto 1_{10} \mod 2 = 1 : q_0 \xrightarrow{1} r_1$;
- $10_2 \mapsto 2_{10} \mod 2 = 0 : q_0 \xrightarrow{1} r_1 \xrightarrow{0} \dot{r}_0$;
- $11_2 \mapsto 3_{10} \mod 2 = 1 : q_0 \xrightarrow{1} r_1 \xrightarrow{1} r_1$;
- $100_2 \mapsto 4_{10} \mod 2 = 0 : q_0 \xrightarrow{1} r_1 \xrightarrow{0} r_0 \xrightarrow{0} \dot{r}_0$;
- $101_2 \mapsto 5_{10} \mod 2 = 1 : q_0 \xrightarrow{1} r_1 \xrightarrow{0} r_0 \xrightarrow{1} r_1$.
- $110_2 \mapsto 6_{10} \mod 2 = 0 : q_0 \xrightarrow{1} r_1 \xrightarrow{1} r_1 \xrightarrow{0} \dot{r}_0$ (nessuna nuova transazione aggiunta, punto fisso raggiunto).



Esercizio 3.14

Verificare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe binarie tali che, se interpretate come numero decimale, sono divisibili per 4.

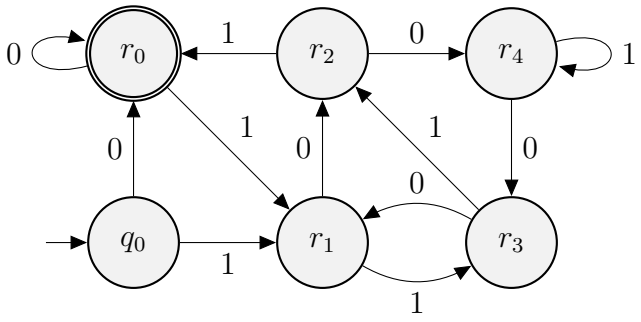
Soluzione. Ragionamento identico a quello dell'esercizio precedente.



Esercizio 3.15

Verificare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe tali che, se interpretate come numero decimale, sono divisibili per 5.

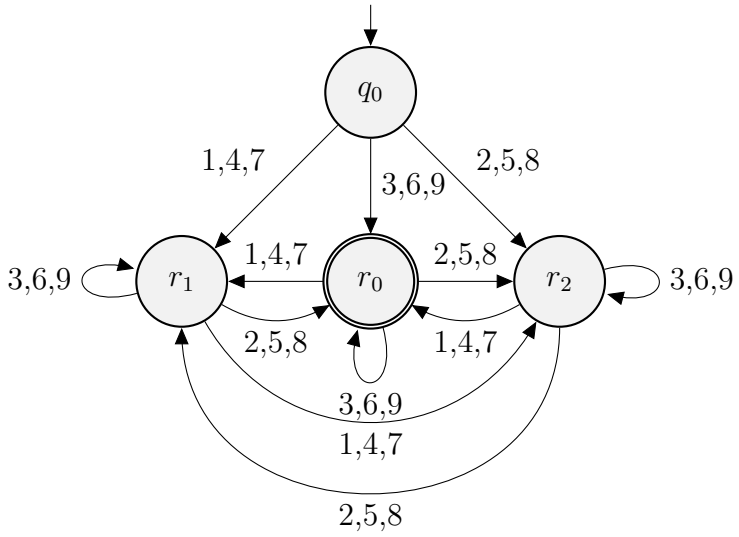
Soluzione. Ragionamento simile a quello dell'esercizio precedente.



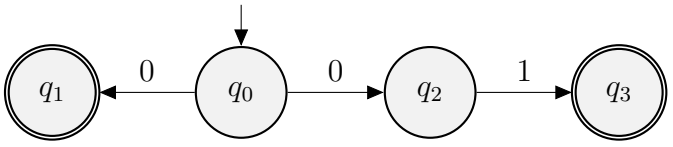
Esercizio 3.16

Veriricare che il seguente linguaggio su $\Sigma = \{0, 1\}$ è regolare: insieme delle stringhe di cifre decimali tali che, sono divisibili per 3.

Soluzione. Seguendo il suggerimento, costruiamo un DFA che si muove tra gli stati $\dot{r}_0, r_1, r_2$ a seconda se la somma delle cifre che compongono la stringa divisa per 3 da resto 0, resto 1 o resto 2 rispettivamente.



3.3 NFA: automi a stati finiti non deterministici



	0	**1**
$\boldsymbol{q_0}$	$\{ q_1, q_2 \}$	$\varnothing$
$\boldsymbol{\dot{q}_1}$	$\varnothing$	$\varnothing$
$\boldsymbol{q_2}$	$\varnothing$	$\{ q_3 \}$
$\boldsymbol{\dot{q}_3}$	$\varnothing$	$\varnothing$

Per come sono definiti gli NFA, a partire da uno stato $q \in Q$, con la lettura un simbolo $a \in \Sigma$, è possibile transire verso nessuno stato, verso uno solo stato $q' \in Q$ o verso più stati contemporaneamente (un insieme di stati $S \subseteq Q$). Questo equivale a dire che, negli NFA, a partire da uno stato $q \in Q$, con la lettura di un simbolo $a \in \Sigma$, si transita in un insieme di stati $S \in \mathcal{P}(Q)$.

3.3.1 Definizioni, Teoremi e Lemmi

Definizione 3.5 (NFA: automa a stati finiti non deterministico)**.** *Un NFA è una quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$, la cui definizione differisce dalla quintupla dei DFA solo nella funzione di transizione che è definita come:*

$$\delta : Q \times \Sigma \mapsto \mathcal{P}(Q).$$

La differenza è che, negli NFA, la funzione di transizione δ non restituisce in uscita un singolo stato che sta in Q ma invece restituisce un *sottoinsieme* degli stati che stanno in Q, ovvero un elemento di $\mathcal{P}(Q)$. L'aggettivo *non deterministico* nel nome è dovuto proprio a questo; l'NFA, a seguito della lettura di un simbolo può transitare verso più stati e quindi trovarsi in più stati contemporaneamente ovvero, non è più deterministico.

Come per i DFA, anche per gli NFA è possibile utilizzare la funzione di transizione δ per la definizione della funzione di transizione su stringhe $\hat{\delta}$.

Definizione 3.6 (Funzione di transizione su stringhe)**.** *Sia M l'NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La funzione di transizione su stringhe $\hat{\delta} : Q \times \Sigma^* \mapsto \mathcal{P}(Q)$ è definita come:*

$$\hat{\delta}(q, x) := \begin{cases} \{\, q \,\} & \text{se } x = \epsilon; \\ \bigcup_{p \in \hat{\delta}(q,y)} \delta(p, a) & \text{se } x = ya. \end{cases}$$

Osservazione 3.2. *Nella definizione di $\hat{\delta}$ vi è un elemento di arbitrarietà. Si può infatti dimostrare che, $\hat{\delta}$ si può sostituire con la funzione $\check{\delta}$ così definita:*

$$\check{\delta}(q, x) := \begin{cases} \{\, q \,\} & \text{se } x = \epsilon; \\ \bigcup_{p \in \delta(q,a)} \hat{\delta}(p, y) & \text{se } x = ay. \end{cases}$$

Definizione 3.7 (Accettazione di una stringa)**.** *Sia M un NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La stringa x è* accettata *da M se e solo se:*

$$\hat{\delta}(q_0, x) \cap F \neq \varnothing.$$

Una stringa x quindi è accettata da un NFA M se, partendo dallo stato iniziale q_0 e leggendo x, l'automa termina in un insieme di stati che ha intersezione non nulla con F.

Definizione 3.8 (Linguaggio accettato da un NFA)**.** *Sia M l'NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Il linguaggio* accettato *da M si indica con $L(M)$ ed è l'insieme di tutte le stringhe accettate da M, ovvero:*

$$L(M) := \{\, x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \varnothing \,\}.$$

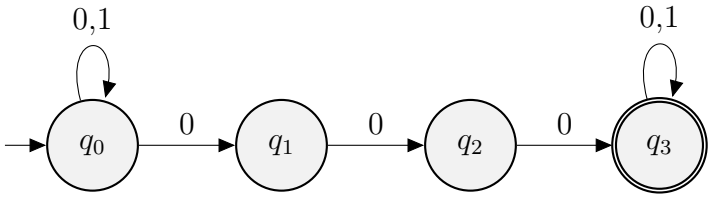
Teorema 3.2 (Rabin-Scott: equivalenza tra NFA e DFA)**.** *Sia $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ un NFA, allora esiste un DFA M' tale che*

$$L(M) = L(M').$$

3.3.2 Costruzione per sottoinsiemi

Dato un NFA $M = \langle Q, \Sigma, \delta, q_0, F\rangle$ è possibile applicare un algoritmo che permette di trovare il DFA equivalente, questo algoritmo prende il nome di *costruzione per sottoinsiemi*. Come abbiamo visto, la funzione di transizione δ su NFA restituisce in uscita un elemento che sta in $\mathcal{P}(Q)$, quindi restituisce un insieme di stati e questo comporta che l'NFA si può trovare contemporaneamente in più stati. L'idea è di costruire un DFA M' che ha uno stato per ogni possibile elemento in $\mathcal{P}(Q)$, così facendo, in ogni momento è possibile "riassumere" l'insieme di stati in cui si trova l'NFA M con un singolo stato del DFA M'. Notare che, il DFA M' rispetto all'equivalente NFA M avrà un numero di stati che può risultare esponenzialmente più grande, infatti nel caso pessimo possono servire proprio $|\mathcal{P}(Q)| = 2^{||Q}$ stati.

Esempio. Grafo e matrice di transizione per un NFA che accetta il linguaggio "insieme di tutte le stringhe aventi tre '0' consecutivi":



	0	**1**
$\boldsymbol{q_0}$	$\{\, q_0, q_1 \,\}$	$\{\, q_0 \,\}$
$\boldsymbol{q_1}$	$\{\, q_2 \,\}$	$\varnothing$
$\boldsymbol{q_2}$	$\{\, q_3 \,\}$	$\varnothing$
$\boldsymbol{\dot{q}_3}$	$\{\, q_3 \,\}$	$\{\, q_3 \,\}$

Costruzione per sottoinsiemi:

1. per definizione di NFA, all'interno delle celle della matrice di transizione troviamo insiemi di stati $S_1 \cdots S_n$. Tra questi identifichiamo gli stati S_i tali che $|S|_i \geq 2$ (ad esempio $\{ q_0, q_1 \}$) e sono raggiungibili da q_0. Se questi non sono già stati "etichettati", etichettiamo questi insiemi di stati, ovvero associamo ad ognuno di questi un nuovo e singolo stato che li rappresenta, ad esempio $\{ q_0, q_1 \} = q_{01}$.

   Se all'interno dell'insieme di stati che si sta etichettando è presente almeno uno stato finale, lo stato che si utilizza come etichetta diventa anche stato finale del DFA equivalente che stiamo costruendo.

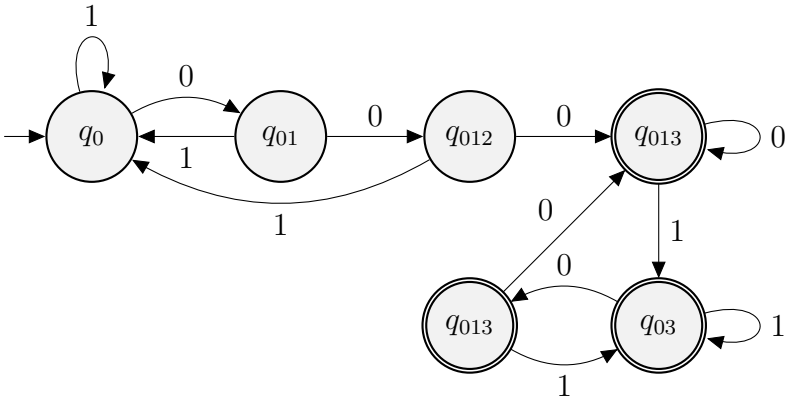
   Per ogni nuovo insieme di stati trovato ed etichettato, aggiungiamo una riga alla matrice con tutte le celle vuote tranne la prima (che conterrà proprio l'insieme di stati oppure in maniera equivalente il nuovo stato-etichetta che gli abbiamo associato);

2. riempire le celle vuote, univocamente identificabili dalla coppia (S, a), inserendovi dentro $\bigcup_{s \in S} \delta(s, a)$.

3. ripetere dal punto 1 fino a che non si raggiunge un punto fisso.

	0	**1**
$\boldsymbol{q_0}$	q_{01}	q_0
$\boldsymbol{q_{01}}$	q_{012}	q_0
$\boldsymbol{q_{012}}$	q_{0123}	q_0
$\boldsymbol{\dot{q}_{0123}}$	q_{0123}	q_{03}
$\boldsymbol{\dot{q}_{03}}$	q_{013}	q_{03}
$\boldsymbol{\dot{q}_{013}}$	q_{0123}	q_{03}

Grafo di transizione del DFA equivalente con i soli stati raggiungibili a partire da q_0:



Notare che gli stati q_{013}, q_{03} potrebbero essere eliminati, il DFA generato utilizzando la costruzione per sottoinsiemi quindi non è *minimo*.

3.3.3 Esercizi

Esercizio 3.17

Sia Q un insieme di stati. Sia Σ un alfabeto. Quanti sono gli NFA che si possono costruire fissati Q e Σ?

Soluzione. Un NFA è una quintupla $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, se Q possiamo contare i possibili NFA nel seguente modo:

- numero di modi in cui possiamo scegliere lo stato iniziale q_0 è

$$|Q|;$$

- numero di modi in cui possiamo scegliere l'insieme degli stati finali $F \subseteq Q$ è

$$|\mathcal{P}(Q)| = 2^{|Q|};$$

- numero di modi in cui possiamo definire la funzione di transizione δ è

$$\left(2^{|Q|}\right)^{|Q||\Sigma|},$$

  perché ogni in ogni cella della matrice di transizione possiamo inserire $2^{|Q|}$ valori differenti, e le dimensioni della matrice sono $|Q||\Sigma|$.

Quindi, fissati Q e Σ, è possibile costruire

$|Q| \cdot 2^{|Q|} \cdot \left(2^{|Q|}\right)^{|Q||\Sigma|} = |Q| \cdot 2^{|Q|} \cdot 2^{|Q|^2|\Sigma|} = |Q| \cdot 2^{|Q|^2|\Sigma|+|Q|} = |Q| \cdot 2^{|Q|(|Q||\Sigma|+1)}$

differenti NFA.

Esercizio 3.18

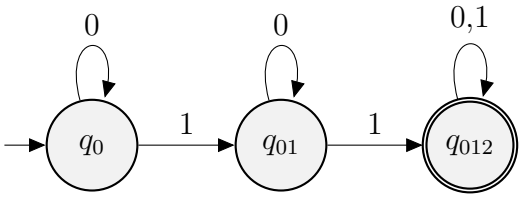
Determinare il DFA equivalente al seguente NFA e il linguaggio accettato.

	0	**1**
$\boldsymbol{q_0}$	$\{ q_0 \}$	$\{ q_0, q_1 \}$
$\boldsymbol{q_1}$	$\{ q_1 \}$	$\{ q_0, q_2 \}$
$\boldsymbol{\dot{q}_2}$	$\{ q_1, q_2 \}$	$\{ q_0, q_1, q_2 \}$

Soluzione. Applichiamo la costruzione per sottoinsiemi:

	0	**1**
$\boldsymbol{q_\varnothing}$	$q_\varnothing$	$q_\varnothing$
$\boldsymbol{q_0}$	q_0	q_{01}
$\boldsymbol{q_{01}}$	q_{01}	q_{012}
$\boldsymbol{\dot{q}_{012}}$	q_{012}	q_{012}

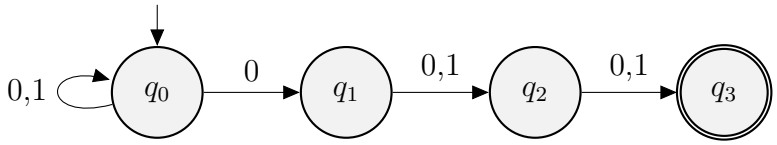
Grafo di transizione dei soli stati raggiungibili a partire da q_0:



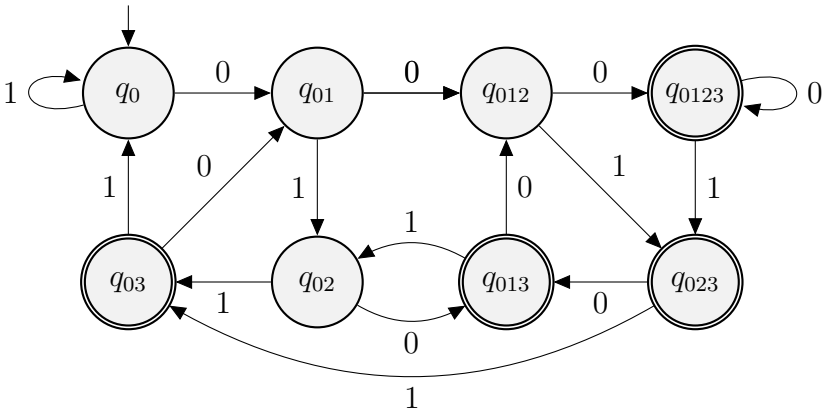
$L(M) = \{ x \in \Sigma^* \mid x$ contiene almeno due occorrenze del simbolo '1' $\}$.

Esercizio 3.19

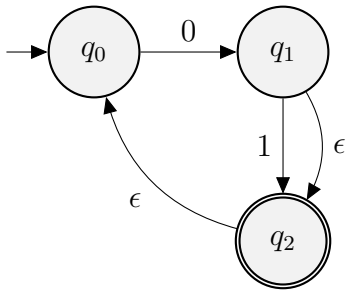
Costruire un NFA con 4 stati che accetta il seguente linguaggio: $L = \{ x \in \{ 0, 1 \} \mid x$ ha '0' come terzultimo simbolo $\}$ e trovare il DFA equivalente.



	0	**1**
$\boldsymbol{q_0}$	q_{01}	q_0
$\boldsymbol{q_{01}}$	q_{012}	q_{02}
$\boldsymbol{q_{012}}$	q_{0123}	q_{023}
$\boldsymbol{q_{02}}$	q_{013}	q_{03}
$\boldsymbol{\dot{q}_{0123}}$	q_{0123}	q_{023}
$\boldsymbol{\dot{q}_{023}}$	q_{013}	q_{03}
$\boldsymbol{\dot{q}_{013}}$	q_{012}	q_{02}
$\boldsymbol{\dot{q}_{03}}$	q_{01}	q_0



3.4 ϵ-NFA: automi a stati finiti non deterministici con ϵ-transizioni



	ϵ	0	1
q_0	$\varnothing$	$\{ q_1 \}$	$\varnothing$
q_1	$\{ q_2 \}$	$\varnothing$	$\{ q_2 \}$
$\dot{q}_2$	$\{ q_0 \}$	$\varnothing$	$\varnothing$

Per come sono definiti gli ϵ-NFA, a partire da uno stato $q \in Q$, è possibile transire verso un insieme di stati $S \in \mathcal{P}(Q)$ anche senza aver letto nessun simbolo, questo è possibile grazie a particolari transizioni, le *ϵ-transizioni*.

3.4.1 Definizioni, Teoremi e Lemmi

Definizione 3.9 (ϵ-NFA: automa a stati finiti non deterministico con ϵ-transizioni)**.** *Un ϵ-NFA è una quintupla $\langle Q, \Sigma, \delta, q_0, F \rangle$, la cui definizione differisce dalla quintupla degli NFA solo nella funzione di transizione che è definita come:*

$$\delta : Q \times (\Sigma \cup \{ \epsilon \}) \mapsto \mathcal{P}(Q).$$

La funzione di transizione negli ϵ-NFA quindi può prendere in ingresso sia simboli dell'alfabeto che il simbolo ϵ da cui derivano le ϵ-transizioni. Prima di definire la funzione di transizione su stringhe, definiamo formalmente *ϵ-step* ed *ϵ-closure* che consistono nell'applicare delle ϵ-transizioni ad insiemi di stati.

Definizione 3.10 (ϵ-step)**.** *Sia M un ϵ-NFA definito dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$ e sia $S \in \mathcal{P}(Q)$. ϵ-step: $\mathcal{P}(Q) \mapsto \mathcal{P}(Q)$ è definita come:*

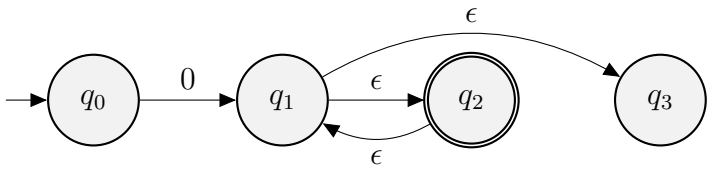
$$\epsilon\text{-}step(S) = \{\, q \in Q \mid \exists p \in S \,.\, q \in \delta(p, \epsilon) \,\}.$$

L'ϵ-step applicata ad un insieme S di stati quindi ritorna come risultato un insieme di stati che è ottenuto applicando le *epsilon*-transizioni a tutti gli elementi di S.

Definizione 3.11 (ϵ-step iterati)**.** *Sia M un ϵ-NFA definito dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$ e sia $S \in \mathcal{P}(Q)$,*

$$\epsilon\text{-}step^n(S) := \begin{cases} S & \text{se } n = 0; \\ \epsilon\text{-}step(\epsilon\text{-}step^{n-1}(S)) & \text{altrimenti.} \end{cases}$$

Esempio. Grafo di transizione di un ϵ-NFA M:



Alcuni ϵ-step sugli stati di M:

$$\epsilon\text{-step}^0\big(\{\, q_0 \,\}\big) = \{\, q_0 \,\} \qquad \text{[def. di } \cdot^0\text{]}.$$

$$\epsilon\text{-step}\big(\{\, q_0 \,\}\big) = \varnothing \qquad \text{[def. di } \epsilon\text{-step]}.$$

$$\epsilon\text{-step}\big(\{\, q_1 \,\}\big) = \{\, q_2, q_3 \,\} \qquad \text{[def. di } \epsilon\text{-step]}.$$

$$\begin{aligned} \epsilon\text{-step}^2\big(\{\, q_1 \,\}\big) &= \epsilon\text{-step}\big(\epsilon\text{-step}(\epsilon\text{-step}^0(\{\, q_1 \,\}))\big) && \text{[def. di } \cdot^n\text{]} \\ &= \epsilon\text{-step}\big(\epsilon\text{-step}(\{\, q_1 \,\})\big) && \text{[def. di } \cdot^0\text{]} \\ &= \epsilon\text{-step}\big(\{\, q_2, q_3 \,\}\big) && \text{[def. di } \epsilon\text{-step]} \\ &= \{\, q_1 \,\} && \text{[def. di } \epsilon\text{-step]}. \end{aligned}$$

Definizione 3.12 (ϵ-closure)**.** *Sia M un ϵ-NFA definito dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia $S \in \mathcal{P}(Q)$. ϵ-closure$: \mathcal{P}(Q) \mapsto \mathcal{P}(Q)$ è definita come:*

$$\epsilon\text{-}closure(S) := \bigcup_{i\in\mathbb{N}} \epsilon\text{-}step^i(S).$$

Sia $q \in Q$ allora ϵ-closure$: Q \mapsto \mathcal{P}(Q)$ è definita come:

$$\epsilon\text{-}closure(q) := \bigcup_{i\in\mathbb{N}} \epsilon\text{-}step^i(\{\, q \,\}).$$

Esempio.

$$\begin{aligned}
\epsilon\text{-closure}(q_1) &= \epsilon\text{-step}^0(\{\, q_1 \,\}) \cup \epsilon\text{-step}(\{\, q_1 \,\}) \cup \epsilon\text{-step}^2 \cdots && \text{[def. di } \epsilon\text{-closure]}\\
&= \{\, q_1 \,\} \cup \{\, q_2, q_3 \,\} \cup \{\, q_1 \,\} \cup \{\, q_2, q_3 \,\} \cup \cdots && \text{[def. di } \epsilon\text{-step]}\\
&= \{\, q_1, q_2, q_3 \,\} && \text{[def. di } \cup\text{]}.
\end{aligned}$$

Notare che anche se l'ϵ-closure è un unione su tutto $\mathbb{N}$, dal momento che gli stati dell'automa sono finiti e quindi anche il numero di transizioni sono finite, si arriva ad un certa i per la quale applicare ϵ-step^i aggiunge soltando duplicati al nostro insieme quindi il punto fisso è stato raggiunto e non serve continuare, nell'esempio il punto fisso si raggiunge per $i = 1$, ovvero per $i = 2$ viene aggiunto un duplicato.

Lemma 3.2 (Idempotenza di ϵ-closure)**.** *Sia $q \in Q$ uno stato.*

$$\epsilon\text{-}closure(\epsilon\text{-}closure(q)) = \epsilon\text{-}closure(q).$$

Dimostrazione.

$$\begin{aligned}
\epsilon\text{-closure}(\epsilon\text{-closure}(q)) &= \bigcup_{i\in\mathbb{N}} \epsilon\text{-step}^i\Big(\bigcup_{j\in\mathbb{N}} \epsilon\text{-step}^j\big(\{\, q \,\}\big)\Big) && \text{[def. di } \epsilon\text{-closure]}\\
&= \bigcup_{i\in\mathbb{N}}\bigcup_{j\in\mathbb{N}} \epsilon\text{-step}^i\big(\epsilon\text{-step}^j(\{\, q \,\})\big) && \text{[def. di } \cup\text{]}\\
&= \bigcup_{i\in\mathbb{N}}\bigcup_{j\in\mathbb{N}} \epsilon\text{-step}^{ij}\big(\{\, q \,\}\big) && \text{[def. di } \cdot^n\text{]}\\
&= \bigcup_{k\in\mathbb{N}} \epsilon\text{-step}^k\big(\{\, q \,\}\big) && [\{\, ij \mid i \in \mathbb{N}, j \in \mathbb{N} \,\} = \mathbb{N}]\\
&= \epsilon\text{-closure}(q) && \text{[def. di } \epsilon\text{-closure]}.
\end{aligned}$$

□

Definizione 3.13 (Funzione di transizione su stringhe)**.** *Sia M un ϵ-NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La funzione di transizione su stringhe $\hat{\delta} : Q \times \Sigma^* \mapsto \mathcal{P}(Q)$ è definita come:*

$$\hat{\delta}(q, x) := \begin{cases} \epsilon\text{-}closure(q) & se\ x = \epsilon; \\ \bigcup_{p \in \hat{\delta}(q,y)} \epsilon\text{-}closure\big(\delta(p, a)\big) & se\ x = ya. \end{cases}$$

Osservazione 3.3. *Nella definizione di $\hat{\delta}$ vi è un elemento di arbitrarietà. Si può infatti dimostrare che, $\hat{\delta}$ si può sostituire con la funzione $\check{\delta}$ così definita:*

$$\check{\delta}(q, x) := \begin{cases} \epsilon\text{-}closure(q) & se\ x = \epsilon; \\ \bigcup_{r \in \left(\bigcup_{p \in \epsilon\text{-}closure(q)} \delta(p,a)\right)} \hat{\delta}(r, y) & se\ x = ay \end{cases}$$

Nella pratica, se un ϵ-NFA M si trova in un insieme di stati S, prima di leggere il prossimo simbolo in ingresso a, viene applicata la ϵ-closure(S) che ha come conseguenza la transizione di M dall'insieme di stati S verso un insieme $S' = \epsilon$-closure(S). Da S' è poi possibile procedere con la lettura del prossimo simbolo in ingresso a che farà transitare M verso un altro insieme di stati S''. Si procede in questo modo finché tutti i simboli sul nastro sono stati letti, ovvero fino a che sul nastro di ingresso rimane la stringa vuota ϵ. Con la sola stringa vuota sul nastro di ingresso, si applica un ultima volta la ϵ-closure ottenendo un ultimo insieme di stati T. Se T ha intersezione non vuota con F, allora la stringa che è stata letta un simbolo alla volta è accettata da M.

Definizione 3.14 (Accettazione di una stringa)**.** *Sia M un ϵ-NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Sia x una stringa su Σ^*. La stringa x è* accettata *da M se e solo se:*

$$\hat{\delta}(q_0, x) \cap F \neq \varnothing.$$

Una stringa x quindi è accettata da un ϵ-NFA M se, partendo dallo stato iniziale q_0 e leggendo x, l'automa termina in un insieme di stati che ha intersezione non nulla con F.

Definizione 3.15 (Linguaggio accettato da un NFA)**.** *Sia M un ϵ-NFA descritto dalla quintupla $\langle Q, \Sigma, \delta, q_0, F\rangle$. Il linguaggio* accettato *da M si indica con $L(M)$ ed è l'insieme di tutte le stringhe accettate da M, ovvero:*

$$L(M) := \{\, x \in \Sigma^* \mid \hat{\delta}(q_0, x) \cap F \neq \varnothing \,\}.$$

Teorema 3.3 (Equivalenza tra ϵ-NFA e NFA)**.** *Sia $M = \langle Q, \Sigma, \delta, q_0, F \rangle$ un ϵ-NFA, allora esiste un NFA M' tale che*

$$L(M) = L(M').$$

Corollario 3.1 (Equivalenza tra DFA, NFA e ϵ-NFA)**.** *Le classi di linguaggi accettati da DFA, NFA ed ϵ-NFA coincidono (linguaggi regolari).*

3.4.2 Eliminazione delle ϵ-transizioni

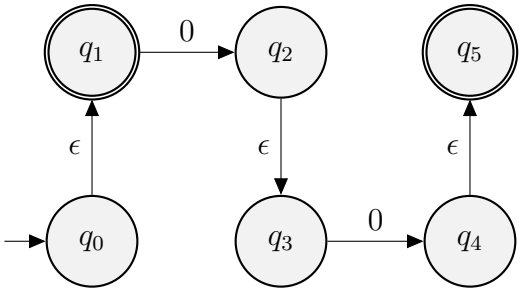
Dato un ϵ-NFA, per trovare un NFA equivalente, è necessario eliminare le ϵ-transizioni. Per fare questo, eseguiamo i seguenti passaggi $\forall q \in Q, \forall a \in \Sigma$:

1. applichiamo ϵ-closure(q_0), ottenendo un insieme di stati S. Se $q = q_0$ e ϵ-closure$(q_0) \cap F \neq \varnothing$ allora q_0 oltre che stato iniziale diventa anche stato accettante nell'NFA equivalente che stiamo costruendo;

2. $\forall s \in S$ applichiamo la $\delta(s, a)$ e collezioniamo gli stati ottenuti in un insieme che chiameremo $S'' = \bigcup_{s \in S} \delta(s, a)$;

3. applichiamo la ϵ-closure$(S'') = S'''$ ottenendo un insieme di stati che, se stiamo costruendo una matrice di transizione, andremo a scrivere in corrispondenza della cella $[q, a] = S'''$.

Esempio. Descrizione di un ϵ-NFA tramite grafo di transizione:



Per completare ad esempio la cella $(\dot{q}_0, 0)$ della matrice di transizione del NFA qeuivalente partiamo dallo stato q_0 del ϵ-NFA e applichiamo:

$$q_0 \overset{\epsilon\text{-closure}}{\longrightarrow} \{ q_0, q_1 \} \overset{0}{\longrightarrow} \{ q_2 \} \overset{\epsilon\text{-closure}}{\longrightarrow} \{ q_2, q_3 \}.$$

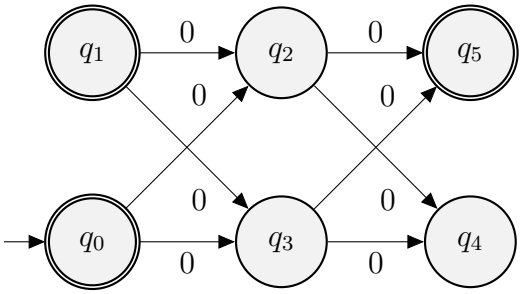
dal momento che nella ϵ-closure(q_0) è presente lo stato finale $\dot{q}_1$, lo stato q_0 nel NFA equivalente diventa stato finale, stesso discorso per q_1:

$$q_1 \overset{\epsilon\text{-closure}}{\longrightarrow} \{\, q_1 \,\} \overset{0}{\longrightarrow} \{\, q_2 \,\} \overset{\epsilon\text{-closure}}{\longrightarrow} \{\, q_2, q_3 \,\}.$$

Ripetendo questo processo per ogni stato dell ϵ-NFA e ogni simbolo dell'afabeto, si ottiene la matrice di transizione dell'NFA equivalente:

	0	**1**
$\dot{\boldsymbol{q}}_{\mathbf{0}}$	$\{\, q_2, q_3 \,\}$	$\varnothing$
$\dot{\boldsymbol{q}}_{\mathbf{1}}$	$\{\, q_2, q_3 \,\}$	$\varnothing$
$\boldsymbol{q}_{\mathbf{2}}$	$\{\, q_4, q_5 \,\}$	$\varnothing$
$\boldsymbol{q}_{\mathbf{3}}$	$\{\, q_4, q_5 \,\}$	$\varnothing$
$\boldsymbol{q}_{\mathbf{4}}$	$\varnothing$	$\varnothing$
$\dot{\boldsymbol{q}}_{\mathbf{5}}$	$\varnothing$	$\varnothing$

Grafo di transizione del NFA equivalente:



Nota in generale.

1. Fornire a un DFA la possibilità di produrre output, eventualmente scrivendo sul nastro, non ne aumenta la capacità computazionale perché ciò che viene scritto non potrebbe essere mai letto.

2. Fornire a un DFA la possibilità di muovere la testina sia a sinistra che a destra non ne aumenta la capacità computazionale perché i dati letti rimarrebbero sempre gli stessi.

3. La combinazione dei punti 1 e 2 permette di passare a un formalismo più potente (*macchine di Turing*).

3.4.3 Esercizi

Esercizio 3.20

Sia Q un insieme di stati. Sia Σ un alfabeto. Quanti sono gli ϵ-NFA che si possono costruire fissati Q e Σ?

Soluzione. Un ϵ-NFA è una quintupla $M = \langle Q, \Sigma, \delta, q_0, F \rangle$, se Q possiamo contare i possibili ϵ-NFA nel seguente modo:

- numero di modi in cui possiamo scegliere lo stato iniziale q_0 è

$$|Q|;$$

- numero di modi in cui possiamo scegliere l'insieme degli stati finali $F \subseteq Q$ è

$$|\mathcal{P}(Q)| = 2^{|Q|};$$

- numero di modi in cui possiamo definire la funzione di transizione δ è

$$\left(2^{|Q|}\right)^{|Q|(|\Sigma|+1)},$$

  perché ogni in ogni cella della matrice di transizione possiamo inserire $2^{|Q|}$ valori differenti, e le dimensioni della matrice sono $|Q|\big(|\Sigma|+1\big)$.

Quindi, fissati Q e Σ, è possibile costruire

$$|Q| \cdot 2^{|Q|} \cdot \left(2^{|Q|}\right)^{|Q|(|\Sigma|+1)} = |Q| \cdot 2^{|Q|} \cdot 2^{|Q|^2(|\Sigma|+1)} = |Q| \cdot 2^{|Q|^2(|\Sigma|+1)+|Q|}$$

differenti ϵ-NFA.