Appunti di
Linguaggi Formali

Matteo Cavallaro
matteo.cavallaro@studium.unict.it

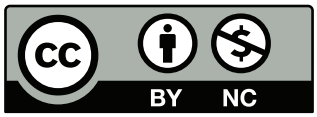
Revisione del 30 giugno 2020

Dipartimento di Matematica e Informatica
Università degli Studi di Catania

Anno Accademico 2018-2019

Prefazione

Questo documento contiene una rielaborazione personale degli appunti del corso di Linguaggi Formali tenuto dalla professoressa Marina Madonia [1] presso il Dipartimento di Matematica e Informatica dell'Università di Catania.



[1] mail: madonia@dmi.unict.it
website: www.dmi.unict.it/~madonia/

Indice

Capitolo 1

Introduzione

Un **linguaggio** è un insieme di parole a partire da un **alfabeto** Σ finito. Analizzeremo solo linguaggi di parole finite; linguaggi con parole di lunghezza infinita sono detti *omega-linguaggi*.

Definizione 1.0.1. Sia Σ un alfabeto; una *parola* (o *stringa*) su Σ è così definita:

- ε è una stringa;
- se x è una stringa e $a \in \Sigma$, allora xa è una stringa.

Il simbolo ε identifica la stringa vuota.
Σ^* denota l'insieme delle stringhe su Σ.

1.1 Operazioni su stringhe

Definizione 1.1.1. Se $x, y \in \Sigma^*$, la *concatenazione* di x e y ($\circ$) è così definita:

$$x \circ y = xy$$
$$x \circ \varepsilon = x$$

Osservazione 1.1.1. *L'operazione di concatenazione gode delle seguenti proprietà:*

1. $\forall x \in \Sigma^* : x \circ \varepsilon = x$ *(ovvero* $\varepsilon \in \Sigma^*$ *è* elemento neutro *rispetto a* $\circ$*);*
2. $\forall x, y \in \Sigma^* : x \circ y \in \Sigma^*$ *(ovvero* $\circ$ *è un'*operazione intera *a* Σ^**);*
3. $\forall x, y, z \in \Sigma^* : (x \circ y) \circ z = x \circ (y \circ z)$ *(ovvero* $\circ$ *gode della* p. associativa*);*
4. $(\Sigma^*, \circ, \varepsilon)$ *è un monoide, detto* monoide libero *o* non sintetico *su* Σ^**.*

Nota: è possibile definire gli inversi degli elementi di un alfabeto.

Esempio 1.1.1. $\Sigma = \{a, b, a^{-1}, b^{-1}\}, \quad aa^{-1} = \varepsilon.$

Definizione 1.1.2. Sia Σ un alfabeto e sia $x \in \Sigma^*$. La *potenza n-esima* di x è così definita:

$$\forall x \in \Sigma : \begin{cases} x^n := \varepsilon & \text{se } n = 0 \\ x^n := x^{n-1} \circ x & \text{se } n > 0 \end{cases}$$

Definizione 1.1.3. Sia Σ un alfabeto. La *lunghezza* di una stringa $|\ | : \Sigma^* \to \mathbb{N}$ è così definita:

1. $|\varepsilon| = 0$;
2. $\forall a \in \Sigma, x \in \Sigma^* : |xa| = |x| + 1$

Definizione 1.1.4. Dato un alfabeto Σ, un insieme $L \subseteq \Sigma^*$ è detto *linguaggio*.

Osservazione 1.1.2. *Tutte le operazioni su insiemi possono essere definite per i linguaggi.*

Definizione 1.1.5. Il *complementare* di L rispetto a Σ^* è il linguaggio $\bar{L} = \Sigma^* \setminus L$.

Definizione 1.1.6. Siano $L_1 \subseteq {\Sigma_1}^*$ e $L_2 \subseteq {\Sigma_2}^*$. La *concatenazione* di L_1 e L_2 ($\circ$) è così definita:

$$L_1 \circ L_2 = \{xy \mid x \in L_1, y \in L_2\}$$

L'operazione di concatenazione tra linguaggi gode di alcune proprietà:

- $L_1 \circ L_2$ è un linguaggio su $(\Sigma_1 \cup \Sigma_2)$;
- $\forall L_1, L_2, L_3 \subseteq \Sigma^* : (L_1 \circ L_2) \circ L_3 = L_1 \circ (L_2 \circ L_3)$ (proprietà associativa);

Teorema 1.1.3. $L_1 \subseteq L_1 \circ L_2 \quad \Leftrightarrow \quad \varepsilon \in L_2$

Dimostrazione.

$\Leftarrow$ banale

$\Rightarrow$ Sia $x \in L_1$ una stringa di lunghezza minimale di L_1, ovvero

$$|x| = \min\{|y| \mid y \in L_1\}$$

Si ha che $\exists x_1 \in L_1, x_2 \in L_2 : x = x_1 \circ x_2$.
Supponiamo per assurdo che $x_2 \neq \varepsilon$, ovvero che $|x_2| > 0$:

$$|x| = |x_1| + |x_2| \quad \Rightarrow \quad |x| > |x_1|$$

Il che è assurdo per ipotesi, per cui necessariamente $x_2 = \varepsilon$.

□

Proposizione 1.1.4.

1. $\forall L \subseteq \Sigma^* : L \circ \{\varepsilon\} = L$
2. $\forall L \subseteq \Sigma^* : L \circ \varnothing = \varnothing$
3. $(\mathcal{P}(\Sigma^*), \circ, \{\varepsilon\})$ *è un monoide.*

Definizione 1.1.7. Dato $L \subseteq \Sigma^*$, la *potenza n-esima* di L, con $n \geq 0$, è definita come:

$$\begin{cases} L^0 := \{\varepsilon\} \\ L^n := L \circ L^{n-1} \quad \text{se } n > 0 \end{cases}$$

Definizione 1.1.8. Dato $L \subseteq \Sigma^*$, la *chiusura* (o *star*) *di Kleene* di L denotata L^* è definita come:

$$L^* = \bigcup_{n=0}^{\infty} L^n$$

Osservazione 1.1.5. *Alcune proprietà:*

1. $\varnothing^* = \{\varepsilon\}$
2. $\{\varepsilon\}^* = \{\varepsilon\}$
3. $\exists x \in L : x \neq \varepsilon \quad \Rightarrow \quad |L^*| = |\mathbb{N}|$
4. $\forall L \subseteq \Sigma^* : \varepsilon \in L^*$

Nota: questa definizione dell'operatore * può essere anche applicata quando $L = \Sigma$ per definire Σ^*.

Definizione 1.1.9. Dato $L \subseteq \Sigma^*$, la *chiusura positiva di Kleene* di L denotata L^+ è definita come:

$$L^+ = \bigcup_{n=1}^{\infty} L^n$$

Osservazione 1.1.6. *Alcune proprietà:*

1. $\varnothing^+ = \varnothing$
2. $\{\varepsilon\}^+ = \{\varepsilon\}$
3. $\varepsilon \in L^+ \quad \Leftrightarrow \quad \varepsilon \in L$
4. $L^* = L^+ \cup L^0 = L^+ \cup \{\varepsilon\}$
5. $L^+ = LL^* = L^*L$
6. $\varepsilon \notin \Sigma^+$

1.2 Rappresentazioni finite di linguaggi infiniti

I metodi usati per descrivere linguaggi infiniti in maniera finita sono:

- metodi **riconoscitivi**: un riconoscitore è un dispositivo che prende in input una stringa e risponde se $x \in L$ o $x \notin L$;
- metodi **generativi**: una grammatica è un insieme finito di simboli e regole che generano tutte le parole di un dato linguaggio L.

1.3 Riconoscitori

Un riconoscitore è formato da:

- un **nastro** (infinito o semi infinito) diviso in *celle*; in ogni cella vi può essere un carattere appartenente a un alfabeto (finito) detto *alfabeto di nastro*;
- il **controllo**, ovvero lo *stato* del riconoscitore (appartenente a un insieme finito di stati);
- la **testina**, oggetto che può leggere e/o scrivere in una cella alla volta e può muoversi; in base alle capacità della testina, si possono definire diversi tipi di riconoscitori (per es. *read only*, *read and write*, *one way*, *two way*);
- **memoria ausiliaria**, come code, pile, altri nastri, ecc.

Nota: È possibile dimostrare che gli automi a stati finiti 2-way sono equivalenti al rispettivo modello 1-way.

I riconoscitori con ε-*move* o ε-*transizioni* (equivalenti ai rispettivi automi semplici) possono non muovere la testina nel passaggio da una configurazione all'altra.

Per **configurazione** si intende l'insieme di:

- contenuto del nastro;
- posizione della testina;
- stato del controllo;
- eventuale contenuto della memoria.

Nel caso di riconoscitori 1-way, il contenuto del nastro può essere inteso come il contenuto a destra della testina. È importante evidenziare alcune configurazioni particolari:

- configurazione **iniziale** di un input x:
  - l'input viene scritto sul nastro a partire dalla prima cella;
  - la testina è sulla prima cella;
  - lo stato è uno stato iniziale q_0.
- configurazione **finale** o in generale **di accettazione**:
  - la testina si trova dopo la prima cella vuota;
  - lo stato q appartiene all'insieme di stati finali F;
  - eventualmente la memoria si trova in una particolare condizione.

Il linguaggio *accettato* o *riconosciuto* è l'insieme di tutte e sole le stringhe x la cui computazione termina in una configurazione finale.

Nota: nelle macchine di Turing si fa differenza tra linguaggi riconosciuti e linguaggi accettati.

$$\{L \text{ riconosciuti}\} \subseteq \{L \text{ accettati}\}$$

I linguaggi riconosciuti da una TM sono detti anche *linguaggi ricorsivamente enumerabili*, poiché è possibile creare una TM che scrive tutte le stringhe del linguaggio in un determinato ordine. I linguaggi accettati da una TM sono detti anche *linguaggi ricorsivi*.

In un riconoscitore *deterministico*, ogni configurazione ha al più una configurazione successiva; la sua computazione può essere rappresentata sotto forma di un ramo.
In un riconoscitore *non deterministico*, ogni configurazione può avere più di una configurazione successiva; la sua computazione può essere rappresentata sotto forma di un albero. Se esiste un ramo accettante, l'input è accettato.
È possibile dimostrare che la classe di linguaggi riconosciuti da DTM è equivalente alla classe di linguaggi riconosciuti da NDTM e che la classe di linguaggi accettati da DTM è equivalente alla classe di linguaggi accettati da NDTM.

Capitolo 2

Linguaggi regolari

Automi a stati finiti deterministici e non deterministici, espressioni regolari, e grammatiche regolari rispettivamente accettano, rappresentano e generano la stessa classe di linguaggi, detti *linguaggi regolari*.
I linguaggi regolari godono di molte proprietà, tra cui la possibilità di essere caratterizzati algebricamente e logicamente.

2.1 Automi a stati finiti

Definizione 2.1.1. Un *automa a stati finiti deterministico* (*ASFD*) è una quintupla $m = (Q, \Sigma, \delta, q_0, F)$, dove:

- Q è l'*insieme* (finito) *degli stati*;
- Σ è l'*alfabeto* (finito) di input;
- $\delta : Q \times \Sigma \to Q$ è la *funzione* (totale) *di transizione*;
- $q_0 \in Q$ è lo *stato iniziale*;
- $F \subseteq Q$ è l'*insieme degli stati finali*.

La funzione di transizione spesso viene rappresentata tramite tabella o diagramma di transizione.

Definizione 2.1.2. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFD. Si definisce la *funzione di transizione estesa* $\delta^* : Q \times \Sigma^* \to Q$ nel seguente modo:

$$\begin{cases} \forall q \in Q : & \delta^*(q, \varepsilon) := q \\ \forall q \in Q, x \in \Sigma^*, a \in \Sigma : & \delta^*(q, xa) := \delta(\delta^*(q, x), a) \end{cases}$$

Proposizione 2.1.1. *δ^* è un'estensione di δ, infatti:*

$$\forall q \in Q, \forall a \in \Sigma : \delta(q, a) = \delta^*(q, a)$$

Definizione 2.1.3. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFD. Il *linguaggio accettato* (o *riconosciuto*) da m è il linguaggio

$$L(m) = \{x \in \Sigma^* \mid \delta^*(q_0, x) \in F\}$$

Definizione 2.1.4. Un linguaggio L è detto *AF-regolare* se esiste un ASFD m tale che $L = L(m)$.

Definizione 2.1.5. Un *automa a stati finiti non deterministico* (*ASFND*) è una quintupla $m = (Q, \Sigma, \delta, q_0, F)$, dove:

- Q è l'*insieme* (finito) *degli stati*;
- Σ è l'*alfabeto* (finito) di input;
- $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ è la *funzione* (totale) *di transizione*;
- $q_0 \in Q$ è lo *stato iniziale*;
- $F \subseteq Q$ è l'*insieme degli stati finali.*

Definizione 2.1.6. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFND. Si definisce la *funzione di transizione estesa* $\delta^* : Q \times \Sigma^* \to \mathcal{P}(Q)$ nel seguente modo:

$$\begin{cases} \forall q \in Q : & \delta^*(q, \varepsilon) := \{q\} \\ \forall q \in Q, x \in \Sigma^*, a \in \Sigma : & \delta^*(q, xa) := \bigcup_{q' \in \delta^*(q,x)} \delta(q', a) \end{cases}$$

Proposizione 2.1.2. *δ^* è un'estensione di δ, infatti:*

$$\forall q \in Q, \forall a \in \Sigma : \delta(q, a) = \delta^*(q, a)$$

Definizione 2.1.7. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFND. Il *linguaggio accettato* (o *riconosciuto*) da m è il linguaggio

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

Definizione 2.1.8. Un linguaggio L è detto *AFN-regolare* se esiste un ASFND m tale che $L = L(m)$.

Teorema 2.1.3. *Se un linguaggio L è AF-regolare, allora L è AFN-regolare.*

Dimostrazione. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFD che accetta L. Definiamo l'ASFND $\widetilde{m} = (Q, \Sigma, \widetilde{\delta}, q_0, F)$, dove $\widetilde{\delta}$ è definita nel seguente modo:

$$\begin{array}{rcl} \widetilde{\delta} : Q \times \Sigma & \to & \mathcal{P}(Q) \\ (q, a) & \mapsto & \{\delta(q, a)\} \end{array}$$

È banale dimostrare che $L(m) = L(\widetilde{m})$. □

Teorema 2.1.4. *Se un linguaggio L è AFN-regolare, allora L è AF-regolare.*

Dimostrazione. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFND che accetta L. Definiamo l'ASFD $\widetilde{m} = (\widetilde{Q}, \Sigma, \widetilde{\delta}, \widetilde{q_0}, \widetilde{F})$, dove:

- $\widetilde{Q} := \mathcal{P}(Q)$
- $\begin{array}{rcl} \widetilde{\delta} : \widetilde{Q} \times \Sigma & \to & \widetilde{Q} \\ (\widetilde{q}, a) & \mapsto & \displaystyle\bigcup_{q \in \widetilde{q}} \delta(q, a) \end{array}$
- $\widetilde{q_0} := \{q_0\}$
- $\widetilde{F} := \{\widetilde{q_f} \in \widetilde{Q} \mid \widetilde{q_f} \cap F \neq \varnothing\}$

Si dimostra che $L(m) = L(\widetilde{m})$. □

Corollario 2.1.5.

$$\{L \mid L \textit{ AF-regolare}\} \equiv \{L \mid L \textit{ AFN-regolare}\}$$

Proposizione 2.1.6. *Esiste un linguaggio $L \subseteq \Sigma^*$ riconosciuto da un ASFND $m = (Q, \Sigma, \delta, q_0, F)$ tale che per ogni ASFD $\widetilde{m} = (\widetilde{Q}, \Sigma, \widetilde{\delta}, \widetilde{q_0}, \widetilde{F})$ si ha:*

$$|\widetilde{Q}| \geq 2^{|Q|}$$

Definizione 2.1.9. Un *automa a stati finiti non deterministico con ε-transizioni* è una quintupla $m = (Q, \Sigma, \delta, q_0, F)$, dove:

- Q è l'*insieme* (finito) *degli stati*;
- Σ è l'*alfabeto* (finito) di input;
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \to \mathcal{P}(Q)$ è la *funzione* (totale) *di transizione*;
- $q_0 \in Q$ è lo *stato iniziale*;
- $F \subseteq Q$ è l'*insieme degli stati finali.*

Definizione 2.1.10. L'insieme degli stati raggiungibili da q mediante un cammino di etichetta "ε" è denotato $\varepsilon\text{-}closure(q)$.

Definizione 2.1.11. Sia $P \subseteq Q$, si definisce l'$\varepsilon\text{-}closure$ di P nel seguente modo:

$$\varepsilon\text{-}closure(P) = \bigcup_{q \in P} \varepsilon\text{-}closure(q)$$

Definizione 2.1.12. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFND con ε-transizioni. Si definisce la funzione $\widehat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$ nel seguente modo:

$$\begin{cases} \forall q \in Q : & \widehat{\delta}(q, \varepsilon) := \varepsilon\text{-}closure(q) \\ \forall q \in Q, x \in \Sigma^*, a \in \Sigma : & \widehat{\delta}(q, xa) := \varepsilon\text{-}closure(P), \\ & \text{con } P = \{p \mid \exists r \in \widehat{\delta}(q, x) : p \in \delta(r, a)\} \end{cases}$$

Nota: tale funzione è denotata $\widehat{\delta}$ e non δ^* poiché non è un'estensione di δ (ovvero non è detto che $\forall q \in Q, a \in \Sigma : \widehat{\delta}(q, a) = \delta(q, a)$).

Lemma 2.1.7.

$$\varepsilon\text{-}closure\left(\bigcup_{p \in \widehat{\delta}(q,w)} \delta(p, a)\right) = \bigcup_{p \in \widehat{\delta}(q,w)} \widehat{\delta}(p, a)$$

Definizione 2.1.13. Sia $m = (Q, \Sigma, \delta, q_0, F)$ un ASFND con ε-transizioni. Il *linguaggio accettato* (o *riconosciuto*) da m è il linguaggio

$$L(m) = \{x \in \Sigma^* \mid \widehat{\delta}(q_0, x) \cap F \neq \varnothing\}$$

Teorema 2.1.8. *Se un linguaggio L è accettato da un ASFND con ε-transizioni, allora esiste un ASFND accettante L.*

Dimostrazione. Sia $m = (Q, \Sigma, \delta, q_0, F)$ ASFND con ε-transizionitale che $L = L(m)$. Si consideri $m' = (Q, \Sigma, \delta', q_0, F')$ ASFND, dove:

- $\begin{array}{rcl} \delta' : Q \times \Sigma & \to & \mathcal{P}(Q) \\ (q, a) & \mapsto & \{\widehat{\delta}(q, a)\} \end{array}$
- $F' = \begin{cases} F \cup \{q_0\} & \text{se } \varepsilon\text{-}closure(q_0) \cap F \neq \varnothing \\ F & \text{altrimenti} \end{cases}$

Dimostriamo innanzitutto per induzione[1] sulla lunghezza di x che

$$\forall q \in Q : \forall x \neq \varepsilon : \quad \delta'^*(q,x) = \widehat{\delta}(q,x)$$

PB: $|x| = 1 \quad \Rightarrow \quad x = a, a \in \Sigma$
$\delta'^*(q,a) = \delta'(q,a) = \widehat{\delta}(q,a)$

PI: Supponiamo che la tesi sia vera per stringhe di lunghezza n, con $n \geq 1$. Sia $x = wa$, con $|w| = n$, $a \in \Sigma$.

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

$$= \bigcup_{p \in \widehat{\delta}(q,w)} \widehat{\delta}(p,a) = \varepsilon\text{-}closure\left(\bigcup_{p \in \widehat{\delta}(q,w)} \delta(p,a) \right) = \widehat{\delta}(q_0,wa) = \widehat{\delta}(q_0,x)$$

Dimostriamo ora che $\forall x \in \Sigma^* : x \in L(m) \Leftrightarrow x \in L(m')$, ovvero:

$$\widehat{\delta}(q_0,x) \cap F \neq \varnothing \quad \Leftrightarrow \quad \delta'^*(q_0,x) \cap F' \neq \varnothing$$

- per $x = \varepsilon$ si ha:

$$\widehat{\delta}(q_0,x) = \varepsilon\text{-}closure(q_0) \qquad \delta'^*(q_0,x) = \{q_0\}$$

  $\Rightarrow$ Se ε-$closure(q_0) \cap F \neq \varnothing$, allora $F' = F \cup \{x_0\}$, per cui $q_0 \in F'$ e $\{q_0\} \cap F' \neq \varnothing$.

  $\Leftarrow$ $\{q_0\} \cap F' \neq \varnothing \quad \Rightarrow \quad q_0 \in F \quad \vee \quad \varepsilon\text{-}closure(q_0) \cap F \neq \varnothing$
  Se $q_0 \in F$, allora ε-$closure(q_0) \cap F \neq \varnothing$ poiché $q_0 \in \varepsilon$-$closure(q_0)$.
  Per cui in generale si ha ε-$closure(q_0) \cap F \neq \varnothing$.

- per $x \neq \varepsilon$

  $\Rightarrow$ $\widehat{\delta}(q_0,x) \cap F \neq \varnothing \quad \Rightarrow \quad \delta'^*(q_0) \cap F \neq \varnothing \quad \Rightarrow \quad \delta'^*(q_0) \cap F' \neq \varnothing$

  $\Leftarrow$ $\delta'^*(q_0,x) \cap F' \neq \varnothing \quad \Rightarrow \quad \widehat{\delta}(q_0,x) \cap F' \neq \varnothing$
  Supponiamo che $\left(\widehat{\delta}(q_0,x) \cap F'\right) \cap F = \varnothing$: allora si ha
  $\widehat{\delta}(q_0,x) \cap F' = \{q_0\}, \quad q_0 \notin F, \quad q_0 \in F', \quad q_0 \in \widehat{\delta}(q_0,x).$

$$q_0 \notin F, q_0 \in F' \quad \Rightarrow \quad \varepsilon\text{-}closure(q_0) \cap F \neq \varnothing$$

[1] **PB**: passo base, **PI**: passo induttivo. La presente notazione sarà usata anche in successive dimostrazioni.

$$q_0 \in \widehat{\delta}(q_0, x) \quad \Rightarrow \quad \varepsilon\text{-}closure(q_0) \subseteq \widehat{\delta}(q_0, x) \quad \Rightarrow \quad \widehat{\delta}(q_0, x) \cap F \neq \varnothing$$

Il che è assurdo poiché $\widehat{\delta}(q_0, x) \cap F = \varnothing$.
Per cui necessariamente si ha $\left(\widehat{\delta}(q_0, x) \cap F'\right) \cap F \neq \varnothing$.

$$\left(\widehat{\delta}(q_0, x) \cap F'\right) \cap F \neq \varnothing \quad \Rightarrow \quad \widehat{\delta}(q_0, x) \cap F \neq \varnothing$$

□

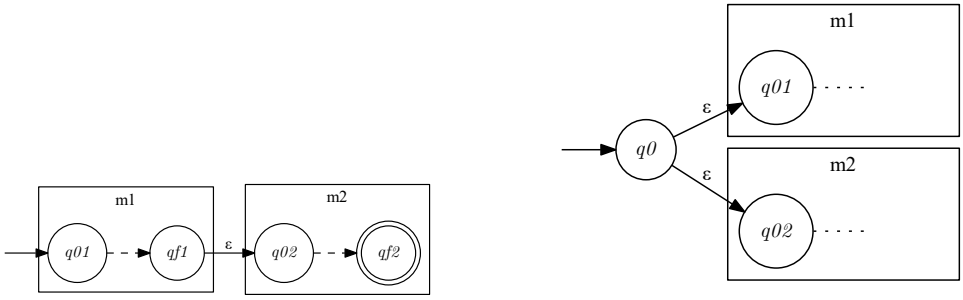
Teorema 2.1.9. *Se $L_1, L_2 \subseteq \Sigma^*$ sono AF-regolari, allora $L_1 \circ L_2$ è AF-regolare.*

Dimostrazione. Siano $m_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ e $m_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ ASFND con ε-transizionitali che $Q_1 \cap Q_2 = \varnothing$, $L_1 = L(m_1)$ e $L_2 = L(m_2)$. Si consideri $m_3 = (Q_3, \Sigma, \delta_3, q_1, F_2)$ ASFND con ε-transizioni(fig. 2.1a), dove:

- $Q_3 = Q_1 \cup Q_2$
- $\delta_3 : (Q_1 \cup Q_2) \times \Sigma \to (Q_1 \cup Q_2)$

$$\forall q \in (Q_1 \cup Q_2), a \in \Sigma : \quad \delta_3(q, a) = \begin{cases} \delta_1(q, a) & \text{se } q \in Q_1 \setminus F_1 \\ \delta_1(q, a) \cup \{q_2\} & \text{se } q \in F_1 \\ \delta_2(q, a) & \text{se } q \in Q_2 \end{cases}$$

È banale dimostrare che $L_1 \circ L_2 = L(m_3)$ □



(a) $m : L(m) = L(m_1) \circ L(m_2)$ (b) $m : L(m) = L(m_1) \cup L(m_2)$

Figura 2.1: Composizione di automi a stati finiti

Teorema 2.1.10. *Se $L \subseteq \Sigma^*$ è AF-regolare, allora $\Sigma^* \setminus L$ è AF-regolare.*

Dimostrazione. Sia $m = (Q, \Sigma, \delta, q_0, F)$ ASFD tale che $L = L(m)$. Sia $\overline{m} = (Q, \Sigma, \delta, q_0, Q \setminus F)$ ASFD.

$$x \in L(\overline{m}) \quad \Leftrightarrow \quad x \notin L(m)$$

□

Nota: tale teorema non è dimostrabile direttamente per ASFND in quanto non è detto che la computazione di una stringa legga tutti i caratteri in input.

Teorema 2.1.11. *Se $L_1, L_2 \subseteq \Sigma^*$ sono AF-regolari, allora $L_1 \cup L_2$ è AF-regolare.*

Dimostrazione. Siano $m_1 = (Q_1, \Sigma_1, \delta_1, q_1, F_1)$ e $m_2 = (Q_2, \Sigma_2, \delta_2, q_2, F_2)$ ASFND con ε-transizionitali che $Q_1 \cap Q_2 = \varnothing$, $L_1 = L(m_1)$ e $L_2 = L(m_2)$. Si consideri $m_3 = (Q_3, \Sigma, \delta_3, q_0, F_3)$ ASFND con ε-transizioni(fig. 2.1b), dove:

- $Q_3 = Q_1 \cup Q_2 \cup \{q_0\}$
- $\delta_3 : Q_3 \times \Sigma \to Q_3$

$$\forall q \in Q_3, a \in \Sigma : \quad \delta_3(q,a) = \begin{cases} \{q_1, q_2\} & \text{se } q = q_0 \\ \delta_1(q,a) & \text{se } q \in Q_1 \\ \delta_2(q,a) & \text{se } q \in Q_2 \end{cases}$$

- $F_3 = F_1 \cup F_2$

È banale dimostrare che $L_1 \cup L_2 = L(m_3)$ □

Dimostrazione. Siano $m_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$ e $m_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$ ASFD tali che $L_1 = L(m_1)$ e $L_2 = L(m_2)$. Definiamo l'ASFD $m_3 = (Q_3, \Sigma, \delta_3, q_{03}, F_3)$, dove:

- $Q_3 = Q_1 \times Q_2$
- $$\begin{array}{rcl} \delta_3 : (Q_1 \times Q_2) \times \Sigma & \to & (Q_1 \times Q_2) \\ ((q_1, q_2), a) & \mapsto & (\delta_1(q_1, a), \delta_2(q_2, a)) \end{array}$$
- $q_{03} = (q_{01}, q_{02})$
- $F_3 = \{(q_1, q_2) \mid q_1 \in Q_1 \vee q_2 \in Q_2\}$

È banale dimostrare che $L_1 \cup L_2 = L(m_3)$. □

Teorema 2.1.12. *Se $L_1, L_2 \subseteq \Sigma^*$ sono AF-regolari, allora $L_1 \cap L_2$ è AF-regolare.*

Dimostrazione.

$$L_1 \cap L_2 = \Sigma^* \setminus ((\Sigma^* \setminus L_1) \setminus (\Sigma^* \setminus L_2))$$

□

Dimostrazione. Siano $m_1 = (Q_1, \Sigma, \delta_1, q_{01}, F_1)$ e $m_2 = (Q_2, \Sigma, \delta_2, q_{02}, F_2)$ ASFD tali che $L_1 = L(m_1)$ e $L_2 = L(m_2)$. Definiamo l'ASFD $m_3 = (Q_3, \Sigma, \delta_3, q_{03}, F_3)$, dove:

- $Q_3 = Q_1 \times Q_2$
- $\begin{array}{rcl} \delta_3 : (Q_1 \times Q_2) \times \Sigma & \to & (Q_1 \times Q_2) \\ ((q_1, q_2), a) & \mapsto & (\delta_1(q_1, a), \delta_2(q_2, a)) \end{array}$
- $q_{03} = (q_{01}, q_{02})$
- $F_3 = F_1 \times F_2$

È banale dimostrare che $L_1 \cap L_2 = L(m_3)$. □

Teorema 2.1.13. *Se $L \subseteq \Sigma^*$ è AF-regolare, allora L^* è AF-regolare.*

Teorema 2.1.14. *$\varnothing, \{a\}$ con $a \in \Sigma, \{\varepsilon\}$ sono AF-regolari.*

Teorema 2.1.15. *Un ASFD o un ASFND accetta la parola vuota se e solo se $q_0 \in F$.*

2.2 Espressioni regolari

Definizione 2.2.1. Sia Σ un alfabeto. Le *espressioni regolari* su Σ sono definite come segue:

1. $\underline{\varnothing}$ è una espressione regolare;
2. $\underline{\varepsilon}$ è una espressione regalare;
3. per ogni $a \in \Sigma$, $\underline{a}$ è una espressione regolare;
4. se α e β sono espressioni regolari, allora $(\alpha \cup \beta)$, $\alpha \circ \beta$ e α^* sono espressioni regolari.

Definizione 2.2.2. Il linguaggio denotato da una espressione regolare α si indica con $\langle \alpha \rangle$ ed è così definito:

1. $\langle \underline{\varnothing} \rangle := \varnothing$
2. $\langle \underline{\varepsilon} \rangle := \{\varepsilon\}$
3. $\langle \underline{a} \rangle := \{a\}$
4. $\langle (\alpha \cup \beta) \rangle := \langle \alpha \rangle \cup \langle \beta \rangle$

5. $\langle \alpha \circ \beta \rangle := \langle \alpha \rangle \circ \langle \beta \rangle$
6. $\langle \alpha^* \rangle := \langle \alpha \rangle^*$

Definizione 2.2.3. Un linguaggio L è detto *regolare* se esiste un'espressione regolare α che lo denota, cioè tale che $\langle \alpha \rangle = L$.

Teorema 2.2.1. *Se un linguaggio è finito, allora è regolare.*

Teorema 2.2.2. *Se L è regolare, allora L è AF-regolare.*

Dimostrazione. Se α è una espressione regolare, denotiamo il numero di operatori che compaiono in essa con $n(\alpha)$. La funzione $n : A \to \mathbb{N}$, dove A è l'insieme delle espressioni regolari, può essere definita più formalmente nel seguente modo:

$$n(\alpha) := \begin{cases} 0 & \text{se } \alpha \in \{\underline{\varnothing}, \underline{\varepsilon}, \underline{a}\} \\ n(\beta) + n(\gamma) + 1 & \text{se } \alpha \in \{(\beta \cup \gamma), \beta \circ \gamma\} \\ n(\beta) + 1 & \text{se } \alpha = \beta^* \end{cases}$$

Sia α una espressione regolare tale che $L = \langle \alpha \rangle$. Dimostriamo che L è AF-regolare per induzione sul numero di operatori $n(\alpha)$ che compaiono in α.

PB: se $n(\alpha) = 0$, allora α non contiene operatori, per cui $\alpha \in \{\underline{\varnothing}, \underline{\varepsilon}, \underline{a}\}$, con $a \in \Sigma$. In ogni caso L è AF-regolare.

PI: Sia $n(\alpha) > 0$, allora $\alpha \in \{(\beta \cup \gamma), \beta \circ \gamma, \beta^*\}$, dove $n(\beta) < n(\alpha)$ e $n(\gamma) < n(\alpha)$. Per ipotesi induttiva $\langle \beta \rangle$ e $\langle \gamma \rangle$ sono linguaggi AF-regolari. Quindi:

- se $\alpha = (\beta \cup \gamma)$, si ha $\langle \alpha \rangle = \langle (\beta \cup \gamma) \rangle = \langle \beta \rangle \cup \langle \gamma \rangle$;
- se $\alpha = \beta \circ \gamma$, si ha $\langle \alpha \rangle = \langle \beta \circ \gamma \rangle = \langle \beta \rangle \circ \langle \gamma \rangle$;
- se $\alpha = \beta^*$, si ha $\langle \alpha \rangle = \langle \beta^* \rangle = \langle \beta \rangle^*$.

Per le proprietà dei linguaggi AF-regolari, in ogni caso $\langle \alpha \rangle$ è AF-regolare.

□

Teorema 2.2.3 (Teorema di Kleene)**.** *Se L è AF-regolare, allora L è regolare.*

Dimostrazione. Sia $m = (Q, \Sigma, \delta, q_1, F)$ ASFD tale che $L = L(m)$ e sia $Q = \{q_1, q_2, \ldots, q_n\}$.
Definiamo, per ogni $k \in \{0, \ldots, n\}$, $i, j \in \{1, \ldots, n\}$, l'insieme R_{ij}^k come l'insieme di tutte le stringhe che etichettano cammini che vanno da q_i a q_j attraversando solo stati q_x con $x \leq k$. Se R_{ij}^k è regolare, indichiamo con r_{ij}^k l'espressione che lo denota.

$$R_{ij}^0 = \begin{cases} \{a \in \Sigma \mid \delta(q_i, a) = q_j\} & \text{se } i \neq j \\ \{a \in \Sigma \mid \delta(q_i, a) = q_j\} \cup \{\varepsilon\} & \text{se } i = j \end{cases}$$

$$R_{ij}^k = \{x \in \Sigma^* \mid \delta^*(q_i, x) = q_j \quad \wedge \\ \forall y \in \Sigma^+ : \quad (\exists z \in \Sigma^+ : x = yz) \Rightarrow (\exists h \leq k : \delta^*(q_i, y) = q_h)\}$$

Si ha che $L = \bigcup_{q_j \in F} R_{1j}^n$.
Dimostriamo per induzione su k che $\forall k, i, j : R_{ij}^k$ è regolare.

PB: Per $k = 0$

$$R_{ij}^k = \begin{cases} \{a_{i_1}, a_{i_2}, \ldots, a_{i_s}\} = \left\langle \underline{a_{i_1}} \cup \underline{a_{i_2}} \cup \ldots \cup \underline{a_{i_s}} \right\rangle := \left\langle r_{ij}^0 \right\rangle & \text{se } i \neq j \\ \{a_{i_1}, a_{i_2}, \ldots, a_{i_s}\} \cup \{\varepsilon\} = \left\langle \underline{a_{i_1}} \cup \underline{a_{i_2}} \cup \ldots \cup \underline{a_{i_s}} \cup \underline{\varepsilon} \right\rangle := \left\langle r_{ij}^0 \right\rangle & \text{se } i = j \end{cases}$$

PI: Sia $k > 0$. Per ipotesi induttiva R_{ij}^{k-1} è regolare per ogni i, j. Si ha:

$$R_{ij}^k = R_{ij}^{k-1} \cup R_{ik}^{k-1} \left(R_{kk}^{k-1}\right)^* R_{kj}^{k-1}$$

$$r_{ij}^k = r_{ij}^{k-1} \cup r_{ik}^{k-1} \left(r_{kk}^{k-1}\right)^* r_{kj}^{k-1}$$

□

2.3 Pumping Lemma e sue conseguenze

Lemma 2.3.1 (Pumping Lemma)**.** *Sia $L = L(m)$, con m ASFD con n stati. Sia $x \in L$, con $|x| \geq n$. Allora, come esemplificato in fig. 2.2:*

$$\exists u, v, w \in \Sigma^* : x = uvw, |uv| \leq n, v \neq \varepsilon, \quad \forall i \geq 0 : uv^i w \in L$$

Nota*: non vale il teorema inverso, ovvero: esistono linguaggi non riconosciuti da nessun ASFD che soddisfano la tesi del teorema.*

Teorema 2.3.2. *Sia m un ASFD con n stati. Allora*

$$L(m) \neq \varnothing \quad \Leftrightarrow \quad \exists x \in L(m) : |x| < n$$

Dimostrazione.

$\Leftarrow$ banale

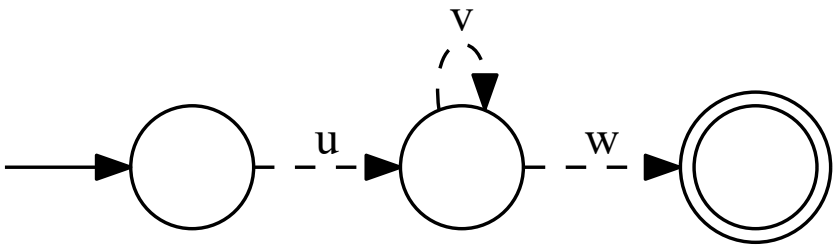


Figura 2.2

$\Rightarrow$ Sia x una parola di lunghezza minimale appartenente a $L(m)$. Supponiamo per assurdo che $|x| \geq n$.
Applicando il Pumping Lemma:

$$\exists u, v, w \in \Sigma^* : x = uvw, |uv| \leq n, v \neq \varepsilon, \quad \forall i \geq 0 : uv^i w \in L(m)$$

Per $i = 0$ si ha $uw \in L(m)$; tuttavia $|uw| < |uvw| = |x|$, il che è assurdo.

□

Corollario 2.3.3. *È decidibile stabilire se $L(m) \neq \varnothing$ o meno.*

Teorema 2.3.4. *Sia m un ASFD con n stati. Allora*

$$|L(m)| = |\mathbb{N}| \quad \Leftrightarrow \quad \forall a \geq n : \exists x \in L(m) : a \leq |x| < a + n$$

Dimostrazione.

$\Leftarrow$ Per ipotesi $x \in L(m)$, con $|x| \geq a \geq n$. Applicando il Pumping Lemma:

$$\exists u, v, w \in \Sigma^* : x = uvw, |uv| \leq n, v \neq \varepsilon, \quad \forall i \geq 0 : uv^i w \in L(m)$$

Per cui $|L(m)| = |\mathbb{N}|$.

$\Rightarrow$ Poiché $L(m)$ è infinito, esso contiene certamente parole di lunghezza $\geq a$. In questo insieme si consideri ora una stringa x di lunghezza minimale. Supponiamo per assurdo che $|x| \geq a + n$.

$$\exists x_1, x_2 \in \Sigma^* : |x_1| = a, |x_2| \geq n, x_1 x_2 = x$$

Applicando il Pumping Lemma al linguaggio $\{y \in \Sigma^* \mid x_1 y \in L(m)\}$ (di cui è possibile dimostrare la regolarità):

$$\exists u, v, w \in \Sigma^* : x_2 = uvw, |uv| \leq n, v \neq \varepsilon, \quad \forall i \geq 0 : x_1 uv^i w \in L(m)$$

Quindi si ha $x_1 uw \in L(m)$ e inoltre:

$$a = |x_1| \leq |x_1 uw| < |x_1 uvw| = |x| < a + n$$

Per cui x non è stringa di lunghezza $\geq a$ di lunghezza minimale.

□

Corollario 2.3.5. *È decidibile stabilire se $|L(m)| = |\mathbb{N}|$ o meno.*

Corollario 2.3.6. *È decidibile stabilire se due ASFD sono equivalenti (cioè accettano lo stesso linguaggio) o meno.*

Dimostrazione. Siano m_1 e m_2 ASFD tali che $L_1 = L(m_1), L_2 = L(m_2)$.
Sia $X = (L_1 \cap \bar{L}_2) \cup (\bar{L}_1 \cap L_2)$. Allora $L_1 = L_2 \Leftrightarrow X = \varnothing$. Poiché X è AF-regolare ed è possibile costruire un ASFD che lo accetta, la tesi è verificata. □

2.4 Relazioni R_L e R_m

Definizione 2.4.1. Sia $L \subseteq \Sigma^*$. Scriviamo $x \; R_L \; y$ (con $x, y \in \Sigma^*$) se

$$\forall z \in \Sigma^* : xz \in L \quad \Leftrightarrow \quad yz \in L$$

Esempio 2.4.1. Siano $\Sigma = \{0, 1\}$ e $L = 0^*10^*$.

1. Siano $x = 01, y = 011$
   per $z = 0 : \; xz = 010 \in L, \quad yz = 0110 \notin L$
   $\exists z \in \Sigma^* : xz \in L \Leftrightarrow xy \notin L$, per cui $01 \not R_L \; 011$.

2. Siano $a, b \geq 0$ e siano $x = 0^a, y = 0^b$

   - se $z \in 0^*$, allora $xz \notin L \wedge yz \notin L$;
   - se $z \in 0^*10^*$, allora $xz \in L \wedge yz \in L$;
   - se $z \in 0^*10^*1(0 \cup 1)^*$, allora $xz \notin L \wedge yz \notin L$.

   Per cui $0^a \; R_L \; 0^b$.

3. In generale, siano $A = 0^*$, $B = 0^*10^*$, $C = 0^*10^*1(0 \cup 1)^*$; si ha:

   $$\forall I, J \in \{A, B, C\}, I \neq J : \; \forall x, y \in I : x \; R_L \; y \; \wedge \; \forall x \in I, y \in J : x \not R_L \; y$$

Osservazione 2.4.1. *Possiamo verificare che per ogni linguaggio L, R_L è una relazione di equivalenza, in quanto gode delle proprietà riflessiva, simmetrica e transitiva.*

Definizione 2.4.2. Sia m ASFD. Scriviamo $x\ R_m\ y$ (con $x, y \in \Sigma^*$) se

$$\delta^*(q_0, x) = \delta^*(q_0, y)$$

È possibile dimostrare che R_L e R_m sono relazioni di equivalenza, ovvero godono delle proprietà riflessiva, simmetrica e transitiva.
Indichiamo una classe $\mathcal{C}$ di una relazione di equivalenza R con $[x]_R$, dove $x \in \mathcal{C}$. L'indice di una relazione di equivalenza i_R è il numero di classi generate dalla relazione stessa:

$$i_R := \{\mathcal{C} \mid \forall a, b : a, b \in \mathcal{C} \Leftrightarrow a\ R\ b\}$$

Definizione 2.4.3. Sia E relazione di equivalenza definita su Σ^*. Diciamo che è *invariante a destra* (rispetto alla concatenazione) se:

$$\forall x, y \in \Sigma^* : x\ E\ y \quad \Rightarrow \quad \forall z \in \Sigma^* : xz\ E\ yz$$

Lemma 2.4.2. *La relazione R_L è invariante a destra.*

Dimostrazione. Siano $x, y \in \Sigma^*$ tali che $x\ R_L\ y$.

$$\forall z, w \in \Sigma^* : x(zw) \in L \quad \Leftrightarrow \quad y(zw) \in L$$

$$\forall z, w \in \Sigma^* : (xz)w \in L \quad \Leftrightarrow \quad (yz)w \in L$$

Per cui $\forall z \in \Sigma^* : xz\ R_L\ yz$. □

Lemma 2.4.3. *La relazione R_m è invariante a destra.*

Dimostrazione. Siano $x, y \in \Sigma^*$ tali che $x\ R_m\ y$.

$$\delta^*(q_0, x) = \delta^*(q_0, y)$$

$$\forall z \in \Sigma^* : \delta^*(q_0, xz) = \delta^*(\delta^*(q_0, x), z) = \delta^*(\delta^*(q_0, y), z) = \delta^*(q_0, yz)$$

Per cui $\forall z \in \Sigma^* : xz\ R_m\ yz$. □

Teorema 2.4.4 (Teorema di Myhill-Nerode)**.** *Sia $L \subseteq \Sigma^*$. Le seguenti affermazioni sono equivalenti:*

1. *L è regolare;*
2. *L è unione di alcune classi di equivalenza di una relazione di equivalenza E invariante a destra e di indice finito;*

3. R_L ha indice finito.

Dimostrazione.

$1 \Rightarrow 2$ L è regolare implica che esiste $m = (Q, \Sigma, \delta, q_0, F)$ ASFD tale che $L = L(m)$. Poiché per ogni classe $\mathcal{C}$ di R_m vale

$$\exists q \in Q : \forall x \in \Sigma^* : x \in \mathcal{C} \Leftrightarrow \delta^*(q_0, x) = q$$

si ha $i_{R_m} \leq |Q|$. Inoltre

$$L = \bigcup_{\delta^*(q_0,x) \in F} [x]_{R_m}$$

$2 \Rightarrow 3$ Dimostriamo che la relazione E è più fine rispetto a R_L, cioè che $\forall x \in \Sigma^* : [x]_E \subseteq [x]_{R_L}$. Allora avremo $i_{R_L} \leq i_E$.

Dimostriamo che $x \; E \; y \quad \Rightarrow \quad x \; R_L \; y$.
Siano $x, y \in \Sigma^*$ tali che $x \; E \; y$. Poiché E è invariante a destra possiamo affermare che $\forall z \in \Sigma^* : xz \; E \; yz$. Ma xz, yz appartengono alla stessa classe di E, e L è unione di classi di E, quindi:

$$\forall z \in \Sigma^* : xz \in L \quad \Leftrightarrow \quad yz \in L$$

Per cui $x \; R_L \; y$.

Nota: è possibile dimostrare che R_m è un raffinamento di E oppure E è un raffinamento di R_m, dove $L = L(m)$.

$3 \Rightarrow 1$ Sia $\bar{m} = (\bar{Q}, \Sigma, \bar{\delta}, \bar{q}_0, \bar{F})$ ASFD dove:

- $\bar{Q} = \{[x]_{R_L} \mid x \in \Sigma^*\}$
- $\bar{\delta} : \bar{Q} \times \bar{\Sigma} \;\; \rightarrow \;\; \bar{Q}$
  $\quad\quad [x]_{R_L}, a \;\; \mapsto \;\; [xa]_{R_L}$
- $\bar{q}_0 = [\varepsilon]_{R_L}$
- $\bar{F} = \{[x]_{R_L} \mid x \in L\}$

Verifichiamo che la definizione di $\bar{\delta} : \bar{Q} \times \bar{\Sigma} \rightarrow \bar{Q}$ è consistente, ovvero $x \; R_L \; y \Rightarrow \bar{\delta}([x]_{R_L}, a) = \bar{\delta}([y]_{R_L}, a)$, cioè $[xa]_{R_L} = [ya]_{R_L}$. Ma questo è vero poiché R_L è invariante a destra, per cui $xa \; R_L \; ya$. Si può infine dimostrare per induzione sulla lunghezza di x che $\bar{\delta}^*([\varepsilon]_{R_L}) = [x]_{R_L}$, per cui:

$$x \in L(\bar{m}) \;\Leftrightarrow\; \bar{\delta}^*(\bar{q}_0, x) \in \bar{F} \;\Leftrightarrow$$
$$\Leftrightarrow\; \bar{\delta}^*([\varepsilon]_{R_L}, x) \in \bar{F} \;\Leftrightarrow\; [x]_{R_L} \in \bar{F} \;\Leftrightarrow\; x \in L$$

□

2.5 Minimizzazione di automi a stati finiti

Teorema 2.5.1. *Dato un linguaggio regolare L, l'ASFD minimale accettante L è unico (a meno di isomorfismi) ed è l'automa $\bar{m}$ costruibile secondo il Teorema di Myhill-Nerode.*

Dimostrazione. Dimostriamo innanzitutto che l'automa $\bar{m}$ è l'ASFD minimale accettante L. Sia $m = (Q, \Sigma, \delta, q_0, F)$ ASFD tale che $L(m) = L$. Si ha

$$|\bar{m}| = i_{R_L} \leq i_{R_m} \leq |Q|$$

Dimostriamo ora che tale automa è unico, ovvero qualunque automa minimale accettante L è isomorfo a $\bar{m}$. Sia $m = (Q, \Sigma, \delta, q_0, F)$ ASFD accettante L minimale. Allora $|Q| = |\bar{Q}|$.
Sia $q \in Q$; poiché m è minimale, $\exists x \in \Sigma^* : \delta^*(q_0, x) = q$. Definiamo quindi la funzione $f : Q \to \bar{Q}$ tale che $\forall q \in Q : f(q) = [x]_{R_L}$, dove $\delta^*(q_0, x) = q$.
Dimostriamo che f è ben definita, ovvero che

$$\forall q \in Q, x, y \in \Sigma^* : \quad \delta^*(q_0, x) = \delta^*(q_0, y) = q \quad \Rightarrow \quad [x]_{R_L} = [y]_{R_L}$$

Poiché $\delta^*(q_0, x) = \delta^*(q_0, y)$, allora $x \ R_m \ y$; siccome R_m è più fine di R_L, si ha $x \ R_L \ y$, ovvero $[x]_{R_L} = [y]_{R_L}$.
Possiamo osservare che la funzione è suriettiva, in quanto ogni classe di R_L ha un rappresentante x, per cui $\forall x \in \Sigma^* : \exists q \in Q : \delta^*(q_0, x) = q$. Poiché $|Q| = |\bar{Q}|$, la funzione è biunivoca.

Dimostriamo infine che

$$\forall p \in Q, a \in \Sigma : \quad \delta(p, a) = q \quad \Rightarrow \quad \bar{\delta}(f(p), a) = f(q)$$

Sia $f(p) = [x]_{R_L}$. Si ha:

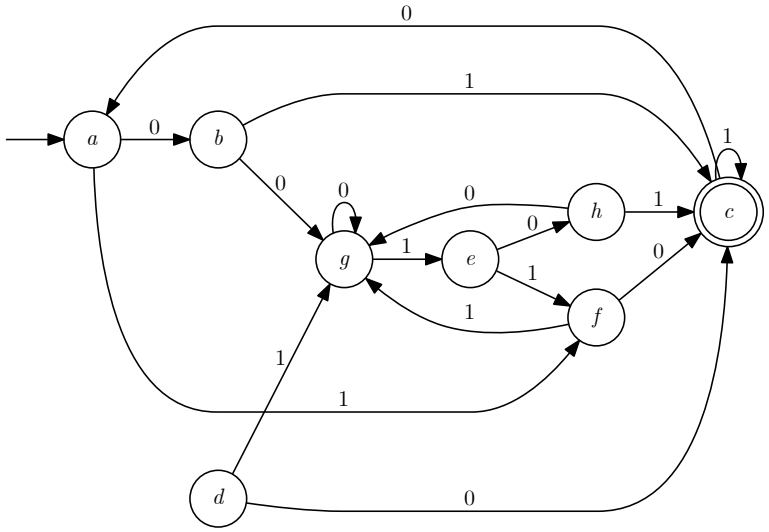
$$\delta^*(q_0, xa) \ = \ \delta(\delta^*(q_0, x), a) \ = \ \delta(p, a) \ = \ q$$

Per cui $f(q) = [xa]_{R_L}$; poiché per definizione $\bar{\delta}([x]_{R_L}, a) = [xa]_{R_L}$, la tesi è verificata. □

Definizione 2.5.1. Sia $m = (Q, \Sigma, \delta, q_0, F)$ ASFD. Definiamo la relazione $\equiv$ su Q come segue: diciamo che p è *equivalente* a q e scriviamo $p \equiv q$ se e solo se:

$$\forall x \in \Sigma^* : \quad \delta^*(p, x) \in F \quad \Leftrightarrow \quad \delta^*(q, x) \in F$$

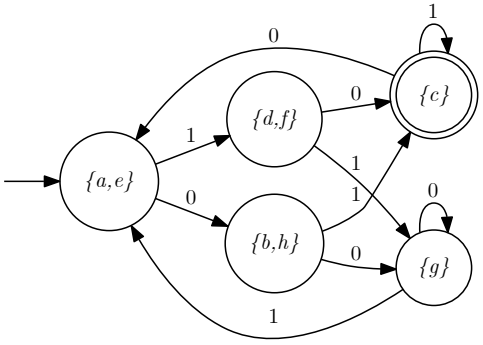
Se $p \not\equiv q$, diremo che p e q sono *distinguibili*.



(a) automa

	a	b	c	d	e	f	g	h
a	·							
b	×	·						
c	×	×	·					
d	×	×	×	·				
e		×	×	×	·			
f	×	×	×		×	·		
g	×	×	×	×	×	×	·	
h	×		×	×	×	×	×	·

(b) tabella m



(c) automa minimale equivalente a 2.3a

Figura 2.3: Alberi di derivazione

Esempio 2.5.1. Le coppie di stati distinguibili dell'automa in fig. 2.3a sono $\{a, e\}$, $\{b, h\}$, e $\{d, f\}$.

Proposizione 2.5.2.

$$p \not\equiv q \quad \Leftrightarrow \quad \exists x \in \Sigma^* : \quad \delta^*(p, x) \in F \quad \Leftrightarrow \quad \delta^*(q, x) \notin F$$

Proposizione 2.5.3. *Se $\delta(s,a) = p$ e $\delta(t,a) = q$, allora*

$$p \not\equiv q \quad \Rightarrow \quad s \not\equiv t$$

Proposizione 2.5.4. *Se $\delta^*(s,x) = p$ e $\delta^*(t,x) = q$, allora*

$$p \not\equiv q \quad \Rightarrow \quad s \not\equiv t$$

Dimostrazione.

PB: se $x = \varepsilon$, la tesi è banalmente verificata

PI: sia $x = ya$, $\delta^*(s,x) = p$, $\delta^*(t,x) = q$.

$$p \not\equiv q \quad \Rightarrow \quad \delta^*(s,x) \not\equiv \delta^*(t,x) \quad \Rightarrow$$
$$\Rightarrow \quad \delta(\delta^*(s,y),a) \not\equiv \delta(\delta^*(t,y),a) \quad \Rightarrow$$
$$\Rightarrow \quad \delta^*(s,y) \not\equiv \delta^*(t,y) \quad \Rightarrow \quad s \not\equiv t$$

□

Lemma 2.5.5. *Sia $\mathcal{A} = (Q, \Sigma, \delta, q_0, F)$ un ASFD. Allora $p \not\equiv q$ se e solo se $\{p,q\} \in m$, dove m è computato tramite la procedura seguente:*

Algorithm 1 *Algoritmo delle marcature* o *table-filling algorithm*

```
1:  procedure MARK(m, l, {p, q})
2:      m ← m ∪ {{p, q}}
3:      for {p′, q′} ∈ l[{p, q}] do
4:          MARK(m, l, {p′, q′})
5:  procedure TABLEFILLING(Q, Σ, δ, F)
6:      m ← ∅
7:      for {p, q} ⊆ Q do
8:          l[{p, q}] ← ∅
9:      for p, q ∈ F × (Q \ F) do
10:         m ← m ∪ {{p, q}}
11:     for {p, q} :  p ≠ q ∧ ({p, q} ⊆ F ∨ {p, q} ⊆ Q \ F) do
12:         if ∃a ∈ Σ : {δ(p, a), δ(q, a)} ∈ m then
13:             MARK(m, l, {p, q})
14:         else
15:             for a ∈ Σ do
16:                 if δ(p, a) ≠ δ(q, a) ∧ {δ(p, a), δ(q, a)} ≠ {p, q} then
17:                     l[{δ(p, a), δ(q, a)}] ← l[{δ(p, a), δ(q, a)}] ∪ {{p, q}}
18:     return m
```

Dimostrazione. Se p e q sono distinguibili, allora esiste una stringa x minimale tale che $\delta^*(p,x) \in F \Leftrightarrow \delta^*(q,x) \notin F$. Dimostriamo per induzione sulla lunghezza di x che $\{p,q\} \in m$.

PB: $|x| = 0, \quad x = \varepsilon$

$$\begin{aligned} \delta^*(p,\varepsilon) \in F &\quad\Leftrightarrow\quad \delta^*(q,\varepsilon) \notin F \\ p \in F &\quad\Leftrightarrow\quad q \notin F \end{aligned}$$

$\{p,q\}$ vengono inseriti in m dall'istruzione 10.

PI: $|x| = n > 0, \quad \exists a \in \Sigma, y \in \Sigma^* : x = ay$
Siano $s = \delta(p,a)$ e $t = \delta(q,a)$; s e t sono distinguibili poiché:

$$\begin{aligned} \delta^*(p,x) \in F &\quad\Leftrightarrow\quad \delta^*(q,x) \notin F \\ \delta^*(\delta(p,a),y) \in F &\quad\Leftrightarrow\quad \delta^*(\delta(q,a),y) \notin F \\ \delta^*(s,y) \in F &\quad\Leftrightarrow\quad \delta^*(t,y) \notin F \end{aligned}$$

Poiché $|y| < n$, $\{s,t\} \in m$ per ipotesi induttiva.

- se $\{s,t\}$ viene inserito in m tramite l'istruzione 10, nel momento in cui si itera il ciclo 11 per $\{p,q\}$, quest'ultima coppia viene inserita in m tramite l'istruzione 13;
- se $\{s,t\}$ viene inserito in m durante il ciclo 11 prima che si iteri il ciclo per $\{p,q\}$, la situazione è analoga al caso precedente;
- se si itera il ciclo 11 per $\{p,q\}$ prima che per $\{s,t\}$, se $\{p,q\}$ non viene inserito in m, verrà comunque aggiunto all'insieme $l[\{s,t\}]$ tramite l'istruzione 17; successivamente, nel momento in cui verrà effettuata l'istruzione $\textsc{Mark}(m,l,\{s,t\})$, anche $\{p,q\}$ sarà aggiunto a m.

□

Esempio 2.5.2. L'esecuzione dell'algoritmo di *table-filling* sull'automa in fig. 2.3a genera la tabella m in fig. 2.3b.

Teorema 2.5.6. *Sia $m = (Q,\Sigma,\delta,q_0,F)$ ASFD. L'automa minimale che accetta lo stesso linguaggio accettato da m è l'automa $m' = (Q',\Sigma,\delta',q_0',F')$, dove:*

- $Q' = \{[q]_\equiv \mid q \in Q \ \wedge\ \exists x \in \Sigma^* : \delta^*(q_0,x) = q\}$
- $\begin{aligned} \delta' : Q' \times \Sigma &\to Q' \\ ([q]_\equiv, a) &\mapsto [\delta(q,a)]_\equiv \end{aligned}$
- $q_0' = [q_0]_\equiv$

- $F' = \{[q]_{\equiv} \mid q \in F \ \wedge \ \exists x \in \Sigma^* : \delta^*(q_0, x) = q\}$

Dimostrazione. Dimostriamo innanzitutto che la definizione di δ' è consistente, cioè che se $p \equiv q$ allora $\delta(p, a) \equiv \delta(q, a)$. Ciò è verificato in quanto se per assurdo $\delta(p, a) \not\equiv \delta(q, a)$, si avrebbe $p \not\equiv q$.

Dimostriamo che $\forall q \in Q, x \in \Sigma^* : \quad \delta'^*([q]_{\equiv}), x) \ = \ [\delta^*(q, x)]_{\equiv}$

PB: $\quad \delta'^*([q]_{\equiv}, \varepsilon) \ = \ [q]_{\equiv} \ = \ [\delta^*(q, \varepsilon)]_{\equiv}$

PI: $\quad \exists y \in \Sigma^*, a \in \Sigma : \ x = ya$

$$\delta'^*([q]_{\equiv}, x) \ = \ \delta'(\delta'^*([q]_{\equiv}, y), a) \ = \ \delta'([\delta^*(q, y)]_{\equiv}, a) \ = \\ = \ [\delta(\delta^*(q, y), a)]_{\equiv} \ = \ [\delta^*(q, x)]_{\equiv}$$

Dimostriamo che $L(m) = L(m')$.

$$x \in L(m') \quad \Leftrightarrow \quad \delta'^*(q_0{}', x) \in F' \quad \Leftrightarrow \quad [\delta^*(q_0, x)]_{\equiv} \in F' \quad \Leftrightarrow \\ \Leftrightarrow \quad \delta^*(q_0, x) \in F \quad \Leftrightarrow \quad x \in L(m)$$

Infine, dimostriamo che m' è minimale. Per far ciò, è sufficiente dimostrare che m' non ha più stati di $\bar{m}$, dove $\bar{m}$ è l'automa costruibile tramite il Teorema di Myhill-Nerode.

Sia $[q]_{\equiv} \in Q'$, allora q è raggiungibile da q_0 in m, per cui $\exists x \in \Sigma^* : \delta^*(q_0, x) = q$. Sia $f : Q' \to \bar{Q}$ tale che $\forall [q]_{\equiv} \in Q' : f([q]_{\equiv}) = [x]_{R_L}$, dove $\delta^*(q_0, x) = q$. Si noti che $\forall x, y \in \Sigma^*$, se $\delta^*(q_0, x) = \delta^*(q_0, y)$, allora si ha $x \ R_m \ y$. Poiché R_m è più fine di R_L, $x \ R_L \ y$, ovvero $[x]_{R_L} = [y]_{R_L}$.

Dimostriamo che la definizione di f è consistente, ovvero che se $p \equiv q$, $\delta^*(q_0, x) = p$, e $\delta^*(q_0, y) = q$, allora $[x]_{R_L} = [y]_{R_L}$. Se $p \equiv q$, si ha $\forall z \in \Sigma^* :$

$$\delta^*(p, z) \in F \quad \Leftrightarrow \quad \delta^*(q, z) \in F \quad \Rightarrow \\ \Rightarrow \quad \delta^*(\delta^*(q_0, x), z) \in F \quad \Leftrightarrow \quad \delta^*(\delta^*(q_0, y), z) \in F \quad \Rightarrow \\ \Rightarrow \quad \delta^*(q_0, xz) \in F \quad \Leftrightarrow \quad \delta^*(q_0, yz) \in F \quad \Rightarrow \\ \Rightarrow \quad xz \in L(m) \quad \Leftrightarrow \quad yz \in L(m)$$

Per cui $x \ R_L \ y$, ovvero $[x]_{R_L} = [y]_{R_L}$.

Dimostriamo l'inettività di f, ovvero che

$$\forall [p]_{\equiv}, [q]_{\equiv} \in Q' : \quad [p]_{\equiv} \neq [q]_{\equiv} \quad \Rightarrow \quad f([p]_{\equiv}) \neq f([q]_{\equiv})$$

Siano $p, q \in Q$ tali che $[p]_{\equiv} \neq [q]_{\equiv}$, cioè $p \not\equiv q$, e siano $x, y \in \Sigma^*$ tali che $\delta^*(q_0, x) = p$ e $\delta^*(q_0, y) = q$. Supponiamo per assurdo che $[x]_{R_L} = [y]_{R_L}$, cioè che $x \ R_L \ y$. Essendo $p \not\equiv q$, $\exists w \in \Sigma^* : \delta^*(p, w) \in F \Leftrightarrow \delta^*(q, w) \notin F$.

$$\begin{aligned}
\delta^*(p,w) \in F \quad &\Leftrightarrow \quad \delta^*(q,w) \notin F \quad \Rightarrow \\
\Rightarrow \quad &\delta^*(\delta^*(q_0,x),w) \in F \quad \Leftrightarrow \quad \delta^*(\delta^*(q_0,y),w) \notin F \quad \Rightarrow \\
\Rightarrow \quad &\delta^*(q_0,xw) \in F \quad \Leftrightarrow \quad \delta^*(q_0,yw) \notin F \quad \Rightarrow \\
\Rightarrow \quad &xw \in L(m) \quad \Leftrightarrow \quad yw \notin L(m) \quad \Rightarrow \quad x \not R_L \ y
\end{aligned}$$

Il che è assurdo, per cui f è iniettiva. Poiché F è iniettiva, si ha $|Q'| \leq |\bar{Q}|$, per cui la tesi è verificata. □

Esempio 2.5.3. L'automa minimale che accetta lo stesso linguaggio accettato dall'automa in fig. 2.3a è l'automa in fig. 2.3c.

Capitolo 3

Grammatiche

Una grammatica è uno strumento per generare un linguaggio. Ha regole di produzione che permettono di generare tutte le stringhe del linguaggio partendo da insiemi finiti.

Definizione 3.0.1. Una *grammatica* è una quadrupla $G = (N, \Sigma, P, S)$ dove:

- N è l'*alfabeto* (finito) dei *simboli non terminali*;
- Σ è l'*alfabeto* (finito) dei *simboli terminali*;
- $P \subseteq ((N \cup \Sigma)^* N (N \cup \Sigma)^*) \times (N \cup \Sigma)^*$ è l'insieme (finito) delle *regole di produzione*;
- $S \in N$ è il *simbolo iniziale*.

Esempio 3.0.1.
$G = (\{S, A\}, \{0, 1\}, P, S)$
$P = \{S \to 0A1, \quad 0A \to 00A1, \quad A \to \varepsilon\}$

Nota: se $A \to \alpha_1$ e $A \to \alpha_2$, scriveremo più brevemente $A \to \alpha_1 \mid \alpha_2$. Regole del tipo $A \to \varepsilon$ sono dette *ε-produzioni*.
Dove diversamente non specificato, denotiamo con lettere greche minuscole $(\alpha, \beta, \ldots)$ elementi di $(N \cup \Sigma)^*$.

Definizione 3.0.2. Sia $G = (N, \Sigma, P, S)$. Se $\alpha, \beta, \gamma \in (N \cup \Sigma)^*$ e $\beta \to \delta \in P$ allora diciamo che $\alpha\beta\gamma$ *deriva direttamente* $\alpha\delta\gamma$ e scriviamo $\alpha\beta\gamma \underset{G}{\Longrightarrow} \alpha\delta\gamma$.

Definizione 3.0.3. Data una relazione binaria R su un insieme A, ovvero $R \subset A \times A$, la *chiusura transitiva* di R, denotata con R^+, è definita come la più piccola relazione $R' \subset A \times A$ tale che:

- $R \subseteq R'$
- $\forall a, b, c \in A : (a, b), (b, c) \in R' \Rightarrow (a, c) \in R'$ (ovvero R' gode della proprietà transitiva).

Definizione 3.0.4. Data una R relazione binaria su un insieme A, ovvero $R \subset A \times A$, la *chiusura riflessiva e transitiva* di R, denotata con R^*, è definita come la più piccola relazione $R' \subset A \times A$ tale che:

- $R \subseteq R'$
- $\forall a \in A : (a, a) \in R'$ (ovvero R' gode della proprietà riflessiva);
- $\forall a, b, c \in A : (a, b), (b, c) \in R' \Rightarrow (a, c) \in R'$ (ovvero R' gode della proprietà transitiva).

Definizione 3.0.5. Diciamo che $\alpha \in (N \cup \Sigma)^*$ *deriva non banalmente* $\beta \in (N \cup \Sigma)^*$ e scriviamo $\alpha \overset{+}{\underset{G}{\Longrightarrow}} \beta$ se:

$$\exists \alpha_0, \ldots, \alpha_n, \text{ con } n \geq 1 : \quad \alpha = \alpha_0 \underset{G}{\Longrightarrow} \alpha_1 \underset{G}{\Longrightarrow} \ldots \underset{G}{\Longrightarrow} \alpha_n = \beta$$

Definizione 3.0.6. Diciamo che $\alpha \in (N \cup \Sigma)^*$ *deriva* $\beta \in (N \cup \Sigma)^*$ e scriviamo $\alpha \overset{*}{\underset{G}{\Longrightarrow}} \beta$ se:

$$\exists \alpha_0, \ldots, \alpha_n, \text{ con } n \geq 0 : \quad \alpha = \alpha_0 \underset{G}{\Longrightarrow} \alpha_1 \underset{G}{\Longrightarrow} \ldots \underset{G}{\Longrightarrow} \alpha_n = \beta$$

Definizione 3.0.7. Sia $G = (N, \Sigma, P, S)$ una grammatica. Il *linguaggio generato* da G è il linguaggio

$$L(G) = \{w \in \Sigma^* \mid S \overset{*}{\underset{G}{\Longrightarrow}} w\}$$

Proposizione 3.0.1. *Possono esistere diverse grammatiche che generano lo stesso linguaggio.*

3.1 Gerarchia di Chomsky

Definizione 3.1.1. Sia $G = (N, \Sigma, P, S)$ una grammatica.

1. G è detta *right-linear* se contiene solo regole di produzione del tipo $A \to xB$ oppure $A \to x$, con $A, B \in N, x \in \Sigma^*$.
2. G è detta *context-free* se contiene solo regole di produzione del tipo $A \to \alpha$, con $A \in N, \alpha \in (N \cup \Sigma)^*$.

3. G è detta *context-sensitive* se contiene solo regole di produzione del tipo $\alpha \to \beta$, con $\alpha, \beta \in (N \cup \Sigma)^*, |\alpha| \leq |\beta|$.

4. G è detta *unrestricted* se non ha restrizioni sulle regole di produzione.

Nota: una regola del tipo $\alpha \to \varepsilon$, con $\alpha \in (N \cup \Sigma)^*$, è detta ε-produzione.

Teorema 3.1.1. *Gerarchia di Chomsky*

- *una grammatica* right-linear *è* context-free*;*
- *una grammatica* context-free *che non contiene* ε*-produzioni è* context-sensitive*;*
- *una grammatica* context-sensitive *è* unrestricted*.*

$$\begin{gathered}\{L(G) \mid G \text{ grammatica right-linear}\} \\ \subset \\ \{L(G) \mid G \text{ grammatica context-free}\} \\ \subset \\ \{L(G) \mid G \text{ grammatica context-sensitive}\} \\ \subset \\ \{L(G) \mid G \text{ grammatica unrestricted}\}\end{gathered}$$

Nota*: le inclusioni sono proprie, nel senso che esistono linguaggi* unrestricted *che non sono* context-sensitive*, esistono linguaggi* context-sensitive *che non sono* context-free*, ed esistono linguaggi* context-free *che non sono* right-linear*.*

Proposizione 3.1.2. *Se G è una grammatica context-sensitive, è possibile costruire una grammatica G' avente solo regole del tipo $\alpha B \beta \to \alpha \gamma \beta$, con $A \in N, \alpha, \beta, \gamma \in (N \cup \Sigma)^*$ tale che $L(G) = L(G')$.*

Teorema 3.1.3. *Esiste una corrispondenza tra tipi di riconoscitori e grammatiche.*

- *un linguaggio L è generato da una* grammatica **right-linear** *se e solo se è accettato da un* automa a **stati finiti***;*
- *un linguaggio L è generato da una* grammatica **context-free** *se e solo se è accettato da un* automa a **pila***;*
- *un linguaggio L è generato da una* grammatica **context-sensitive** *se e solo se è accettato da un* automa **linear-bounded***;*
- *un linguaggio L è generato da una* grammatica **unrestricted** *se e solo se è un linguaggio* ricorsivamente enumerabile*, ovvero, se e solo se è* accettato *da una* **macchina di Turing***.*

3.2 Grammatiche context-free

Definizione 3.2.1. Una grammatica $G = (N, \Sigma, P, S)$ è *context-free* (*cfg*) se tutte le sue regole sono del tipo $A \to \alpha$, con $A \in N, \alpha \in (N \cup \Sigma)^*$.

Definizione 3.2.2. Data una cfg $G = (N, \Sigma, P, S)$, una derivazione $S \underset{G}{\Longrightarrow} \alpha_1 \underset{G}{\Longrightarrow} \alpha_2 \underset{G}{\Longrightarrow} \dots \underset{G}{\Longrightarrow} \alpha_n$ si dice *sinistra* (o *leftmost*) se $\forall i \in \{1 \dots n-1\}$ si ha $\alpha_i = uX\beta_i$ e $\alpha_{i+1} = u\gamma\beta_i$, con $u \in \Sigma^*, X \in N, X \to \gamma \in P$.

Definizione 3.2.3. Data una cfg $G = (N, \Sigma, P, S)$ un *albero di derivazione* Γ per G è un albero tale che:

1. ogni vertice è etichettato in $N \cup \Sigma \cup \{\varepsilon\}$;
2. la radice è etichettata S;
3. ogni vertice interno è etichettato con un simbolo non terminale;
   se un vertice interno è etichettato con A e i suoi figli sono etichettati con $x_1, x_2, \dots, x_n$, dove $x_i \in N \cup \Sigma \cup \{\varepsilon\}$, allora $A \to x_1 \dots x_n$ è in P;
4. se un vertice ν ha etichetta ε, allora ν è una foglia ed è l'unico discendente del proprio genitore.

La stringa che si ottiene leggendo le etichette delle foglie dell'albero da sinistra verso destra si chiama *prodotto dell'albero.*

Teorema 3.2.1. *Sia $G = (N, \Sigma, P, S)$ cfg. Allora $S \overset{*}{\underset{G}{\Longrightarrow}} \alpha$ se e solo se esiste un albero di derivazione per G con prodotto α. È possibile stabilire una corrispondenza biunivoca fra alberi di derivazione e derivazioni sinistre (oppure derivazioni destre).*

Definizione 3.2.4. Una cfg G è detta *ambigua* se esiste una stringa x appartenente a $L(G)$ per cui esistono due diverse derivazioni sinistre (o due diversi alberi di derivazione).

Definizione 3.2.5. Un linguaggio cf si dice *inerentemente ambiguo* se ogni cfg che lo genera è ambigua.

Nota: non è decidibile se un linguaggio cf è inerentemente ambiguo o meno. Per approfondire si veda il *problema della corrispondenza di Post.*

Esempio 3.2.1.

$G_1 = (N_1, \Sigma_1, P_1, S)$

- $N_1 = \{S, A, B\}$
- $\Sigma_1 = \{a, b\}$
- $P_1 = \{$

$$S \to aAS \mid a$$
$$A \to SbA \mid SS \mid ba$$

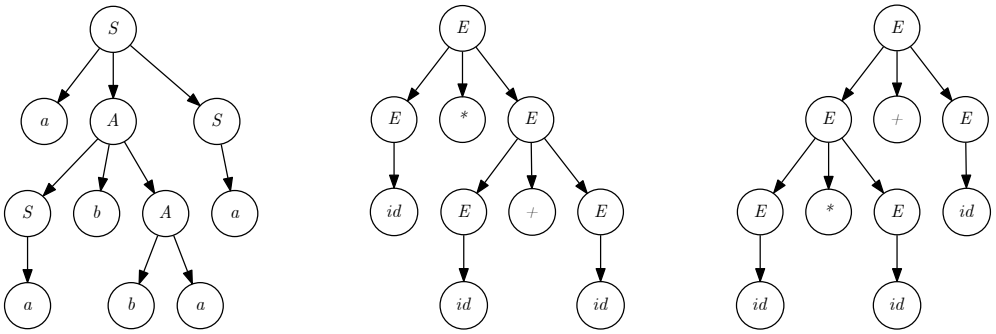
$\}$

$G_2 = (N_2, \Sigma_2, P_2, E)$

- $N_2 = \{E\}$
- $\Sigma_2 = \{id, +, *\}$
- $P_2 = \{$

$$E \to E + E \mid E * E \mid id$$

$\}$



(a) (b) (c)

Figura 3.1: Alberi di derivazione

G_1 è una grammatica non ambigua: infatti l'albero in figura 3.1a è l'unico albero per la grammatica G_1 il cui prodotto è *abaa*.

G_2 è invece una grammatica ambigua: ne è riprova la stringa $id * id + id$, che è prodotto di due alberi diversi per G_2 (in figura 3.1b e 3.1c).

Definizione 3.2.6. Una cfg $G = (N, \Sigma, P, S)$ è detta in *forma normale di Chomsky* (*CNF*) se tutte le regole di produzione sono del tipo:

$$A \to BC \text{ oppure } A \to a, \quad \text{con } A, B, C \in N, a \in \Sigma$$

Definizione 3.2.7. Una cfg $G = (N, \Sigma, P, S)$ è detta in *forma normale di Greibach* (*GNF*) se tutte le regole di produzione sono del tipo:

$$A \to a\alpha, \quad \text{con } A \in N, a \in \Sigma, \alpha \in N^*$$

Osservazione 3.2.2. *Una grammatica in forma normale di Chomsky o di Greibach non può contenere ε-produzioni.*

Definizione 3.2.8. Un simbolo $x \in N \cup \Sigma$ è detto *utile* se esiste una derivazione della forma $S \overset{*}{\underset{G}{\Longrightarrow}} \alpha x \beta \overset{*}{\underset{G}{\Longrightarrow}} w$ con $\alpha, \beta \in (N \cup \Sigma)^*, w \in \Sigma^*$.

Osservazione 3.2.3. *In una grammatica non ambigua un simbolo è utile se e solo se eliminando tutte le regole in cui compare, il linguaggio generato è lo stesso.*

Teorema 3.2.4. *Data una cfg, esiste una grammatica in CNF equivalente alla grammatica data a meno di ε-produzioni.*

Teorema 3.2.5. *Data una cfg, esiste una grammatica in GNF equivalente alla grammatica data a meno di ε-produzioni.*

Definizione 3.2.9. Data una cfg G, un simbolo $A \in N$ è detto *ricorsivo a sinistra* se $A \overset{*}{\underset{G}{\Longrightarrow}} A\alpha$.

Osservazione 3.2.6. *Una grammatica in GNF non ha simboli ricorsivi a sinistra.*

Nota: le grammatiche con simboli ricorsivi a sinistra richiedono particolare attenzione con gli analizzatori sintattici.

Lemma 3.2.7 (Lemma di Bar-Hillel)**.** *Sia $G = (N, \Sigma, P, S)$ cfg in CNF tale che $|N| = n$ e sia $L = L(G)$. Allora $\forall x \in L, |x| > 2^{n-1} : \exists r_1, q_1, r, q_2, r_2 \in \Sigma^* :$*

1. $|q_1 r q_2| \leq 2^n$
2. $q_1 q_2 \neq \varepsilon$
3. $\forall i \geq 0 : \quad r_1 \; {q_1}^i \; r \; {q_2}^i \; r_2 \in L$

Dimostrazione. Dimostriamo per induzione su i che se nel parse tree di $z \in L(G)$, con G cfg in CNF, ogni cammino ha lunghezza $\leq i$, allora $|z| \leq 2^{i-1}$.

PB: Se $i = 1$, l'albero di derivazione deve essere necessariamente del tipo $S \to a$, con $a \in \Sigma$. Per cui necessariamente $z = a$, $|z| = 1$.

PI: Supponiamo che la proprietà sia verificata per i. Se l'albero di radice S ha altezza $i + 1$, i due alberi di radice rispettivamente A e B, sottoalberi della radice, hanno altezza $\leq i$. Siano z_1 e z_2 i prodotti di tali alberi: si ha $|z_1| \leq 2^{i-1}, |z_2| \leq 2^{i-1} \Leftrightarrow |z| = |z_1 z_2| \leq 2^i$.

Nota: il lemma appena dimostrato si applica in generale ad alberi aventi come radice un simbolo non terminale e come prodotto una stringa di simboli terminali.

Nel parse tree di x vi è un cammino di lunghezza $> n$: se per assurdo ogni cammino avesse lunghezza $\leq n$, si avrebbe $|x| \leq 2^{n-1}$.
Sia P un cammino di lunghezza massima nel parse tree di x; per quanto dimostrato prima, $|P| > n$, cioè $|P| \geq n+1$. Quindi P contiene un numero di vertici $\geq n+2$ e un numero di vertici interni $\geq n+1$. Per cui in P esistono due vertici v_1 e v_2 tali che:

1. v_1 e v_2 hanno la stessa etichetta $X \in N$;
2. v_1 è più vicino alla radice rispetto a v_2;
3. la porzione di P che va da v_1 alla foglia ha lunghezza $\leq n+1$.

Sia T_1 il sottoalbero avente radice v_1 e sia T_2 il sottoalbero avente radice v_2. Denotiamo con z il prodotto di T_1 e con r il prodotto di T_2. Allora:

$$\exists q_1, q_2 : z = q_1 r q_2$$

$$\exists r_1, r_2 : x = r_1 z r_2$$

1. $|q_1 r q_2| = |z|$; z è il prodotto di un albero i cui cammini hanno lunghezza $\leq n+1$, per cui $|z| \leq 2^n$.
2. $q_1 q_2 \neq \varepsilon$, poiché altrimenti il nodo v_1 dovrebbe avere come unico figlio il nodo v_2, il che è assurdo in quanto G è in CNF.
3. $S \overset{*}{\underset{G}{\Longrightarrow}} r_1 X r_2, \quad X \overset{*}{\underset{G}{\Longrightarrow}} q_1 X q_2, \quad X \overset{*}{\underset{G}{\Longrightarrow}} r$
   Dimostriamo per induzione che $\forall i \geq 0 : \quad S \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^i X {q_2}^i r_2$.

   PB: $S \overset{*}{\underset{G}{\Longrightarrow}} r_1 X r_2$

   PI: Per ipotesi induttiva si ha $S \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^i X {q_2}^i r_2$.
   $S \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^i X {q_2}^i r_2 \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^{i+1} X {q_2}^{i+1} r_2$

   Per cui $\quad \forall i \geq 0 : \quad S \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^i X {q_2}^i r_2 \overset{*}{\underset{G}{\Longrightarrow}} r_1 {q_1}^i r {q_2}^i r_2$.

□

Esempio 3.2.2.
$L = \{a^m b^m c^m \mid m \geq 0\}$
Supponiamo esista una cfg $G = (N, \Sigma, P, S)$ in CNF tale che $L = L(G)$; poniamo $k > 2^{|N|-1}/3$. Sia $x = a^k b^k c^k$. Chiaramente $x \in L, |x| = 3k > 2^{|N|-1}$. Supponiamo esistano $r_1, q_1, r, q_2, r_2 \in \Sigma^*$ tali che $x = r_1 q_1 r q_2 r_2$:

- se $\{q_1, q_2\} \cap \langle \underline{a}^* \cup \underline{b}^* \cup \underline{c}^* \rangle = \varnothing$, allora si ha
  $\forall i > 1 : \quad r_1 {q_1}^i r {q_2}^i r_2 \notin \{a^l b^m c^n \mid l, m, n \geq 0\} \quad \Rightarrow \quad r_1 {q_1}^i r {q_2}^i r_2 \notin L$

- se $q_1, q_2 \in \langle \underline{a}^* \cup \underline{b}^* \cup \underline{c}^* \rangle$, allora si ha che $\forall i > 1$:

  - ci sarà in x almeno una lettera presente $> k$ volte;
  - ci sarà in x almeno una lettera presente k volte.

  Per cui $\forall i > 1 : x \notin L$.

Poiché la tesi del Lemma di Bar-Hillel è negata, possiamo dedurre che non esiste cfg G tale che $L = L(G)$.

Esempio 3.2.3.
$L = \{a_m b_m \mid m \geq 0\}$
$G = (\{S\}, \{a, b\}, P, S)$
$P = \{S \to aSb \mid \varepsilon\}$
$L = L(G)$

Esempio 3.2.4.
$L = \{a_k b_m c_m \mid k, m \geq 0\}$
$G = (\{S, A, B\}, \{a, b, c\}, P, S)$
$P = \{S \to AB, \quad A \to Aa \mid \varepsilon, \quad B \to bBc \mid \varepsilon\}$
$L = L(G)$

Proposizione 3.2.8. *Siano L_1 e L_2 linguaggi cf. Allora $L_1 \cup L_2$ è cf.*

Dimostrazione. Siano $G_1 = (N_1, \Sigma_1, P_1, S_1)$ e $G_2 = (N_2, \Sigma_2, P_2, S_2)$ cfg tali che $L_1 = L(G_1)$, $L_2 = L(G_2)$. Possiamo supporre senza perdità di generalità che $N_1 \cap N_2 = \varnothing$. Sia $G = (N_1 \cup N_2 \cup \{S\}, \Sigma_1 \cup \Sigma_2, P_1 \cup P_2 \cup \{S \to S_1 \mid S_2\}, S)$ cfg. Dimostriamo che $L(G) = L(G_1) \cup L(G_2)$.

- $L(G_1) \cup L(G_2) \subseteq L(G)$
  Sia $x \in L(G_1) \cup L(G_2)$. Se $x \in L(G_1)$, $S_1 \overset{*}{\underset{G_1}{\Longrightarrow}} x$. Allora $S \underset{G}{\Longrightarrow} S_1 \overset{*}{\underset{G}{\Longrightarrow}} x$.
  Analoga dimostrazione se $x \in L(G_2)$.

- $L(G) \subseteq L(G_1) \cup L(G_2)$
  Sia $x \in L(G)$. Allora

$$S \overset{*}{\underset{G}{\Longrightarrow}} x \quad \Rightarrow \quad S \underset{G}{\Longrightarrow} S_1 \overset{*}{\underset{G}{\Longrightarrow}} x \quad \vee \quad S \underset{G}{\Longrightarrow} S_2 \overset{*}{\underset{G}{\Longrightarrow}} x \quad \Rightarrow$$
$$\Rightarrow \quad S_1 \overset{*}{\underset{G_1}{\Longrightarrow}} x \quad \vee \quad S_2 \overset{*}{\underset{G_2}{\Longrightarrow}} \quad \Rightarrow \quad x \in L(G_1) \vee x \in L(G_2)$$

□

Proposizione 3.2.9. *Siano L_1 e L_2 linguaggi cf. Allora $L_1 \circ L_2$ è cf.*

Osservazione 3.2.10. *Siano L_1 e L_2 linguaggi cf. Non è detto che $\Sigma \setminus L_1$, $\Sigma \setminus L_1$, e $L_1 \cap L_2$ siano cf.*

Proposizione 3.2.11. *L'intersezione di un linguaggio cf e di un linguaggio regolare è un linguaggio cf.*

Capitolo 4

Automi a pila

Definizione 4.0.1. Un *automa a pila non deterministico* è un sistema $m = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ dove:

- Q è l'*insieme* (finito) *degli stati*;
- Σ è l'*alfabeto* (finito) di *input*;
- Γ è l'*alfabeto* (finito) di *stack*;
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma^*)$ è la *funzione* (totale) *di transizione*;
- $q_0 \in Q$ è lo *stato iniziale*;
- $Z_0 \in \Gamma$ è il *simbolo di stack iniziale*;
- $F \subseteq Q$ è l'*insieme degli stati finali.*

Definizione 4.0.2. Una *configurazione* o *descrizione istantanea* (*ID*) è una tripla (q, w, γ) con $q \in Q, w \in \Sigma^*, \gamma \in \Gamma^*$.

Definizione 4.0.3. Definiamo la relazione $\vdash_m$ su $Q \times \Sigma^* \times \Gamma^*$ (ovvero l'insieme delle configurazioni istantanee) come segue: diciamo che $(q, aw, Z\alpha) \vdash_m (p, w, \beta\alpha)$ se e solo se:

- $p, q \in Q, \quad a \in \Sigma \cup \{\varepsilon\}, \quad w \in \Sigma^*, \quad Z \in \Gamma, \quad \alpha, \beta \in \Gamma^*$
- $(p, \beta) \in \delta(q, a, Z)$

Definizione 4.0.4. Sia $m = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ un automa a pila non deterministico. Il *linguaggio accettato* (o *riconosciuto*) *per stati finali* da m è il linguaggio

$$L(m) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \quad \vdash_m^* \quad (q, \varepsilon, \alpha), q \in F, \alpha \in \Gamma^*\}$$

Definizione 4.0.5. Sia $m = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ un automa a pila non deterministico. Il *linguaggio accettato* (o *riconosciuto*) *per stack vuoto* da m è il linguaggio

$$N(m) = \{w \in \Sigma^* \mid (q_0, w, Z_0) \ \vdash_m^* \ (q, \varepsilon, \varepsilon), q \in Q\}$$

Esempio 4.0.1. $L = \{wcw^R \mid w \in \{0,1\}^*, c \notin \{0,1\}\}$

- $Q = \{q_0, q_1\}$
- $\Sigma = \{0, 1, c\}$
- $\Gamma = \{Z_0, A, B\}$
- $$\delta(q_0, a, Z) = \begin{cases} \{(q_0, AZ)\} & \text{se } a = 0 \\ \{(q_0, BZ)\} & \text{se } a = 1 \\ \{(q_1, Z)\} & \text{se } a = c \end{cases}$$
  $$\delta(q_1, a, Z) = \begin{cases} \{(q_1, \varepsilon)\} & \text{se } \begin{array}{r}(a = 0 \wedge Z = A) \vee \\ (a = 1 \wedge Z = B) \vee \\ (a = \varepsilon \wedge Z = Z_0)\end{array} \\ \varnothing & \text{altrimenti} \end{cases}$$

$N(m) = L$

Teorema 4.0.1.

a) $L = N(m_1) \quad \Rightarrow \quad \exists m_2 : L = L(m_2)$

b) $L = L(m_2) \quad \Rightarrow \quad \exists m_1 : L = N(m_1)$

Dove m_1 e m_2 sono automi a pila non deterministici.

Dimostrazione.

a) Sia $L = N(m_1)$ per $m_1 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, \varnothing)$.
Poniamo $m_2 = (Q \cup \{q_0', q_f\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0, \{q_f\})$, dove δ' è così definita:

- $\delta'(q_0', \varepsilon, X_0) := \{(q_0, Z_0 X_0)\}$
- $\forall q \in Q, a \in \Sigma, Z \in \Gamma : \quad \delta'(q, a, Z) := \delta(q, a, Z)$
- $\forall q \in Q : \quad \delta'(q, \varepsilon, X_0) := \{(q_f, \varepsilon)\}$

Si dimostra che $N(m_1) = L(m_2)$.

b) Sia $L = L(m_2)$ per $m_2 = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$.
Poniamo $m_1 = (Q \cup \{q_0', q_e\}, \Sigma, \Gamma \cup \{X_0\}, \delta', q_0', X_0, \varnothing)$, dove δ' è così definita:

- $\delta'({q_0}', \varepsilon, X_0) := \{(q_0, Z_0X_0)\}$
- $\forall q \in Q, a \in \Sigma, Z \in \Gamma : \quad \delta(q,a,Z) = R \quad \Rightarrow \quad \delta'(q,a,Z) \supseteq R$
- $\forall q \in F, Z \in \Gamma \cup \{X_0\} : \quad \delta'(q,\varepsilon,Z) \ni (q_e, \varepsilon)$
- $\forall Z \in \Gamma \cup \{X_0\} : \quad \delta'(q_e,\varepsilon,Z) = \{(q_e,\varepsilon)\}$

i) Dimostriamo che $L(m_2) \subseteq N(m_1)$.
Sia $x \in L(m_2)$. Si ha $(q_0, x, Z_0) \overset{*}{\underset{m_2}{\vdash}} (q, \varepsilon, \alpha)$, con $q \in F$.

$$({q_0}', x, X_0) \quad \underset{m_1}{\vdash} \quad (q_0, x, Z_0X_0) \quad \overset{*}{\underset{m_1}{\vdash}} \quad (q, \varepsilon, \alpha X_0), \text{ con } q \in F \quad \underset{m_1}{\vdash}$$
$$\underset{m_1}{\vdash} \quad (q_e, \varepsilon, \alpha X_0) \quad \overset{+}{\underset{m_1}{\vdash}} \quad (q_e, \varepsilon, \varepsilon) \quad \Rightarrow \quad x \in N(m_1)$$

ii) Dimostriamo che $L(m_2) \supseteq N(m_1)$.
Sia $x \in N(m_1)$. Si ha $({q_0}', x, X_0) \overset{*}{\underset{m_1}{\vdash}} (q_e, \varepsilon, \varepsilon)$.

$$({q_0}', x, X_0) \quad \underset{m_1}{\vdash} \quad (q_0, x, Z_0X_0) \quad \overset{*}{\underset{m_1}{\vdash}} \quad (q, \varepsilon, \alpha X_0), \text{ con } q \in F \quad \underset{m_1}{\vdash}$$
$$\underset{m_1}{\vdash} \quad (q_e, \varepsilon, \alpha X_0) \quad \overset{+}{\underset{m_1}{\vdash}} \quad (q_e, \varepsilon, \varepsilon) \quad \Rightarrow$$
$$\Rightarrow \quad (q_0, x, Z_0) \quad \overset{*}{\underset{m_2}{\vdash}} \quad (q, \varepsilon, \alpha), \text{ con } q \in F \quad \Rightarrow \quad x \in L(m_2)$$

□

Definizione 4.0.6. Un automa a pila $m = (Q, \Sigma, \Gamma, \delta, q_0, Z_0, F)$ è *deterministico* se:

1. $\forall q \in Q, a \in \Sigma \cup \{\varepsilon\}, Z \in \Gamma : \quad |\delta(q,a,Z)| \leq 1$
2. $\forall q \in Q, a \in \Sigma, Z \in \Gamma : \quad \delta(q,a,Z) \neq \varnothing \quad \Rightarrow \quad \delta(q,\varepsilon,Z) = \varnothing$

Esempio 4.0.2. Il linguaggio $L = \{ww^R \mid w \in \{0,1\}^*\}$ (ovvero il linguaggio delle stringhe palindrome sull'alfabeto $\{0,1\}$) non è riconoscibile da alcun automa a pila deterministico.

Teorema 4.0.2. *Per ogni linguaggio L, esiste un automa a pila m tale che $L = L(m)$ se e solo se esiste una cfg G tale che $L = L(G)$.*

Idea di dimostrazione. Senza perdita di generalità supponiamo che G sia in GNF. Facciamo corrispondere a regole del tipo $A \to a\gamma$ transizioni del tipo $\delta(q,a,A) \ni (q,\gamma)$ e viceversa. □

Capitolo 5

Macchine di Turing

Definizione 5.0.1. Un *macchina di Turing deterministica* (*TM*) è un sistema $m = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$, dove:

- Q è l'*insieme* (finito) *degli stati*;
- $\Sigma \subseteq \Gamma$ è l'*alfabeto* (finito) *di input*;
- Γ è l'*alfabeto* (finito) *di nastro*;
- $\delta : Q \times \Gamma \to Q \times \Gamma \times \{L, R\}$ è la *funzione di transizione*;
- $q_0 \in Q$ è lo *stato iniziale*;
- $B \in \Gamma, B \notin \Sigma$ è il *carattere blank*;
- $F \subseteq Q$ è l'*insieme degli stati finali.*

Teorema 5.0.1. *Per ogni L, se esiste una TM non deterministica m tale che $L = L(m)$, allora esiste una TM deterministica m' tale che $L = L(m')$.*

Idea di dimostrazione. Sia $L = L(m)$ dove $m = (Q, \Sigma, \Gamma, \delta, q_0, B, F)$ è una TM non deterministica. Sia $r = \max\{|\delta(q, a)| \mid q \in Q, a \in \Gamma\}$. Ogni elemento di $\delta(q, a)$ può essere numerato e inoltre ogni ramo della computazione può essere rappresentato tramite una sequenza di numeri $\in \{1, \ldots, r\}$.
Possiamo costruire una TM deterministica m' che faccia la ricerca in ampiezza di un ramo della computazione che termini con uno stato finale. Tale macchina ha tre nastri:

1. un nastro di input;
2. un nastro di lavoro, contenente la sequenza che identifica il ramo che sta attualmente esplorando;

3. un nastro di lavoro, in cui la macchina emula il ramo attuale della computazione.

□