Definire formalmente un'espressione regolare e la nozione di implicazione fra espressioni regolari.

Un'espressione regolare è una stringa formata da caratteri dell'alfabeto terminale Σ e dai seguenti operatori: concatenamento, unione, star, insieme vuoto e parantesi.

Di seguito degli esempi di espressioni regolari:

-espressione regolare = ϕ (insieme vuoto)

-espressione regolare = Ɛ (stringa vuota)

-espressione regolare = a (con a ∈ Σ, cioè qualsiasi simbolo che appartiene all'alfabeto)

-espressione regolare = e1 U e2 (unione di due espressioni regolari)

-espressione regolare = e1* e2* (star di due espressioni regolari)

-espressione regolare = e1 · e2 (concatenamento di due espressioni regolari)

Definire che cos'è un linguaggio regolare

Un linguaggio di alfabeto Σ si dice regolare se è esprimibile mediante le operazioni di concatenamento, unione e star applicate per un numero finito di volte ai linguaggi unitari o al linguaggio vuoto.

Definire formalmente l'automa a stati finiti deterministico senza Ɛ-mosse M=<Q, Σ, ∂, q_0, F> e definire un automa non deterministico M =<Q', Σ', ∂', q_0', F'> con Ɛ-mosse

(nota: Q', Σ', ∂', q_0', ed F' devono essere definiti formalmente in termini di Q, Σ, ∂, q0 e F.

L'automa a stati finiti deterministico, quindi con le Ɛ mosse, è definito da cinque elementi:

<Q, Σ, ∂, q_0, F>

Q, rappresenta l'insieme degli stati

Σ, rappresenta l'alfabeto

∂, rappresenta la funzione di transizione, cioè cosa succede ad un certo stato con un certo input.

∂ = Q x Σ -> Q, significa che uno stato con un certo input può andare solo ed esclusivamente in un altro stato.

q_0, rappresenta lo stato iniziale

F, è un sottoinsieme di Q, rappresenta l'insieme degli stati finali

L'automa a stati finiti non deterministico quindi senza le Ɛ mosse, è definito da cinque elementi:

<Q, Σ, ∂, q0, F>

Q, rappresenta l'insieme degli stati

Σ, rappresenta l'alfabeto

∂, rappresenta la funzione di transizione, cioè cosa succede ad un certo stato con un certo input.

∂ = Q x Σ -> 2Q, significa che uno stato con un certo input può andare in altri stati.

q_0, rappresenta lo stato iniziale

F, è un sottoinsieme di Q, rappresenta l'insieme degli stati finiti

Definire le nozioni di configurazioni e di mossa per un automa a pila.

La configurazione mi indica il punto in cui sono arrivato durante la computazione.

(q, y, ɣ) ∈ Q x Σ* x T*,

q è lo stato attuale della computazione,

y è la stringa in input che dovrà essere letta/consumata,

ɣ è il contenuto della pila.

Data una configurazione si chiama **mossa** il passaggio da una configurazione ad un'altra tramite la funzione di transizione.

Ci sono due casi: (ɣ, x ∈ T*)

(q, aY, ɣx) -> (p, Y, ɣπ) se (p, π) ∈ ∂(q, a, x), cioè si consuma un elemento dall'input e si fa una push sulla pila

(q, Y, ɣx) -> (p, ɣπ) se (p, π) ∈ ∂(q, Ɛ, x)

Definire le regole che permettono di costruire in modo automatico a partire da un automa a stati finiti che riconosce il linguaggio complemento.

L'automa complemento di un automa a stati finiti deterministico, è definito da cinque elementi: $<Q', \Sigma', \partial', q_0', F'>$.

q_0' = q_0, cioè rimane sempre lo stesso stato iniziale dell'automa a stati finiti

F' = (Q - F) U {P}, cioè tutti gli stati non finali dell'automa a stati finiti e lo stato P

Q' = Q U {P}, cioè uguale a Q dell'automa a stati finiti con l'aggiunta dello stato P, detto stato pozzo.

Σ', rimane lo stesso di quello dell'automa di partenza

∂', viene suddivisa in casi: (per ogni $q \in Q$ e per ogni $a \in \Sigma$)

- se $\partial(q, a) = k$ allora $\partial'(q, a) = k$, cioè ogni arco dell'automa a stati finiti ci sarà anche nell'automa complemento.
- se $\partial(q, a) = /$ allora $\partial'(q, a) = P$, cioè se nell'automa a stati finiti non c'era l'arco con un simbolo dell'alfabeto, allora in quello complemento andrà nello stato P.
- $\partial'(p, a) = P$, cioè si crea una arco che va da P a P per tutti i simboli dell'alfabeto.

Definire formalmente le regole che permettono di costruire in modo automatico un automa a pila non deterministico data una grammatica context-free.

Si utilizza un'accettazione a pila vuota e occorrono sette entità $<Q, \Sigma, T, \partial, q_0, Z_0, F>$:

- **q_0**, ovvero lo stato iniziale.
- **F**, ovvero l'insieme degli stati finali che in questo caso sarà uguale all'insieme vuoto.
- **$Q = \{q_0\}$**, esiste solo lo stato iniziale.
- **Σ = Σ**, l'alfabeto dell'input è sempre lo stesso della grammatica context-free.
- **$Z_0 = Z_0$**, indica il fondo della pila.
- **$T = \{Z_0\} \cup \Sigma \cup V$**, cioè sulla pila ci sono tanti simboli quanti sono i simboli dell'alfabeto e dei non terminali
- **∂** è la funzione di transizione che genera una serie di regole di produzione,

cioè per ogni $A \rightarrow \alpha\beta$: ($\beta^r$ ovvero i simboli dopo α bisogna scriverli al contrario)

- se α = terminale → $\partial(q_0, a, A) = (q_0, \beta^r)$
- se α = non terminale → $\partial(q_0, \varepsilon, A) = (q_0, \beta^r X)$
- viene gestito ogni simbolo dell'alfabeto, cioè per ogni $A \in \Sigma \rightarrow \partial(q_0, a, a) = (q_0, \varepsilon)$
- vengono gestite le regole della funziona di transizione di inizializzazione e di terminazione.
- inizializzazione → $\partial(q_0, \varepsilon, Z_0) = (q_0, ⇙S)$
- terminazione → $\partial(q_0, ⇙, ⇙) = (q_0, \varepsilon)$

Definire l'accettazione per un input negli automi a pila

L'accettazione di un input in un automa a pila non deterministico avviene per:

- **stato finale**, cioè l'input si accetta quando si consuma tutta l'input e si arriva in uno stato finale $(q_0, A, Z_0) \rightarrow (q, \varepsilon, \gamma)$ se $q \in F$
- **pila vuota**, cioè quando si consuma tutto l'input e alla fine la pila è vuota, cioè non contiene nessun simbolo, nemmeno Z_0 $(q_0, A, Z_0) \rightarrow (q, \varepsilon, \varepsilon)$ se $q \in Q$

Definire formalmente l'automa a pila

L'automa a pila è definito da 7 elementi: $<Q, \Sigma, T, \partial, q_0, Z_0, F>$.

Q, rappresenta l'insieme degli stati.

Σ, rappresenta l'alfabeto dell'input.

T (tao), rappresenta l'alfabeto della pila, cioè i simboli che si possono mettere o togliere dalla pila.

∂, rappresenta la funzione di transizione, cioè cosa succede ad un certo stato con un certo input.

q_0, rappresenta lo stato iniziale.

Z_0, rappresenta la pila vuota ed è un simbolo della pila.

F, è un sottoinsieme di Q, rappresenta l'insieme degli stati finali.

Definire la grammatica context-free (nota è la stessa per le grammatiche dei linguaggi regolari)
Una grammatica context-free è definita da quattro entità: $G=(V, \Sigma, P,S)$
V, è l'insieme dei simboli non terminali che appartengono alla grammatica,
Σ, è l'insieme dei simboli terminali che appartengono alla grammatica,
P, è l'insieme delle regole di produzione,
S, è il non terminale specificato come assioma, ($S \in V$)

Dire perché una grammatica G può essere ambigua
Una grammatica è ambigua quando per una frase x del linguaggio della grammatica G se ha due o più alberi sintattici distinti.

Definizione di go to
Go to (I, x): Mi permette di spostarmi da uno stato ad un altro all'interno di un automa consumando un terminale o un non terminale.

$\forall\ A \rightarrow \alpha^{\nabla} X\ \beta$ e I

Chiusura($\{A \rightarrow \alpha X^{\nabla}\ \beta\}$) e goto(I, x) con x che può essere un non terminale o un terminale