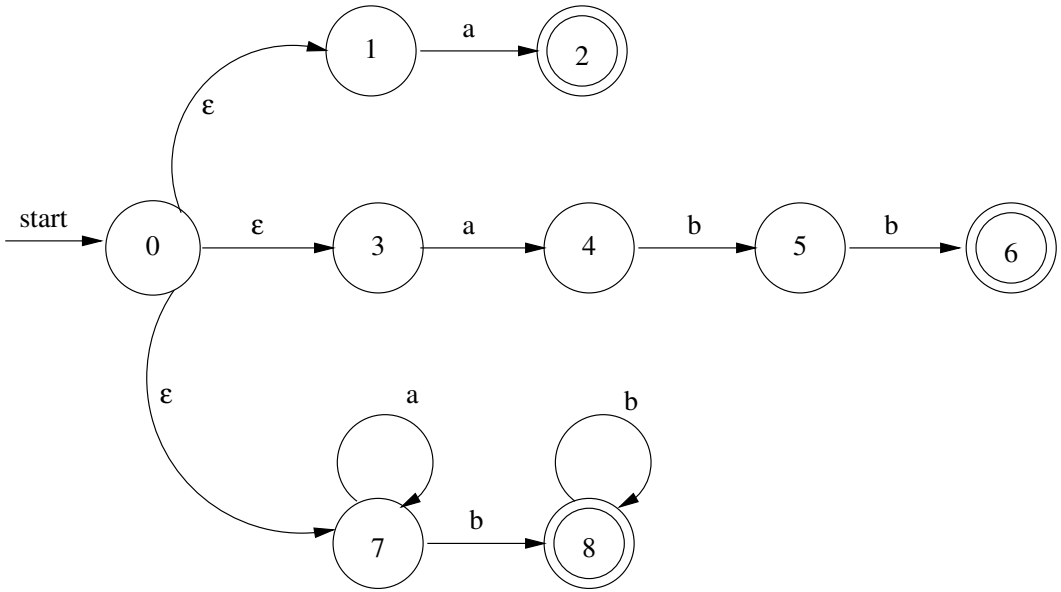
Recall of Implementation of LA: Example

- Let R be :

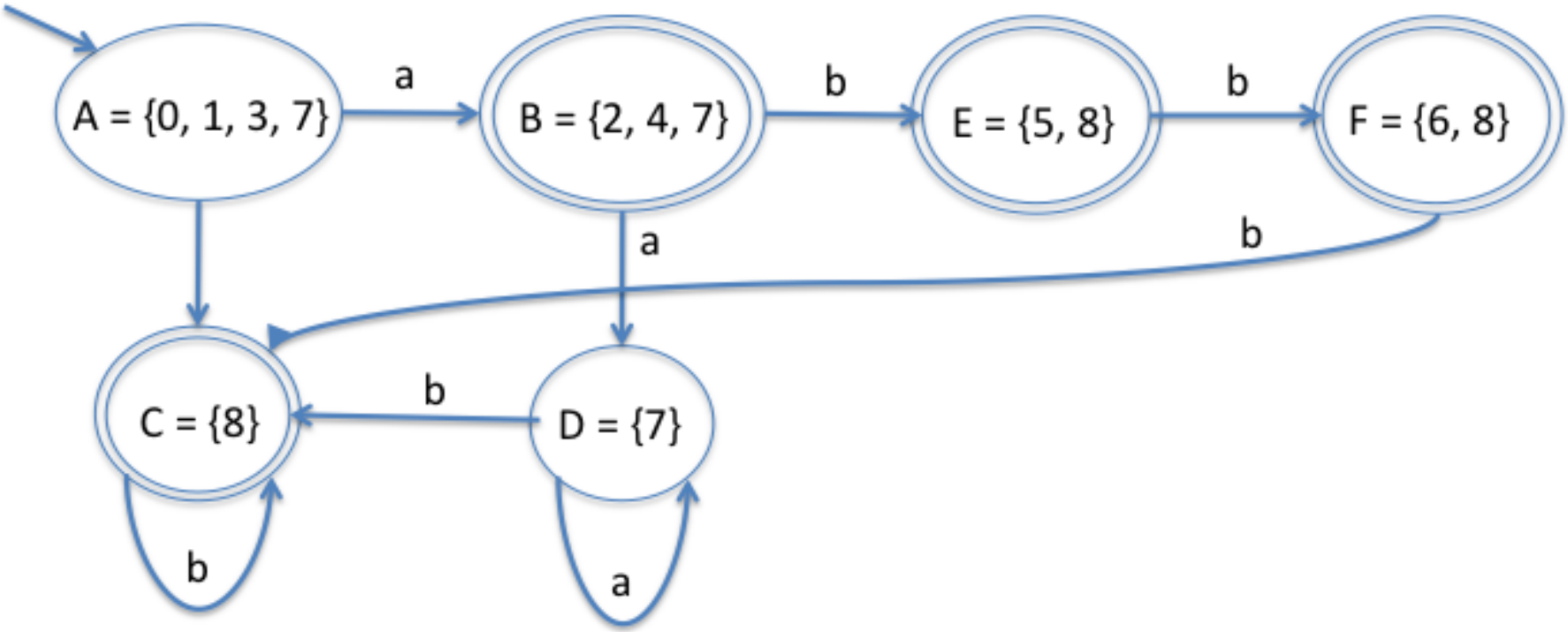
$$\begin{array}{lll} d_1 = & a & \{\texttt{TOKEN1}\} \\ d_2 = & abb & \{\texttt{TOKEN2}\} \\ d_3 = & a^* b^+ & \{\texttt{TOKEN3}\} \end{array}$$

- The combined NFA of the three NFAs obtained from d_1, d_2 and d_3 is the following (the NFA for d_3 is simplified, actually made deterministic):



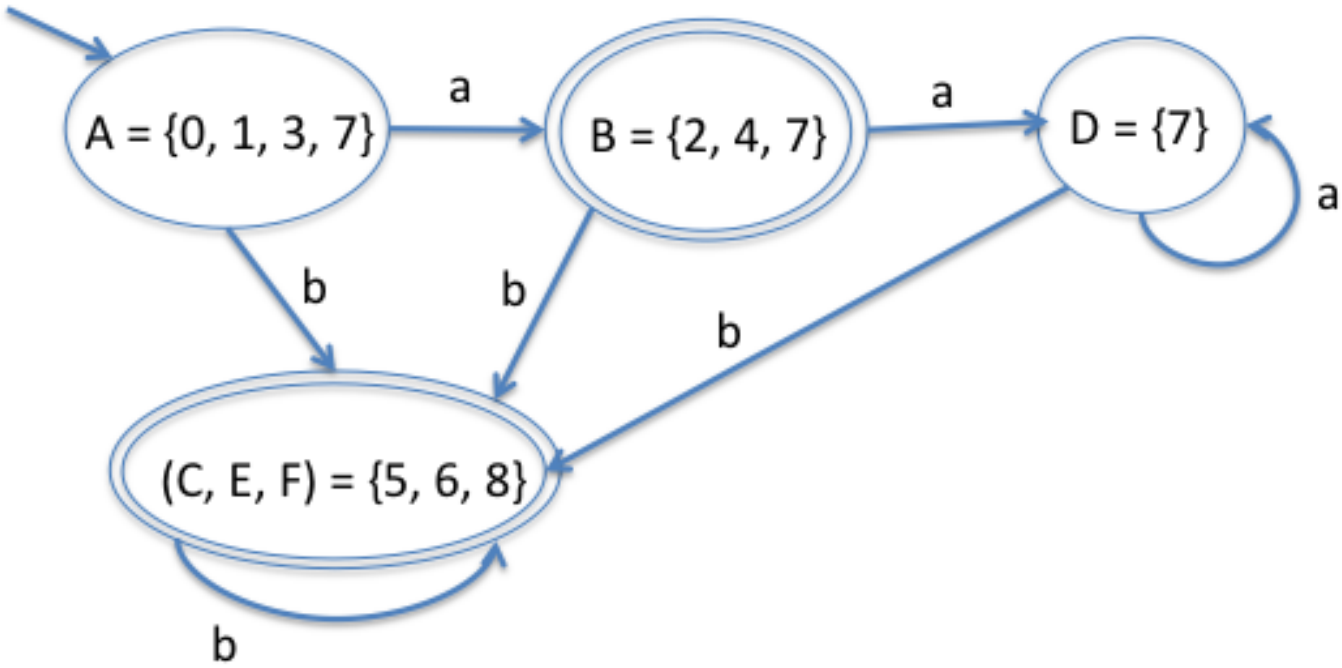
Implementation of LA: Optimisation

- The behaviour of the LA can be optimised by determinizing the NFA and then by minimising the states
- The DFA obtained from the combined NFA for R is:



Implementation of LA: Optimisation

- By performing a standard minimisation the following minimal DFA is obtained:



Implementation of LA: Optimisation

- Let's scan the input *aaba*
- $A \xrightarrow{a} B$, $\texttt{Last\_Final} = \{2\}$, $\texttt{Input\_Pos\_at\_Last\_Final} = 1$
- $B \xrightarrow{a} D$
- $D \xrightarrow{b} (C, E, F)$, $\texttt{Last\_Final} = \{6, 8\}$, $\texttt{Input\_Pos\_at\_Last\_Final} = 3$
- $(C, E, F) \not\xrightarrow{a}$
- The LA cannot decide which token to output! Final state 6 would call for TOKEN 2 (incorrect!) and final state 8 would call for TOKEN 3!

We need to retain the information on the final states!

Implementation of LA: Optimisation

- We must start the minimisation of the DFA by initially splitting the group of final states into subgroups
- A subgroup for each set of reached final states must be created
- subgroup 1 = $\{B\}$ for TOKEN 1 - only final state 2
- subgroup 2 = $\{C, E\}$ for TOKEN 3 - only final state 8
- subgroup 3 = $\{F\}$ for TOKEN 2 and TOKEN 3 - final states $\{6, 8\}$
- The other non-final states can be grouped together as usual

$$\Pi_1 = \{(A, D), (B), (C, E), (F)\}$$

Implementation of LA: Optimisation

- The group (A, D) can be refined: $A \xrightarrow{a} B$ and $D \xrightarrow{a} D$
- $\Pi_2 = \{(A), (D), (B), (C, E), (F)\}$
- The group (C, E) can be refined: $C \xrightarrow{b} C$ and $E \xrightarrow{b} F$
- $\Pi_3 = \{(A), (D), (B), (C), (E), (F)\}$
- Π_3 cannot be refined further!
- The minimal DFA to use for the LA scanning is just the same DFA

Implementation of LA: Optimisation

- Let's scan the input *aaba*
- $A \xrightarrow{a} B$, $\texttt{Last\_Final} = \{2\}$, $\texttt{Input\_Pos\_at\_Last\_Final} = 1$
- $B \xrightarrow{a} D$
- $D \xrightarrow{b} C$, $\texttt{Last\_Final} = \{8\}$, $\texttt{Input\_Pos\_at\_Last\_Final} = 3$
- $C \not\xrightarrow{a}$
- The LA outputs TOKEN 3 with lexeme *aab*, then clear the recognised input and restart
- $A \xrightarrow{a} B$, $\texttt{Last\_Final} = \{2\}$, $\texttt{Input\_Pos\_at\_Last\_Final} = 1$
- $B \not\longrightarrow$ end of input
- The LA outputs TOKEN 1 with lexeme *a*, then stops.

Summary

Lexical Analysis

Relevant concepts we have encountered:

- Tokens, Patterns, Lexemes
- Chomsky hierarchy and regular languages
- Regular expressions
- Problems and solutions in matching strings
- DFA and NFA
- Transformations
  - RegExp $\rightarrow$ NFA
  - NFA $\rightarrow$ DFA
  - DFA $\rightarrow$ Minimal DFA
- Implementation and optimisation of LA