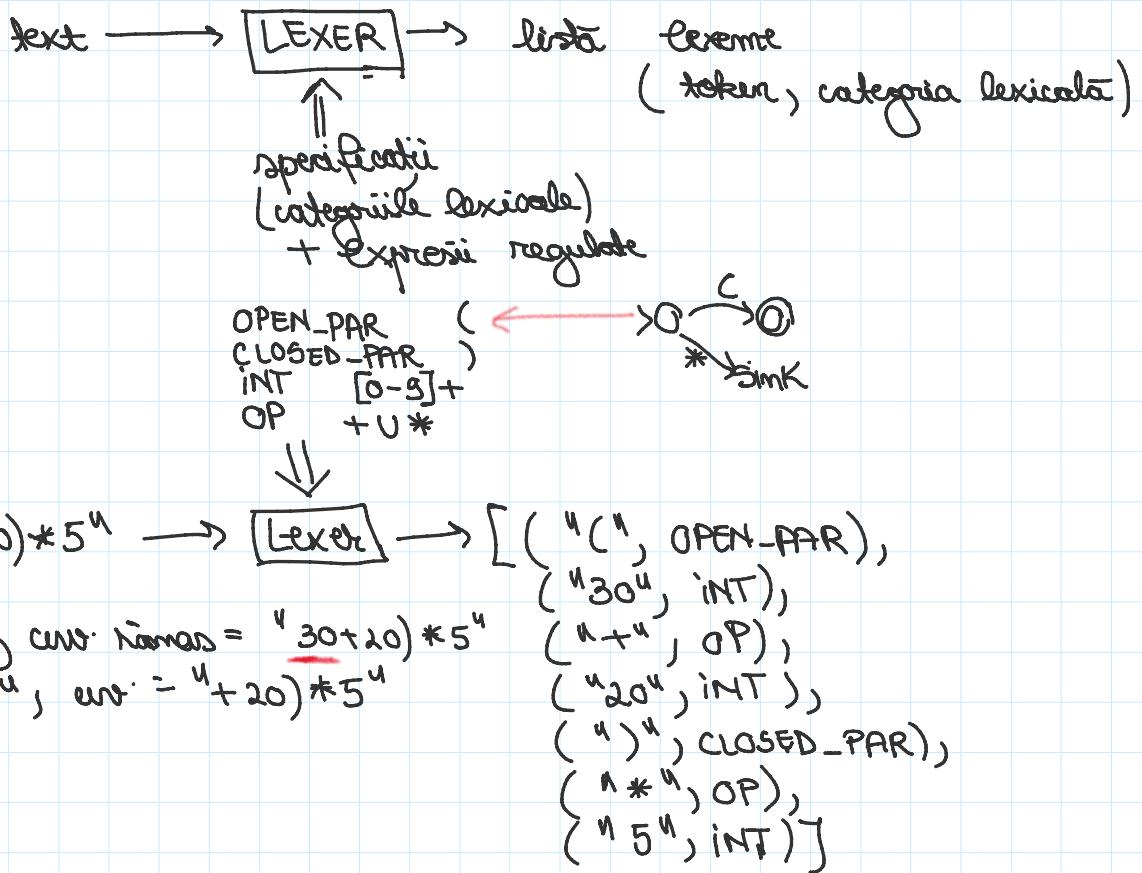


Lexer

→ Finalitate lexicală



Pt. Lab 4 și Proiect Etapa 1: Spec = o lista de DFA -uri.

Pt. vîrstă: Spec = o lista de expr. regulate (conversie expr. reg. → DFA)

Alg. lexer:

- 1) începe cu toate DFA-urile în stare initială.
- 2) rulează toate DFA-urile în paralel pe input.
- 3) când toate DFA-urile au dat REJECT sau au ajuns într-un sink state, alege ultimul DFA care a acceptat, adaugă lexem -ul la output și repeta ① pt. restul inputului.

2 principii:

→ maximal match: se alege cel mai lung prefix acceptat de 1 DFA

- maximal match: se algește cel mai lung prefix acceptat de 1 DFA
- în caz că există mai multe DFA -uri care acceptă cel mai lung prefix, se algește primul DFA din Spec

Ex:

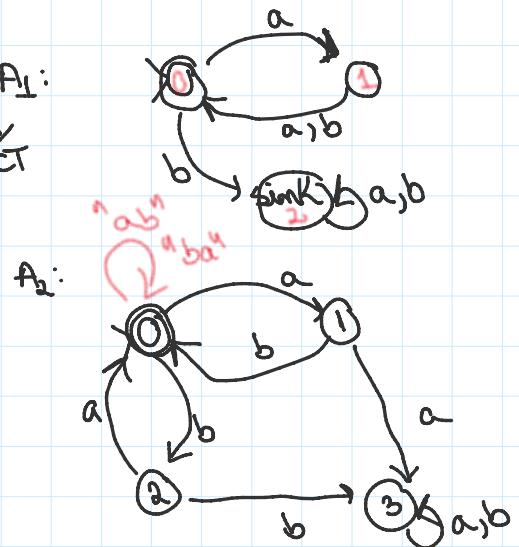
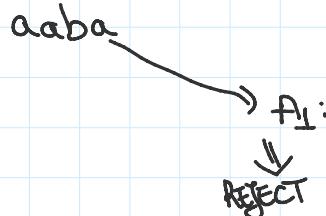
3.1.1.  $w = \underline{aaba}aaabb$

prefix max = aa

3.1.2.  $w = ababba\underline{abb}abaab$ ,  $\text{len}(w) = 14$

lexeme =  $\{("ababbaabbabaab", A_2)\}$

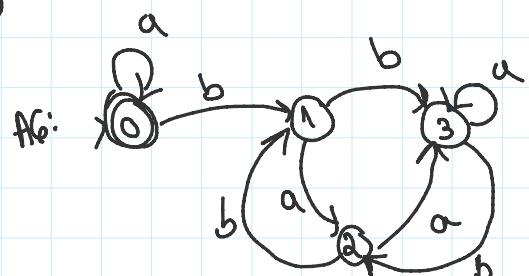
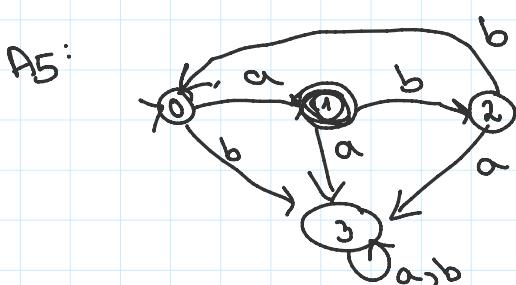
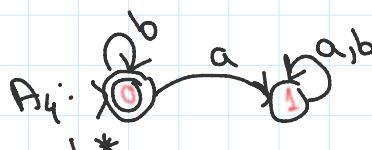
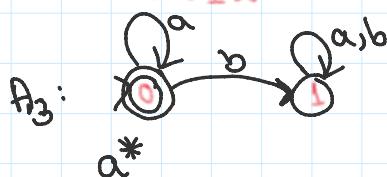
ababbaabbabaab



3.1.3.

$w = \underline{ab}aaa\underline{abb}abaaaab$

$A_3: 0 \xrightarrow{a} 1$	$A_4: 0 \xrightarrow{a} 1 \xrightarrow{b} 2$	$A_5: 0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 3$
$A_3: 1 \xrightarrow{b} 0$	$A_4: 1 \xrightarrow{a} 0$	$A_5: 1 \xrightarrow{a} 0$
$A_3: 0 \xrightarrow{b} 2$	$A_4: 2 \xrightarrow{b} 1$	$A_5: 2 \xrightarrow{b} 1$
$A_3: 1 \xrightarrow{a} 0$	$A_4: 0 \xrightarrow{b} 1$	$A_5: 0 \xrightarrow{a} 1$
$A_3: 2 \xrightarrow{a} 1$	$A_4: 1 \xrightarrow{a} 0$	$A_5: 1 \xrightarrow{a} 0$
$A_3: 0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 3$	$A_4: 0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 3$	$A_5: 0 \xrightarrow{a} 1 \xrightarrow{b} 2 \xrightarrow{a} 3$



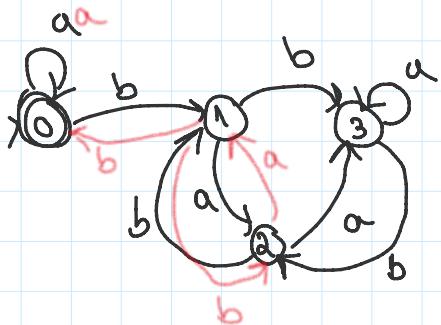
3.2.1.  $w = \underline{def} \underline{\omega} \underline{\text{deffunction}} \underline{\omega}$

A: " $[a-z]^+ \cup$ "

B: "def"

1) Spec :  $\begin{bmatrix} A \\ B \end{bmatrix}$

$\rightarrow$  Lexeme =  $\left[ \begin{array}{l} ("def\_\!\!\_w", A), \\ ("deffunction\_\!\!\_w", A) \end{array} \right]$



2) Spec :  $\begin{bmatrix} B \\ A \end{bmatrix}$

$\rightarrow$  Lexeme =  $\left[ \begin{array}{l} ("def\_\!\!\_w", B), \\ ("deffunction\_\!\!\_w", A) \end{array} \right]$

Alg. găsire sink states:

- $\rightarrow$  rev-delta = funcția inversă de tranziție
- $\rightarrow$  BFS initializată cu stările finale
- $\text{visited} = \text{set}()$   $\leftarrow$  pt. a vizita & săg. datea filtrare store
- xrate din coada um mod 5
- parcurgem toti verii  $v$
- $\forall c \in \Sigma$ ,  $\text{rev-delta}[(v, c)]$
- if  $v$  not in visited:
- adăugăm  $v$  în coada visited-add( $v$ )
- $\rightarrow$  sink states = acele stări care nu apar în visited