- Decidibilità dei linguaggi (tutto il discorso sul rapporto biunivoco tra numeri naturali e stringhe di alfabeto A)

[Decidibilità linguaggi regolari]
Dato un DFA M con n stati:

- L(M)=ø (decidibile)
- L(M)= infinito (decidibile)

[Decidibilità grammatiche libere]

- Decidibile se linguaggio generato da un GL è finito
- Decidibile se linguaggio generato da un GL è vuoto
- Decidibile se una parola appartiene ad un linguaggio generato da una GL

SE:

- L_1 Ω L_2 = Ø
- L_1 Ω L_2 = INFINITO
- L_1 Ω L_2 = linguaggio libero
- L_1 Ω L_2 = linguaggio regolare
- L_1 (sottoinsieme) L_2
- L_1 = L_2 (problema equivalenza)

→ Indecibidili

- Definizione di "punto fisso" di una stringa. Applicazione del punto fisso al funzionale (però negli esercizi si parla sempre di "punto fisso minimo", che noi NON abbiamo fatto).

- Definizione di grammatica

Una grammatica è, intuitivamente, un insieme di regole che permettono di generare un linguaggio. Un ruolo fondamentale tra le grammatiche è costituito dalle grammatiche libere dal contesto mediante le quali vengono solitamente descritti i linguaggi di programmazione.

-definizione di grammatica regolare.

Formato da una quadrupla Gr={NT, T, S, P}

Dove: NT(insieme finito di variabili→simboli non terminali), T(insieme finito di termini primitivi→simboli terminali), S(simbolo iniziale), P(insieme finito di produzioni→dove a sinistra ci deve essere almeno un non terminale ed a destra qualsiasi cosa.)

ES:

Gr= {(S,C), (a,b,c), S, P}

S ::= aS | bC | b
C ::= cC | c

-definizione di grammatica libera dal contesto

Formato da una quadrupla Gr={NT, T, S, P}
Dove: NT(insieme finito di variabili→simboli non terminali), T(insieme finito di termini primitivi→simboli terminali), S(simbolo iniziale), P(insieme finito di produzioni→dove a sinistra ci deve essere almeno un non terminale ed a destra qualsiasi cosa.)

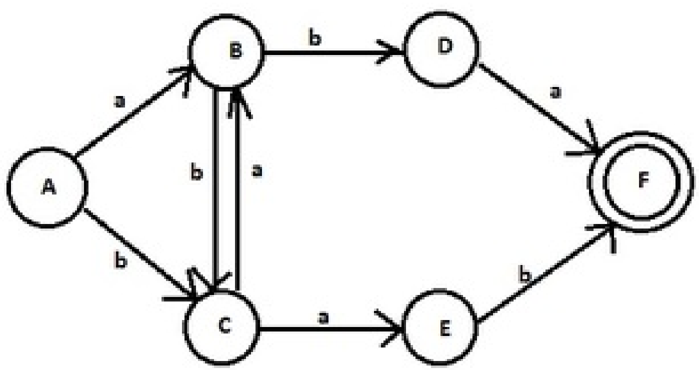
ES:

Gr= {(S,A), (a,b), S, P}
Genera il linguaggio L={ $a^n b^n$ | n>=0}

S ::= aA | ∂
C ::= Sb

Sarà grammatica libera dal contesto; ha *entrambe* le produzioni, destre e sinistre, e quindi non è regolare.

- Passaggio: da una grammatica libera da contesto al relativo automa non deterministico. Dall'automa non deterministico alla relativa grammatica libera dal contesto.



STATI	a	b
A	B	C
B	/	CD
C	BE	/
D	F	/
E	/	F
CD	BEF	/
BE	/	CDF
BEF	/	CDF
CDF	BEF	/

Corrispondente Grammatica:

G={ (A, ,B, C , D, E,F) , (a,b), A , P}

A::= aB | bC
B::= vuoto | bCD
C::= aBE | vuoto
D::= aF | vuoto
E::= vuoto | bF
CD::= aBEF | vuoto
BE::= vuoto | bCDF
BEF::= vuoto | bCDF
CDF::= aBEF | vuoto
F::= vuoto | vuoto

- Rapporto tra linguaggi regolari, grammatiche regolari, espressioni regolari, automi a stati finiti.

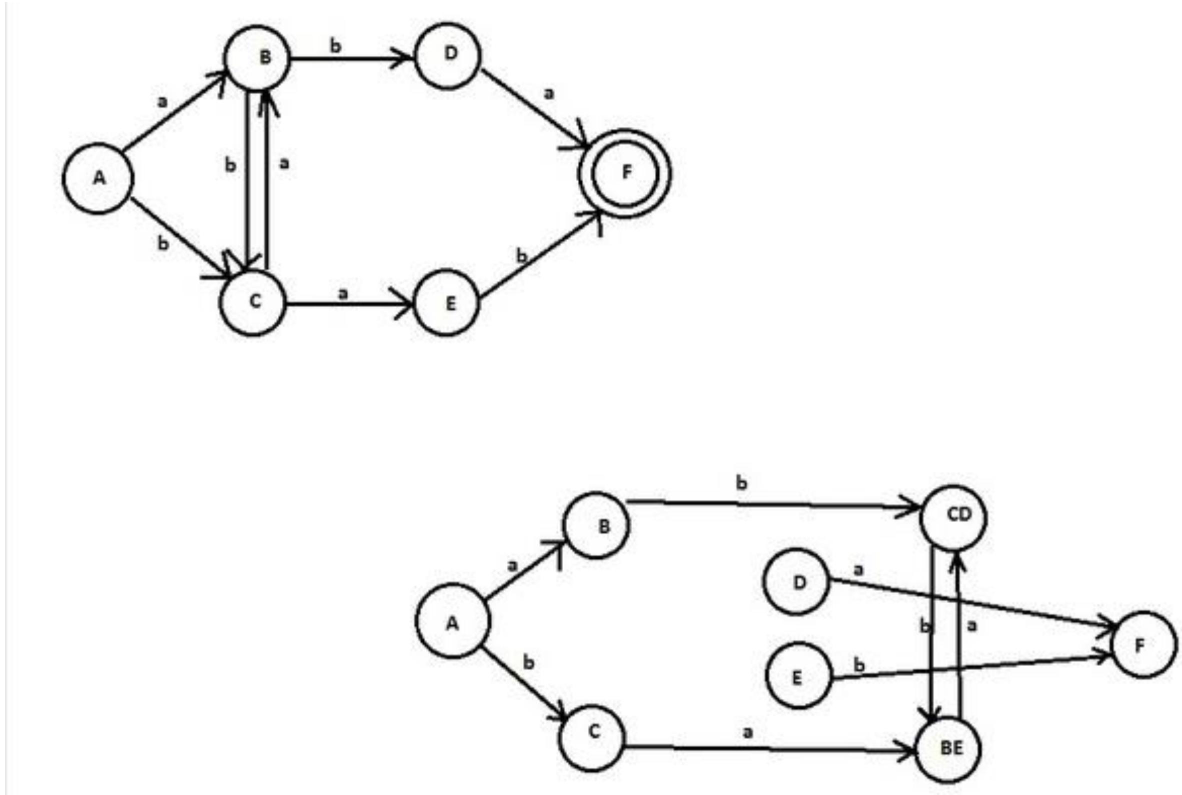
0	Grammatica ricorsivamente enumerabile	AbB->bb (nessun vincolo)	Macchina di Turing
1	Grammatica legata al contesto	AB->Bc (sx mai più corta della dx)	Automi limitati lineari
2	Grammatica Libera dal contesto	A->AB (a sx solo un simbolo non terminale)	Automi a pila
3	Grammatica Regolare (lineare sx/dx)	A->aA|a (a destra solo terminale o terminale+non terminale)	automi a stati finiti

- Definizione di "Produzione" come funzione che, dato un carattere e uno stato, porta all'insieme delle parti degli stati dell'automa.

Attraverso la relazione di transizione δ (NT x S → P(S)) si determina lo stato o gli stati di destinazione in base allo stato corrente ed al simbolo letto. Se dopo aver letto l'ultimo simbolo la macchina si trova in almeno uno degli stati appartenenti ad F, la macchina accetta la stringa, altrimenti la rifiuta. L'insieme di tutte le stringhe accettate dall'automa a stati finiti non deterministico è il linguaggio accettato dall'automa.Il linguaggio accettato dagli automi a stati finiti non deterministico è un linguaggio regolare.

- Dimostrazione: un automa non deterministico può essere trasformato in automa deterministico. +, concatenazione e * attraverso automi

determinististici.



(per tabella vedere qualche domanda sopra)

Relazione d'equivalenza (1)

Per ogni automa a stati finiti non deterministico è possibile costruire un automa a stati finiti deterministico in grado di riconoscere lo stesso linguaggio utilizzando la costruzione dei sottoinsiemi.
Infatti i linguaggi accettati dai DFA e dagli NFA coincidono, poiché in un DFA si può vedere come δ **(q,a)** restituisce sempre insiemi costituiti da un solo stato (detti anche singoletti), si ha che ogni linguaggio regolare è un linguaggio accettato da un qualche NFA.

Relazione d’equivalenza (2)

Sia $A = (Q, \Sigma, \delta, q_0, F)$ un DFA, e $\{p, q\} \subseteq Q$. Definiamo

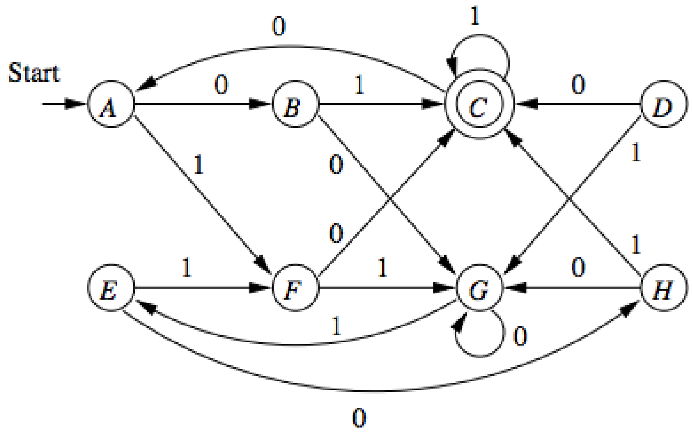
$$p \equiv q \;\Leftrightarrow\; \forall w \in \Sigma^* \;:\; \hat{\delta}(p, w) \in F \text{ se e solo se } \hat{\delta}(q, w) \in F$$

- Se $p \equiv q$ diciamo che p e q sono *equivalenti*
- Se $p \not\equiv q$ diciamo che p e q sono *distinguibili*

In altre parole: p e q sono distinguibili se e solo se

$$\exists w : \hat{\delta}(p, w) \in F \text{ e } \hat{\delta}(q, w) \notin F, \text{ o viceversa}$$

Ovvero se e solo se esiste almeno una stringa w che li differenzia, ovvero l'automa si muove da p in uno stato accettante e da q in uno stato non accettante (o viceversa).



$\hat{\delta}(C, \epsilon) \in F, \hat{\delta}(G, \epsilon) \notin F \Rightarrow C \not\equiv G$
$\hat{\delta}(A, 01) = C \in F, \hat{\delta}(G, 01) = E \notin F \Rightarrow A \not\equiv G$

- **invarianza a destra**
-

- **l'automa minimo.**