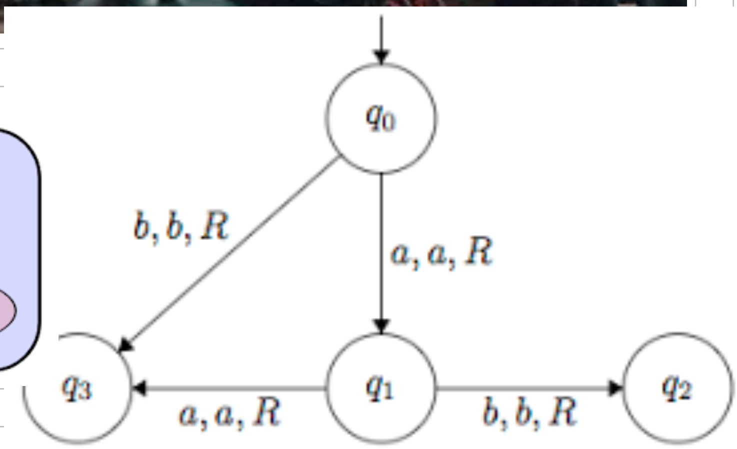
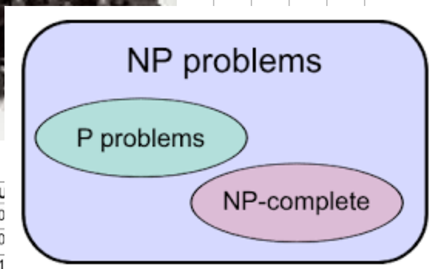
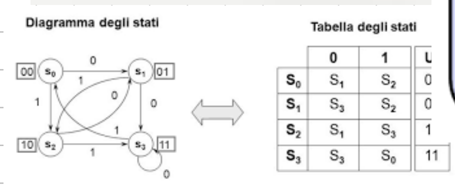
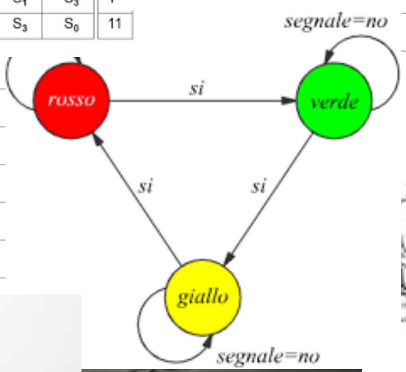
Per tracce esercizi visita il seguente link:

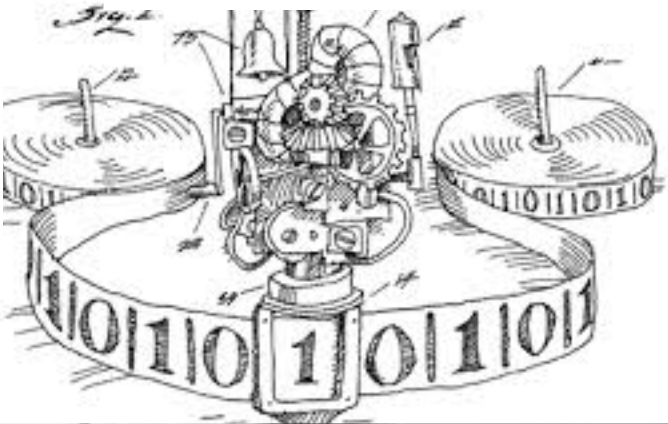
https://github.com/Exyss/university-notes/blob/main/Bachelor/Terzo%20Anno/Automi%2C%20Calcolabilita%20e%20Complessita.pdf

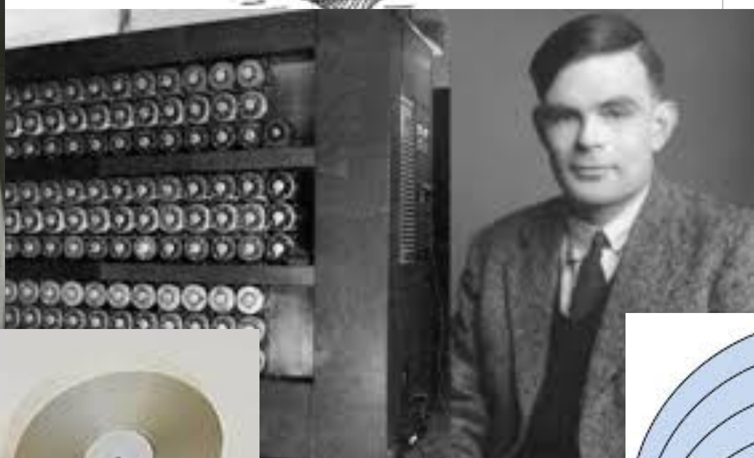


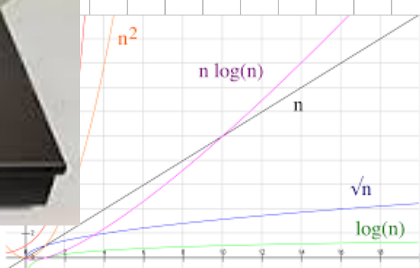


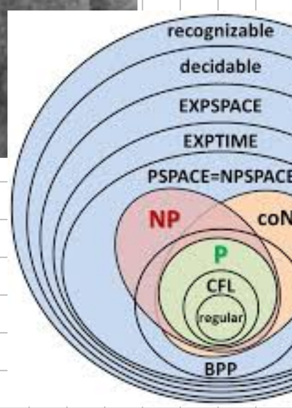










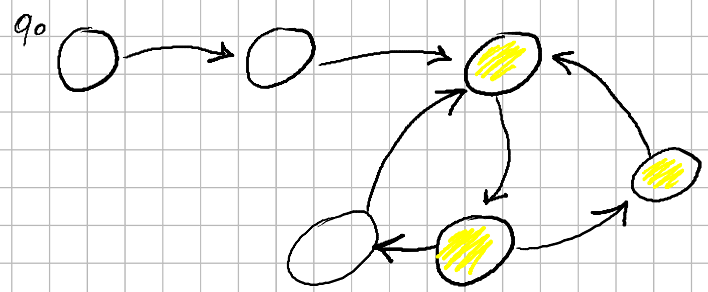


automi, complessità, calcolabilità — ESERCIZI

LINGUAGGI REGOLARI

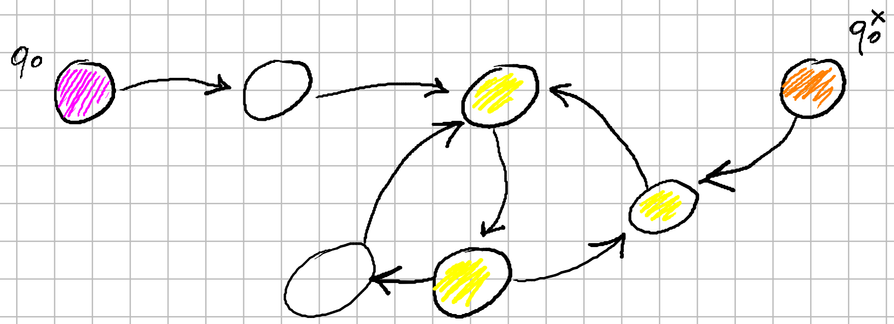
ESERCIZIO 1.1.

PARTIAMO DA UN AUTOMA D DFA $(Q, \Sigma, \delta, q_0, F)$



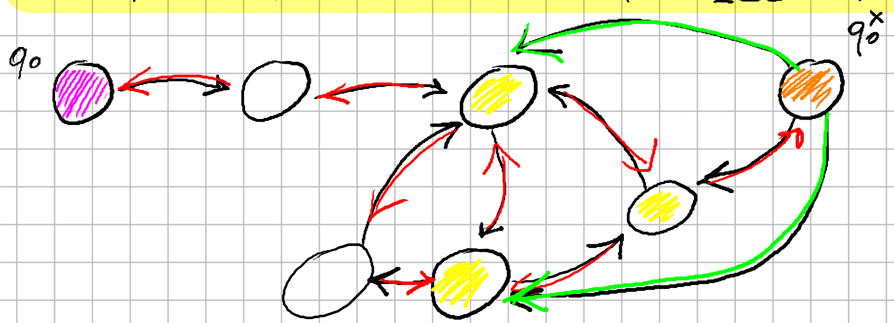
(i nodi in giallo sono i nodi appartenenti a F)

CREIAMO ORA UN NUOVO AUTOMA D_2 NFA



(il nodo arancio è un nuovo stato, $F_2 = \{q_0\}$)

INVERTO IL VERSO DI TUTTE LE FRECCIE



IN SIMBOLI:

$$D_2 = NFA\left(Q_2, \Sigma, \delta_2, q_0^x, F_2 = \{q_0\}\right)$$

$$Q_2 = Q$$

$$\begin{cases} \forall\, q_1, q_2 \in Q \quad \delta_2(q_1, a) = q_2 \iff \delta(q_2, a) = q_1 \\ \text{se } a = \varepsilon,\ q_1 \in F \rightarrow \delta_2(q_0^x\ \varepsilon) = q_1 \end{cases}$$

$$F_2 = \{q_0\}$$

POSSIAMO DIRE CHE $w \in D \iff w \in D_2$ $\square$

ESERCIZIO 1.2

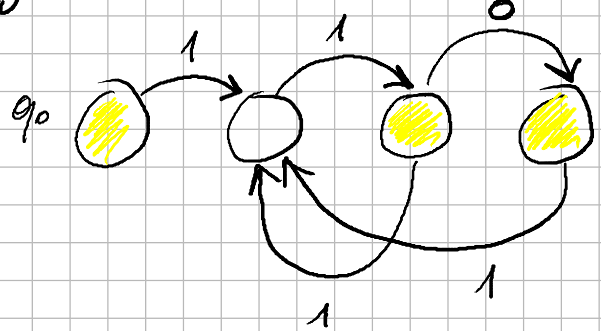
$L = \{11, 110\}^*$

L^*

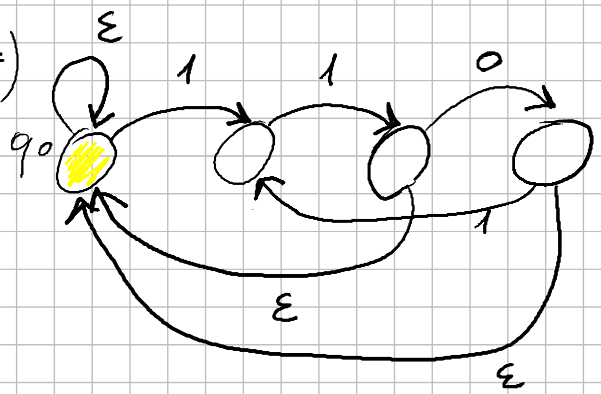
QUESTO E` UN LINGUAGGIO POSTO SOTTO OPERATORE STAR DI KLEENE

$L^* = \{11, 110\}^* = \varepsilon, 11, 110, 11110, 11011, 111110 \ldots$

DFA DIRETTO



NFA$(Q, \Sigma, \delta, q_0, F)$



DFA$(Q_2, \Sigma_2, \delta_2, q_0^*, F_2)$

IN SIMBOLI:

$Q_2 = Q$

$$\delta_2(R, a) = \left(\bigcup_{r \in R} \delta(E(r, a)) \right)$$

$q_0^* = E q_0$

$F_2 = R \mid (R \cap F) \neq 0$

ESERCIZIO 1.3.

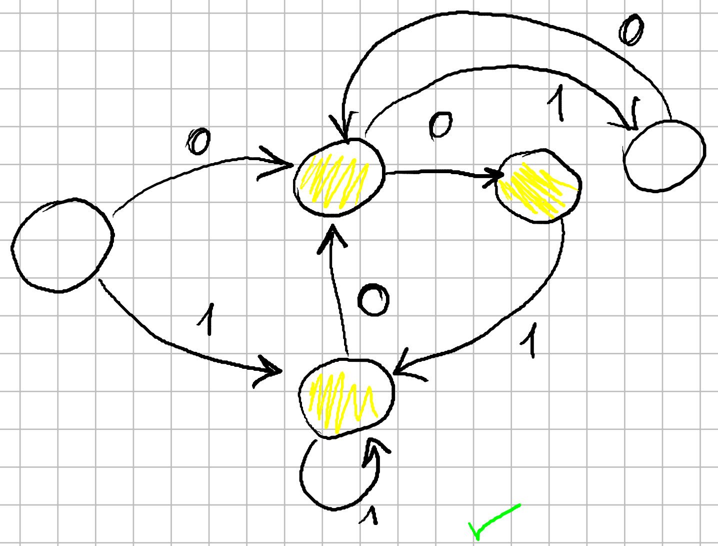
$R = (01^+)^*$ costruisci D t.c.

$L(D) = (w \in \{0,1\}^* \wedge w \notin L(R))$

$R = \varepsilon, 01, 0101, 010101, \dots$ ALLORA

$R = (01)^*$

$L(D) = \cancel{\varepsilon}, 0, 1, 010, 01010 \dots, 101, 111, 1011, \dots$ NO



ESERCIZIO 1.4.

$L = \{ w \in \{0,1\}^* \mid |w|_0 = |w|_1 \}$

DIMOSTRARE CHE $L \notin REG$

DIMOSTRO TRAMITE IL PUMPING LEMMA

- Sia p la lunghezza del pumping

consideriamo la stringa

$w = 0^p 1^p$

$\left.\begin{array}{l} |xy| \leq p \\ |y| \geq 0 \end{array}\right\}$ $w = 0^m 0^{p-m} 1^p = \underbrace{0^{(m)}}_{p-k} \underbrace{0^{(p-m)}}_{k = p-m} 1^p = 0^{p-k} (0^k)^J 1^p$

PROVIAMO PER $J = 0$

$0^{p-k} 1^p$! queste stringe non e' in $L \rightarrow L \notin REG$

ESERCIZIO 1.5.

$L = \{ 1^{n^2} \mid n \in \mathbb{N} \} \in REG$?? ? ? ?

USIAMO IL PUNPING LEMMA

Supponiamo che:

$$\left.\begin{array}{l} |xy| \leq p \\ |y| \geq 0 \\ 1^{n^2} = \underbrace{1^n \cdot 1^n \cdot 1^n \cdot \ldots \cdot 1^n}_{n \text{ VOLTE}} \end{array}\right\} \quad w = \underbrace{1^p}_{(p-k \text{ VOLTE})} \cdot \underbrace{1^p}_{(k \text{ VOLTE})} \cdot \underbrace{1^p}_{(p \text{ VOLTE})} = \overbrace{1^p}^{p-m \text{ VOLTE}} \cdot \left(\overbrace{1^p}^{m \text{ VOLTE}}\right)^i \cdot \overbrace{1^p}^{p \text{ VOLTE}}$$

PROVIAMO CON $i = 0$

OTTENGO:

$$\underbrace{1^p \cdot 1^p \cdot 1^p \cdot 1^p \cdot \ldots \cdot 1^p}_{p-m \text{ VOLTE}} \cdot \underbrace{1^p \cdot \ldots \cdot 1^p}_{p \text{ VOLTE}} \notin L \rightarrow L \notin \text{REG}$$

1 Automi

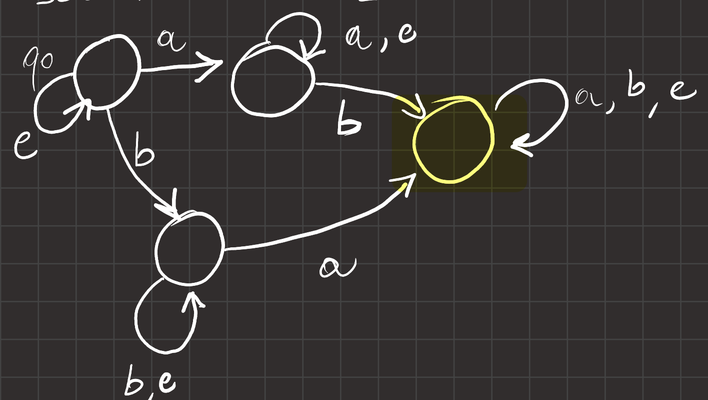
10 Points

- Si scriva un'espressione regolare per il linguaggio costituito dall'insieme di stringhe su alfabeto $\{a, b, c\}$ contenenti almeno una a ed almeno una b. Si determini un DFA che riconosca lo stesso linguaggio.
- Definire la forma normale di Chomsky per grammatiche acontestuali. Dimostrare che ogni linguaggio acontestuale è generato da una grammatica acontestuale in forma normale di Chomsky.

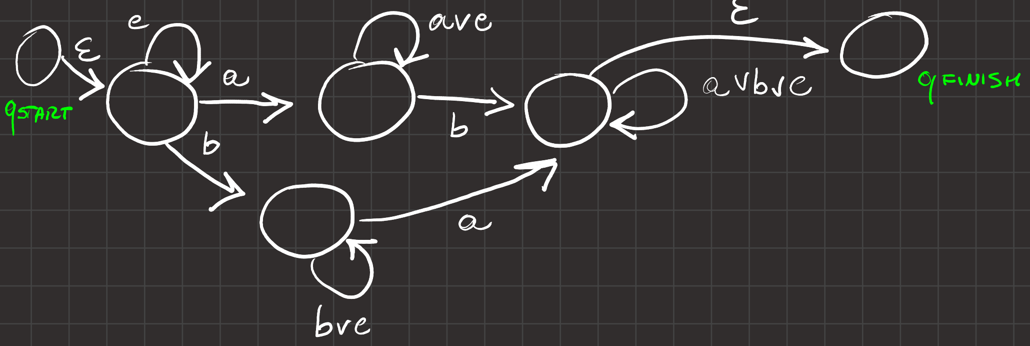
(1) DATO CHE $\mathcal{L}(DFA) = \mathcal{L}(RE)$

DISEGNAMO PRIMA L'AUTOMA

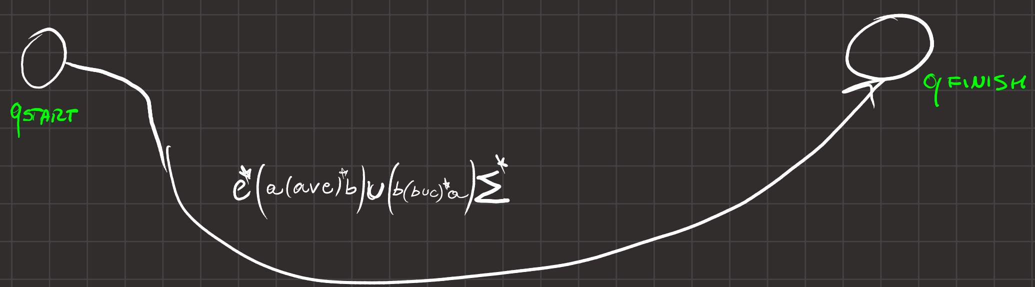
STRINGHE ACCETTATE.
aabc
abc
bca
acb
ab
aaba ...



LO CONVERTIAMO IN UN GNFA EQUIVALENTE



RIDUCIAMO



ESPRESSIONE: $c^*(a(a\cup c)^*b)\cup(b(b\cup c)^*a)\Sigma^*$

(2) Una grammatica in forma normale di Chomsky e' una CFG G tale che: tutte le regole in R di G hanno le forme seguente:

$S \to \varepsilon$, $A \to a$, $A \to BC$

dove S = STATO INIZIALE

$A \in V \wedge A \notin \Sigma$

$B, C \in V \cup \Sigma$

$a \in \Sigma$

- VERO PERCHÈ OGNI LINGUAGGIO ACONTESTUALE È GENERATO DA UNA CFG E PER OGNI CFG G ESISTE G' CFG IN CNF.

È POSSIBILE MODIFICARE G PER RENDERLO IN CNF MEDIANTE IL SEGUENTE ALGORITMO.

1. CREO UN NUOVO STATO INIZIALE S_0, CREO LA REGOLA $S_0 \to S$
2. SE HO REGOLE TIPO $A \to \varepsilon$ (dette unitarie) LE ELIMINO,

   POI PER OGNI REGOLA CON A A DESTRA AGGIUNGO REGOLE

   CON TUTTE LE COMBINAZIONI. ESEMPIO: HO $B \to \varepsilon A \varepsilon A \varepsilon$

   AGGIUNGO $B \to \varepsilon \varepsilon A \varepsilon$

   $B \to \varepsilon \varepsilon \varepsilon$

   ...

3. SE HO REGOLE TIPO $A \to B$ DOVE $A, B \in V \cup \Sigma$, SE HO REGOLE TIPO $B \to C$ dove $C \in \Sigma \cup V$, RIMUOVO $B \to C$, AGGIUNGO $A \to C$
4. SE HO REGOLE TIPO $A \to u_1 u_2 \dots u_k$ DOVE $\forall u_i$ $u_i \in \Sigma$

   CREO $A_1, \dots, A_{k-1}$ DOVE $\forall A_i$ $A_i \in V \cup \Sigma$

   AGGIUNGO LE REGOLE

   $A \to A_1 u_1$

   $A_1 \to A_2 u_2$

   ...

   $A_{k-2} \to A_{k-1} u_k$

   RIMUOVO LA REGOLA $A \to u_1 u_2 \dots u_k$ INIZIALE
5. PER OGNI REGOLA $A \to u_1 u_2$ dove $u_1, u_2 \in V \cup \Sigma$

   SCRIVO LA REGOLA $U_1 \to u_1$ SE $u_1 \in \Sigma$ dove $U_1 \in V$

   SOSTITUISCO LA PRIMA REGOLA CON $A \to U_1 u_2$