Compiler theory
MS1

The Tiny Language

Team : G3_56

Team Members:

NAME	ID	SECTION
كريم السيد عبدالقوي	20201700600	6
محمد هشام زين العابدين	20201700754	8
عبدالله مصطفي عبدالسلام	20201700493	5

Lexemes & Tokens

Lexemes	Tokens	Lexemes	Tokens
/**/	T_Comment	+	T_Arithmatic_Operator
int	T_Datatype	main	T_ Reserved_Keywords
sum	T_FunctionName	"Iteration number ["	T_String
(	T_OP	3	T_Number
a	T_identifier	||	T_Boolean_Operator
,	T_OP	=	T_ Condition_Operator
{	T_OP	----	----

Regular expression

Digit := [0-9]

OP := (. | ; | , | (|) | := | < | > | ! | + | _ | * |=)

Letter := [a-z]|[A-Z]

1) Number := (+|-)? Digit^{+}(.Digit^{+})?
2) String := (")(Number|letter|op|whitespace)$^{+}$(")
3) Reserved_Keywords := (*int*| *string* | *read* |float| *write* | *if*| *until* | *repeat*| *elseif*| *else*| *then*| *return*| *endl*)
4) Comment_Statement := (/*)(Digit|letter|op|whitespace)$^{+}$ (*/)
5) Identifier := Letter (Letter|Digit)*
6) Function_Call := Identifier(()(Identifier(,Identifier)*)*())
7) Term := Number | Identifier | Function_Call
8) Arithmatic_Operator := (+ | - | * | /)
9) Equation := (Term | ( )(Term | Arithmatic_Operator)$^{+}$())$^{+}$
10) Expression :=(String | Term | Equation)
11) Assignment_Statement := Identifier (:=)Expression
12) Datatype := (int|float|string)
13) Declaration_Statement := Datatype (Identifier | Assignment_Statement) (, Identifier |, Assignment_Statement)*;
14) Write_Statement := write(expression|endl);
15) Read_Statement := read(Identifier);
16) Return_Statement := return (Expression);
17) Condition_Operator := < | > | = | <>
18) Condition := (Identifier) (Condition_Operator) (Term)
19) Boolean_Operator := (&&)|(||)
20) Condition_Statement := Condition((Boolean_Operator)(Condition))*
21) If_Statement:=if(Condition_Statement)(then)(Write_Statement|Read_Statement|Assignment_Statement)(Else_If_Statement|Else_Statement|end)*
22) Else_If_Statement:= elseif(Condition_Statement)(then)(Write_Statement|Read_Statement|Assignment_Statement)(Else_If_Statement|Else_Statement|end)*
23) Else_Statement := else)(Write_Statement|Read_Statement|Assignment_Statement)(Else_If_Statement|Else_Statement)*end

24) Repeat_Statement := repeat(Write_Statement|Read_Statement|Assignment_Statement)(Else_If_Statement|Else_Statement)*(until)(Condition_Statement)

25) FunctionName := Identifier

26) Parameter := Datatype (Identifier)

27) Function_Declaration := Datatype(FunctionName)(()((Parameter)(,Parameter)*)?())

28) Function_Body := {(Write_Statement|Read_Statement|Assignment_Statement)(Else_If_Statement|Else_Statement)*(Return_Statement)}

29) Function_Statement :=(Function_Declaration)(Function_Body)

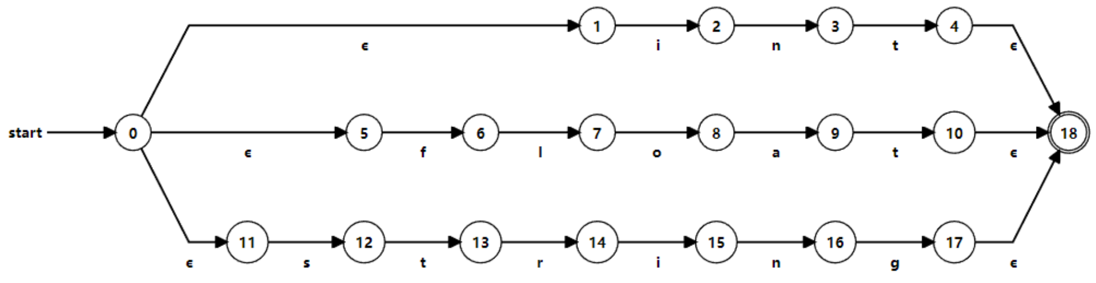
30) Main_Function := Datatype(main)(())(Function_Body)

31) Program := (Function_Statement)* (Main_Function)

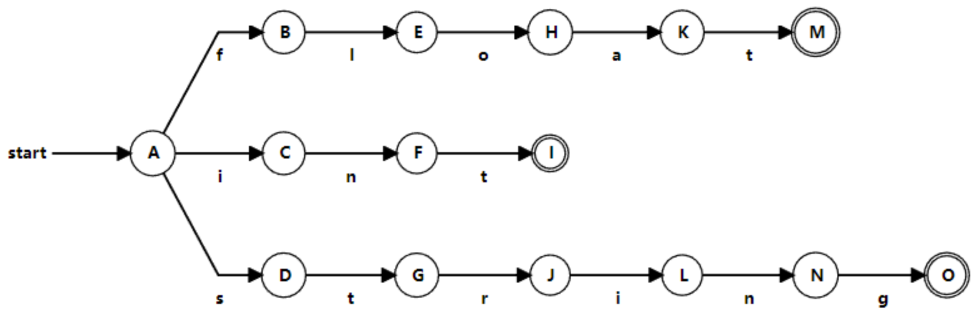
NFAs & DFAs & MIN DFAs

Datatype (NFA-DFA-MIN DFA):

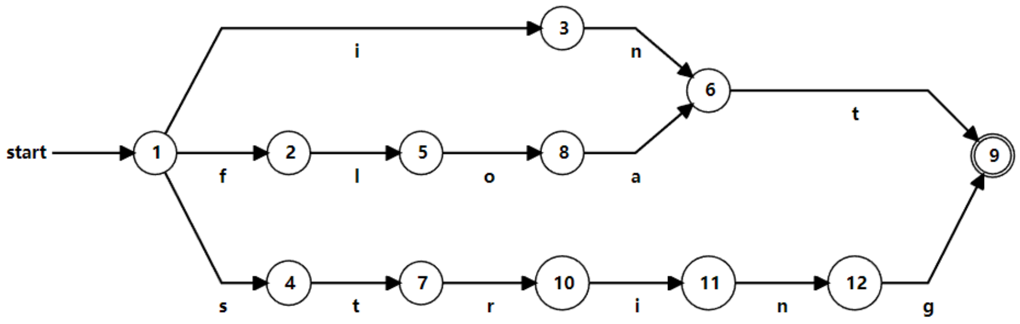
-NFA



-DFA

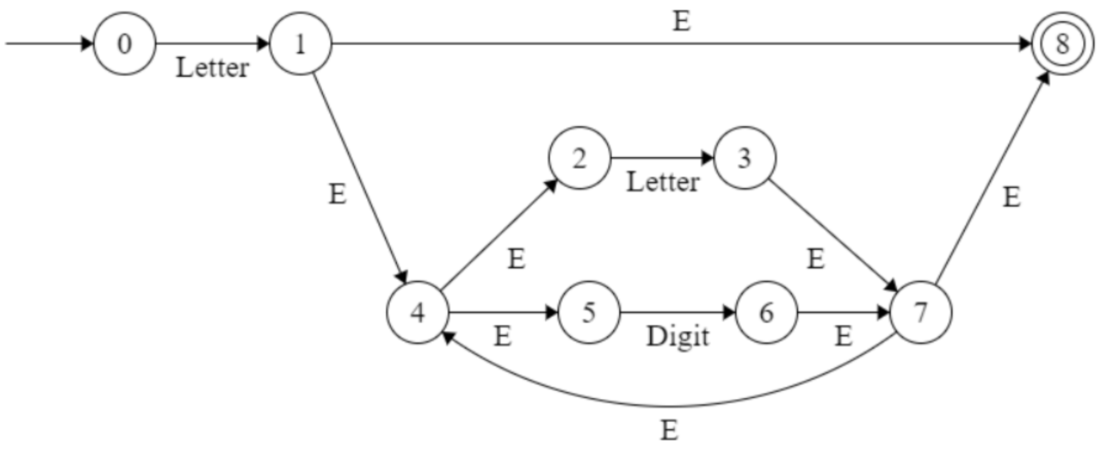


-MIN DFA

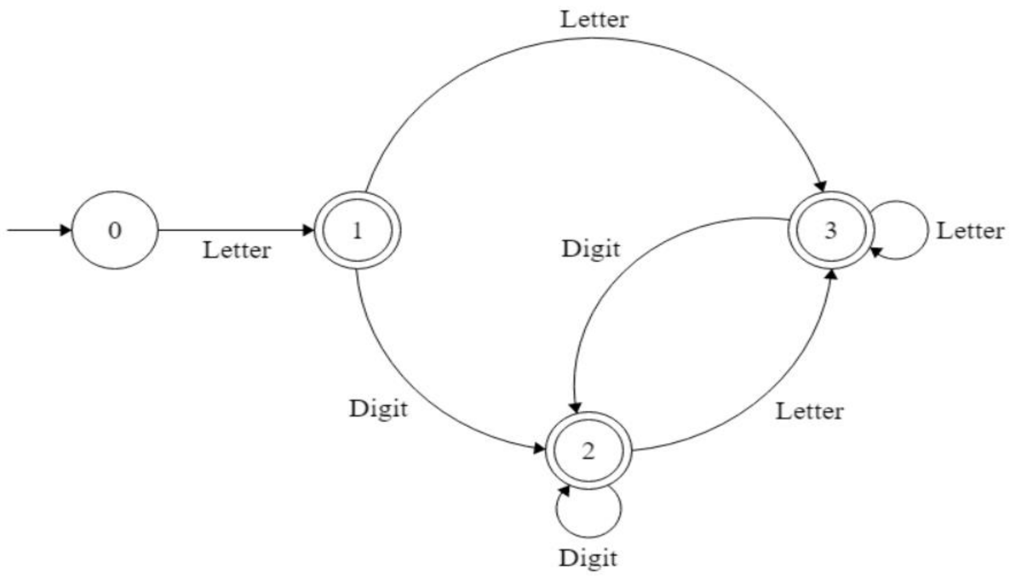


Identifier & Function_name (NFA-DFA-MIN DFA):

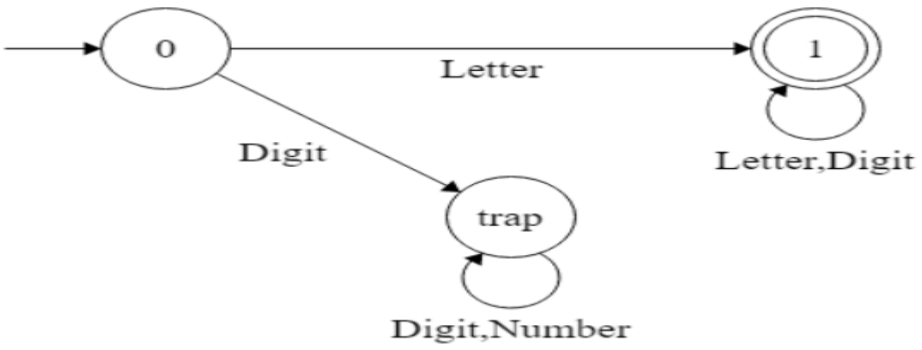
-NFA



-DFA

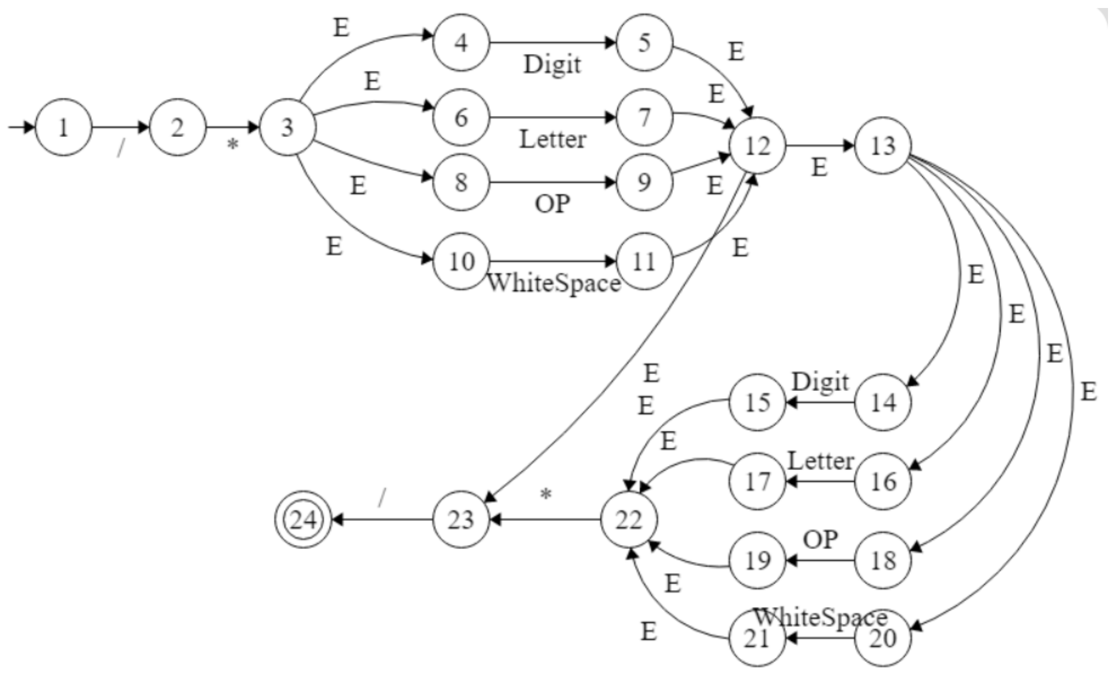


-MIN DFA

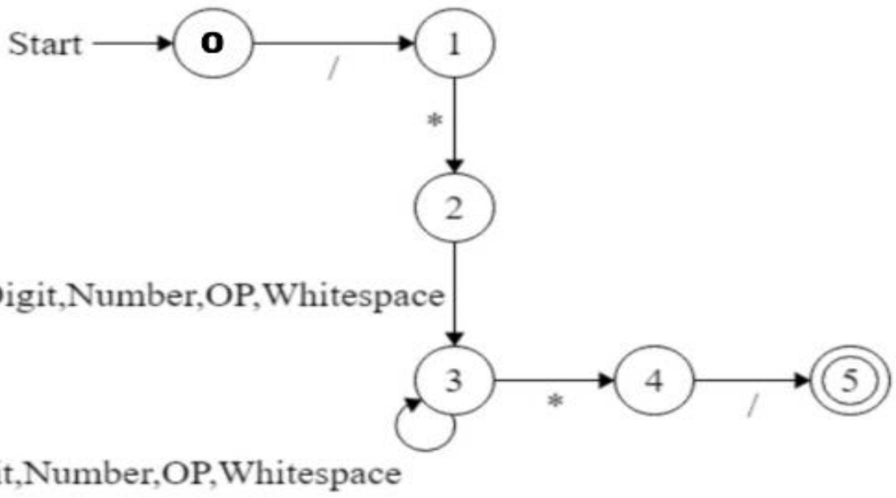


Comment (NFA-DFA-MIN DFA):

-NFA

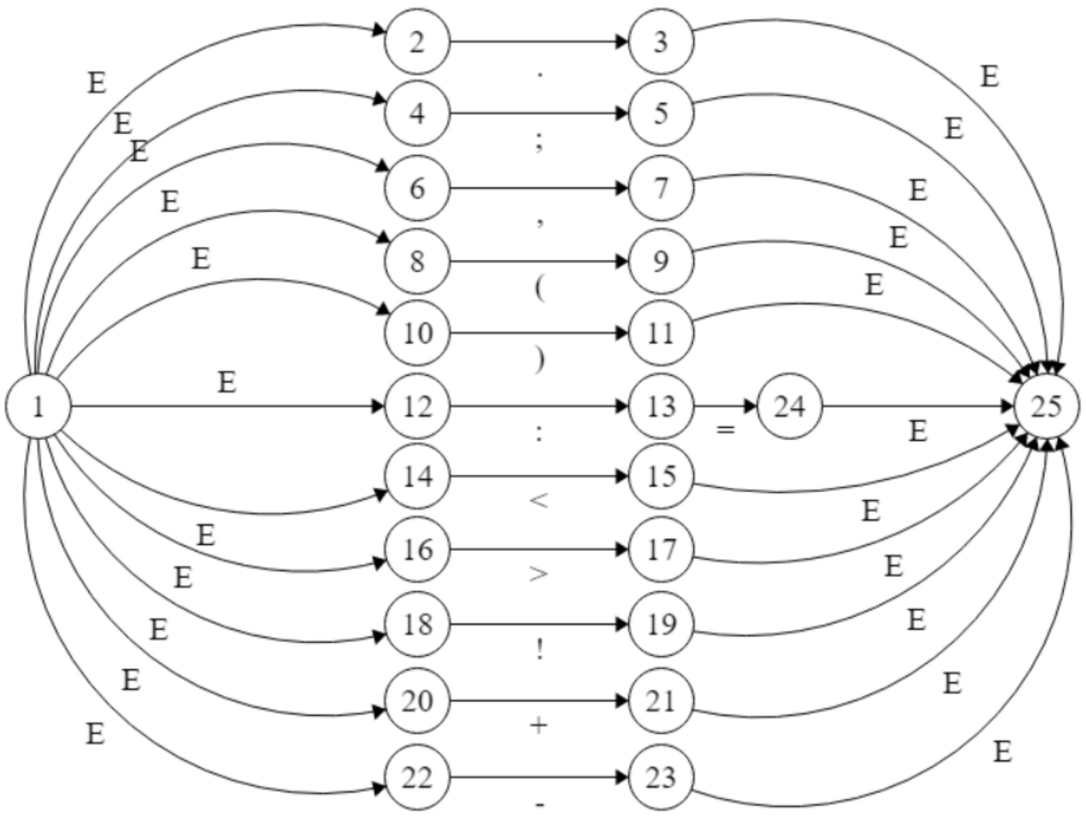


-DFA

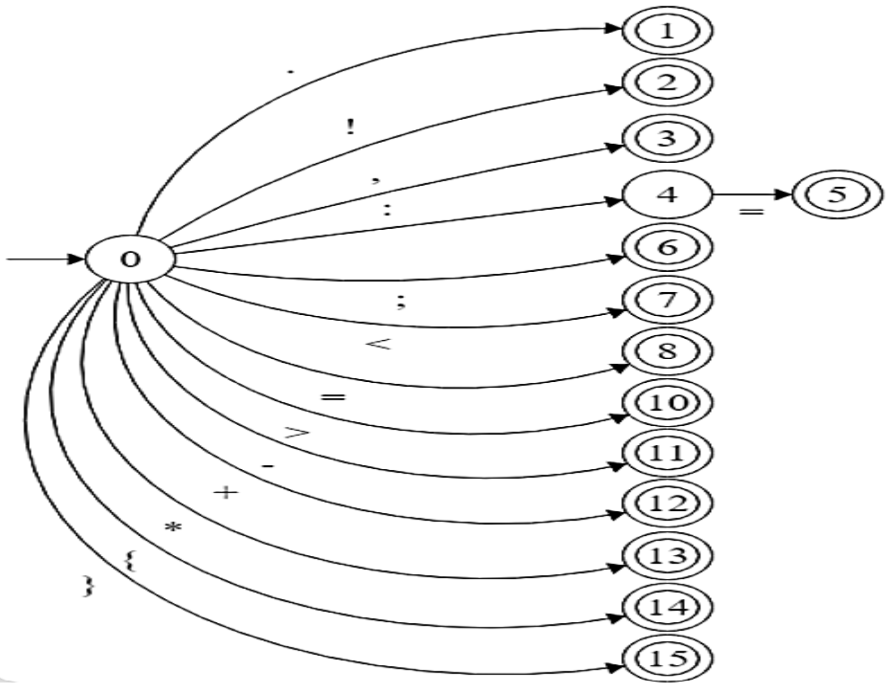


OP (NFA-DFA-MIN DFA):

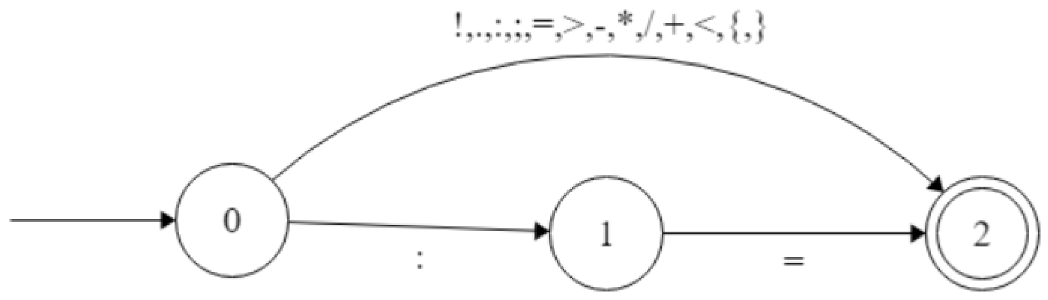
-NFA



-DFA

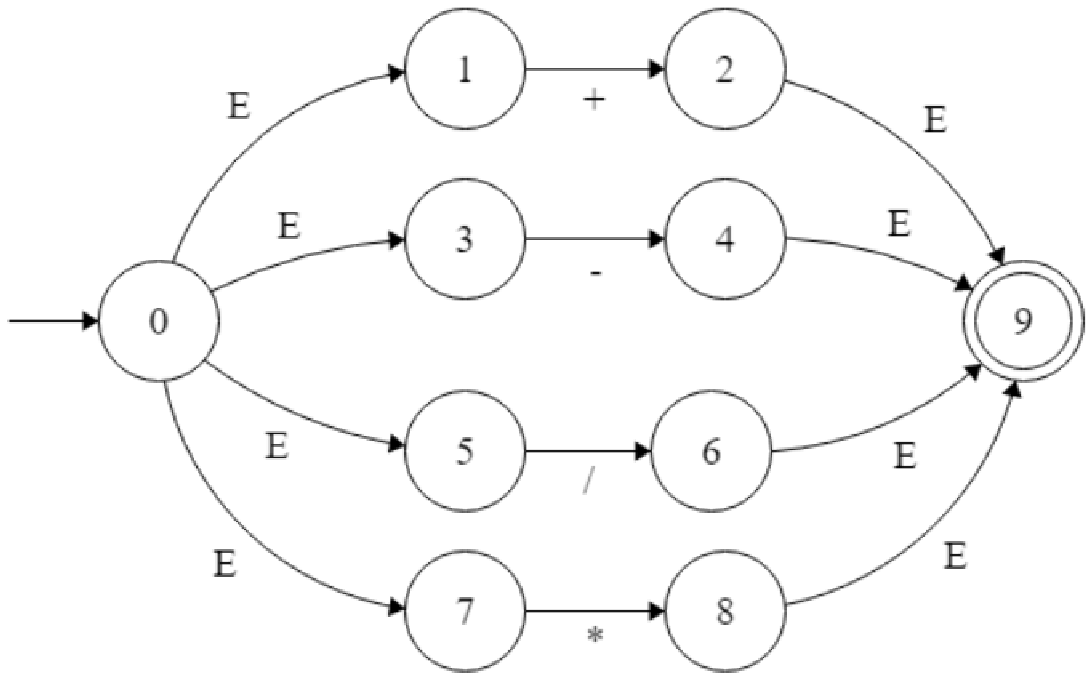


MIN DFA:

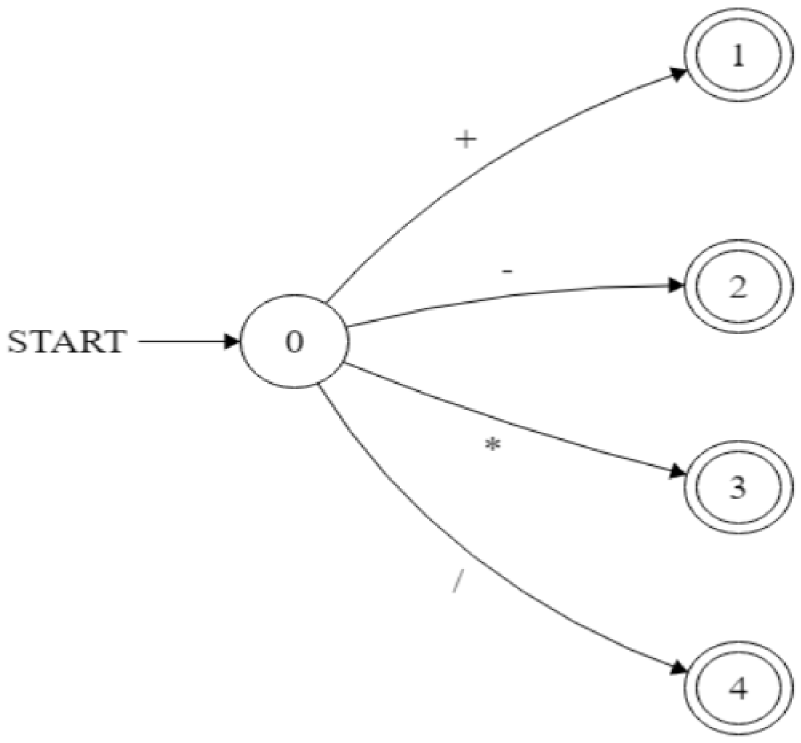


Arithmatic_Operator (NFA-DFA-MIN DFA):

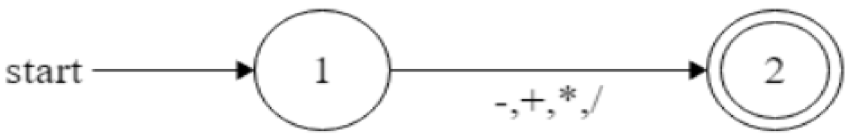
-NFA



-DFA

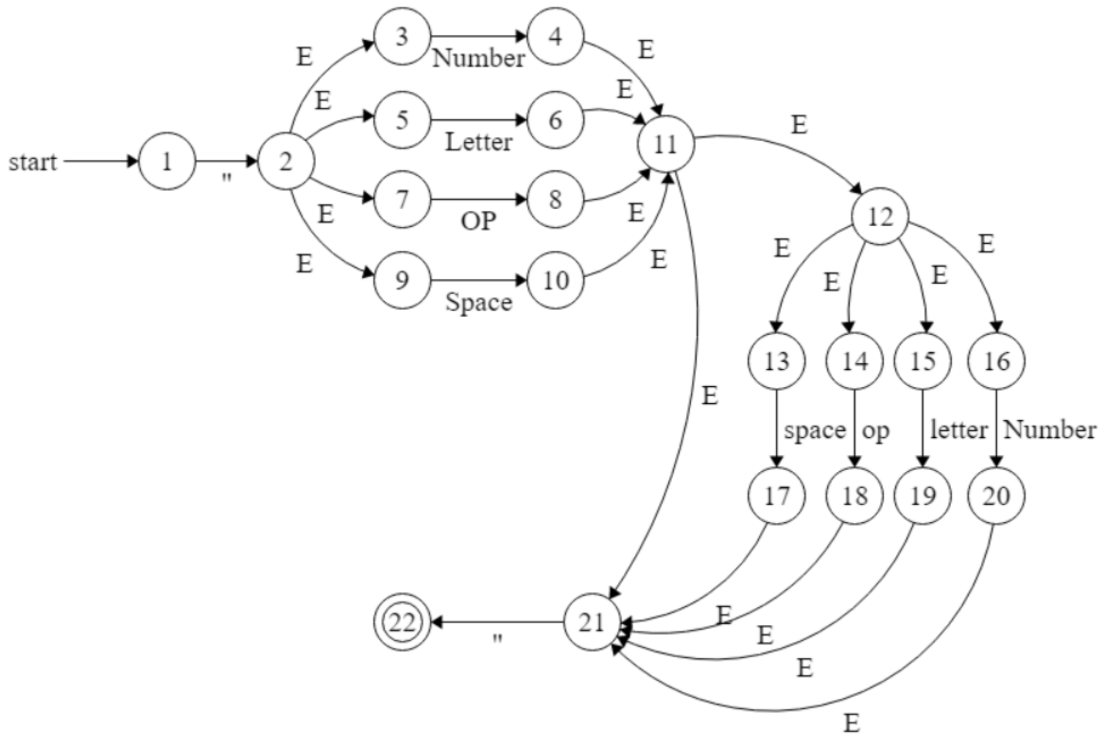


-MIN DFA

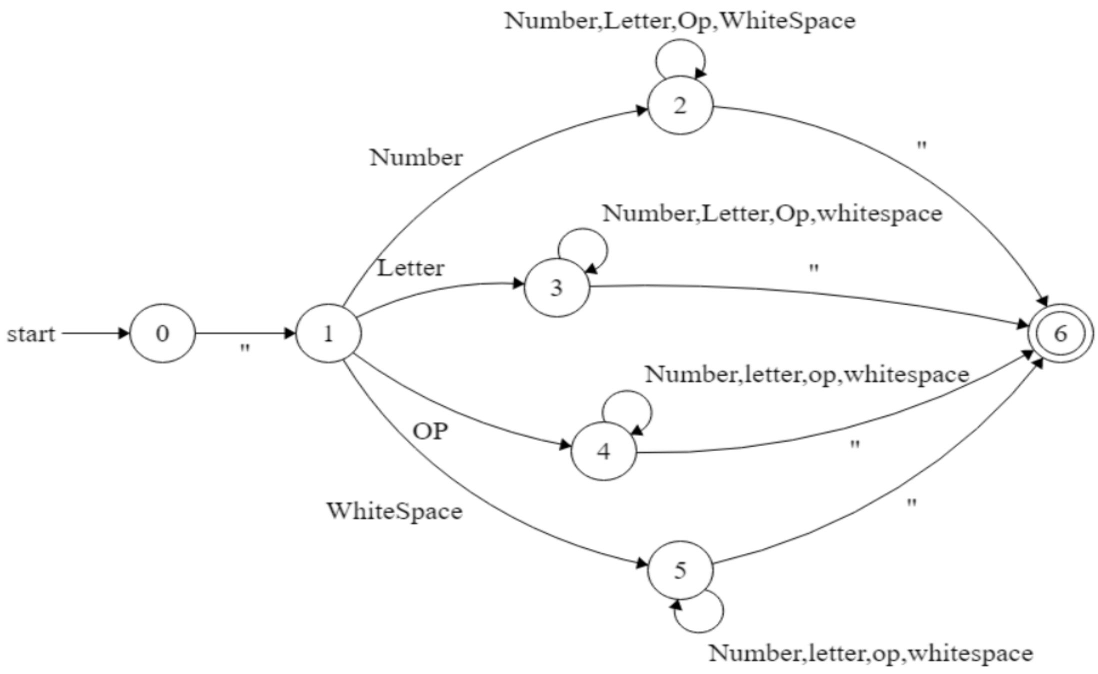


String (NFA-DFA-MIN DFA):

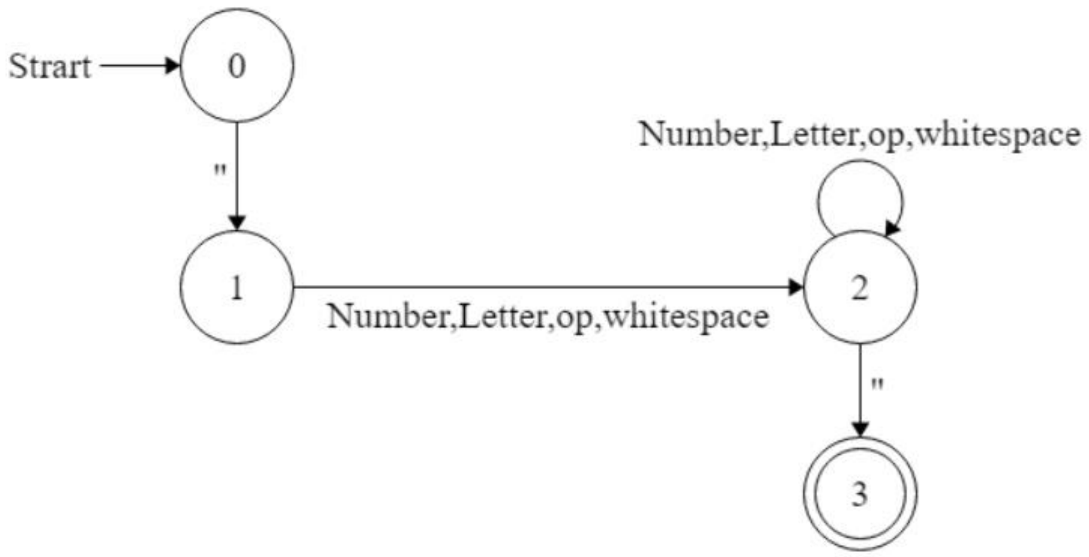
-NFA



-DFA

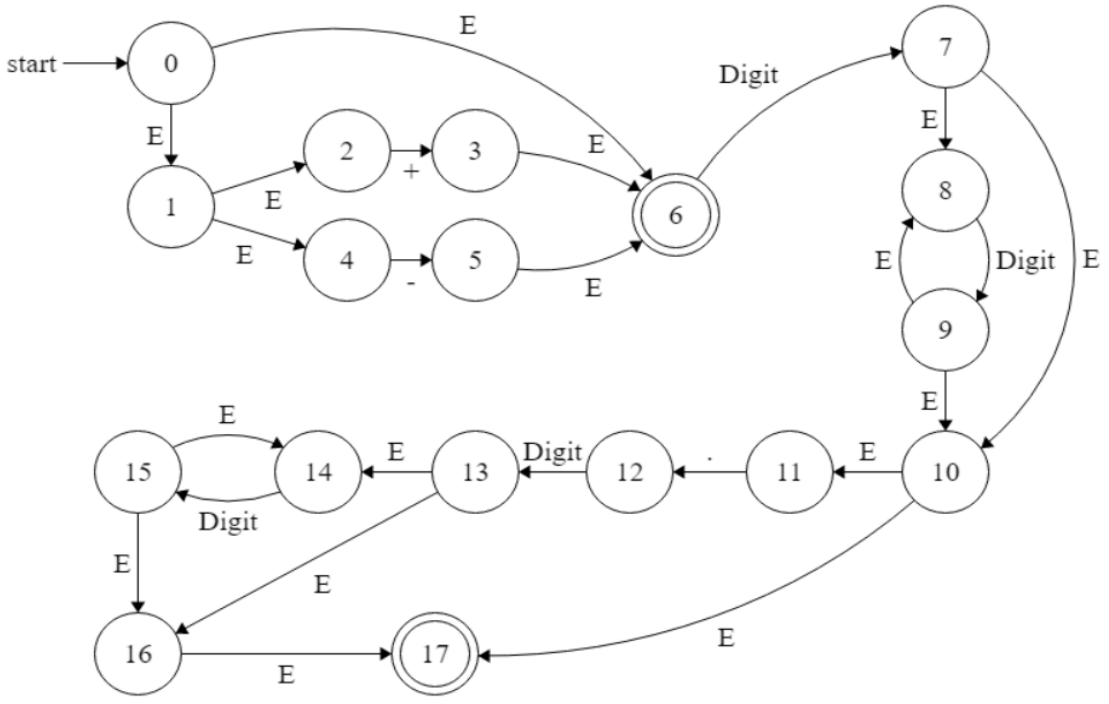

Number,Letter,Op,WhiteSpace
Number
Number,Letter,Op,whitespace
Letter
start
"
OP
Number,letter,op,whitespace
WhiteSpace
Number,letter,op,whitespace

-MIN DFA

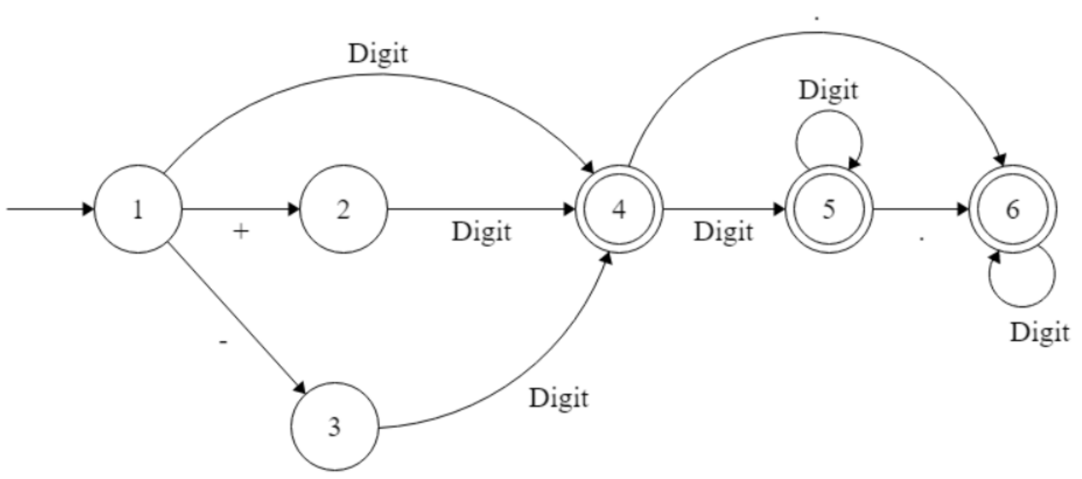


Number (NFA-DFA-MIN DFA):

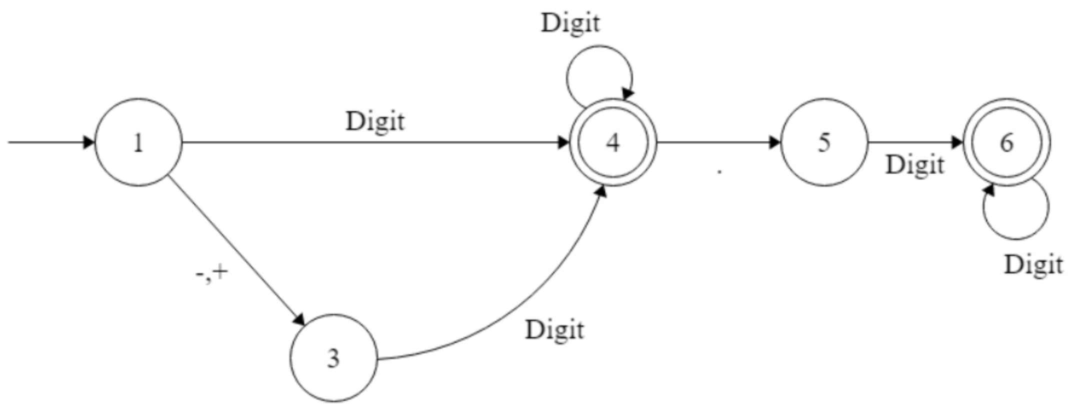
-NFA



-DFA

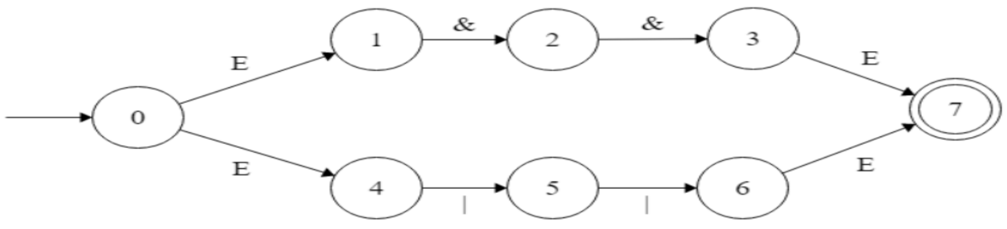


-MIN DFA

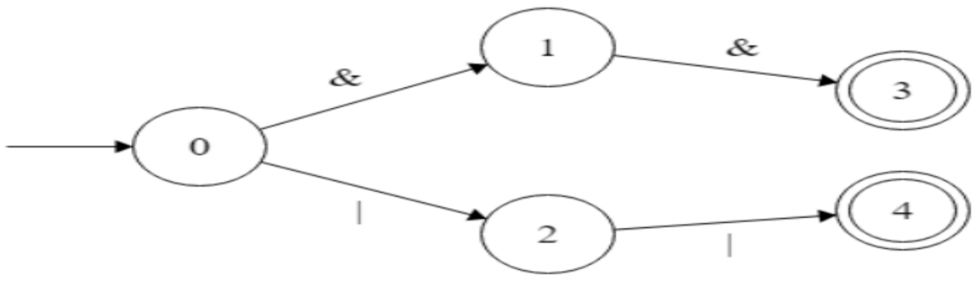


Boolean_Operator (NFA-DFA-MIN DFA):

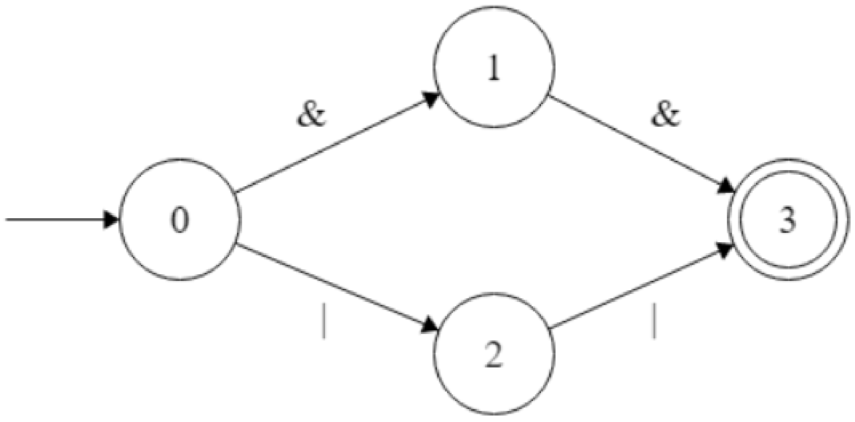
NFA:



-DFA

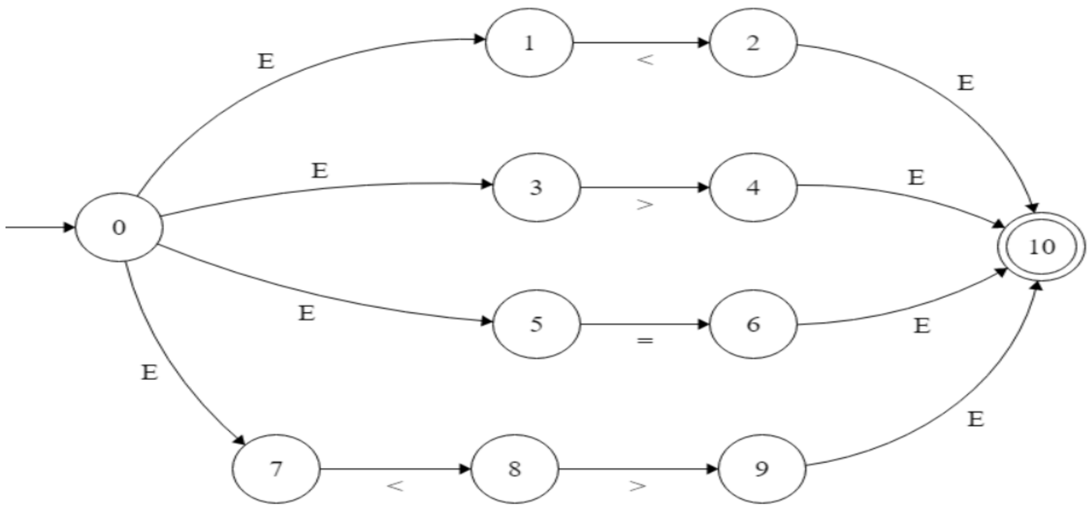


-MIN DFA

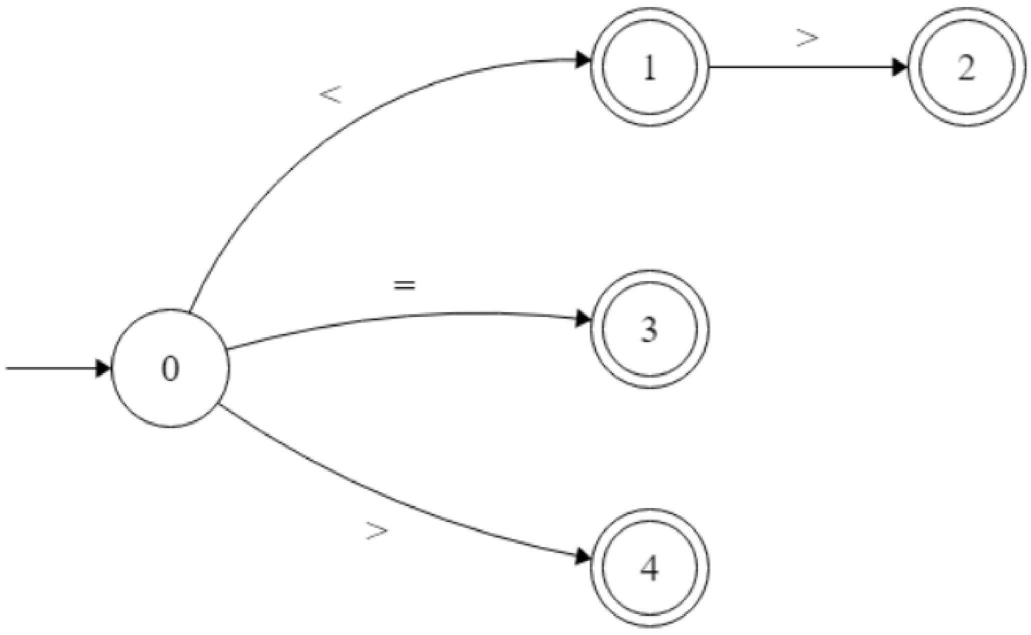


Condition_Operator (NFA-DFA-MIN DFA):

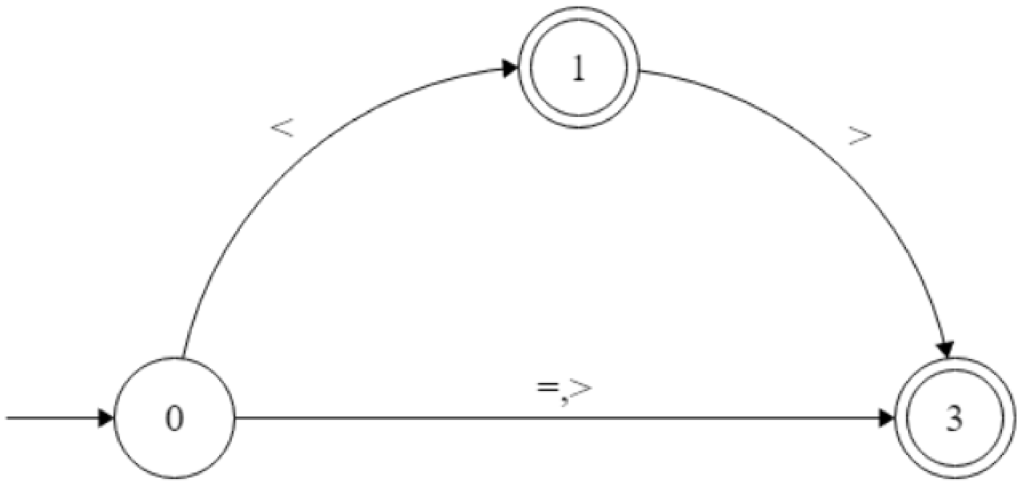
-NFA



-DFA

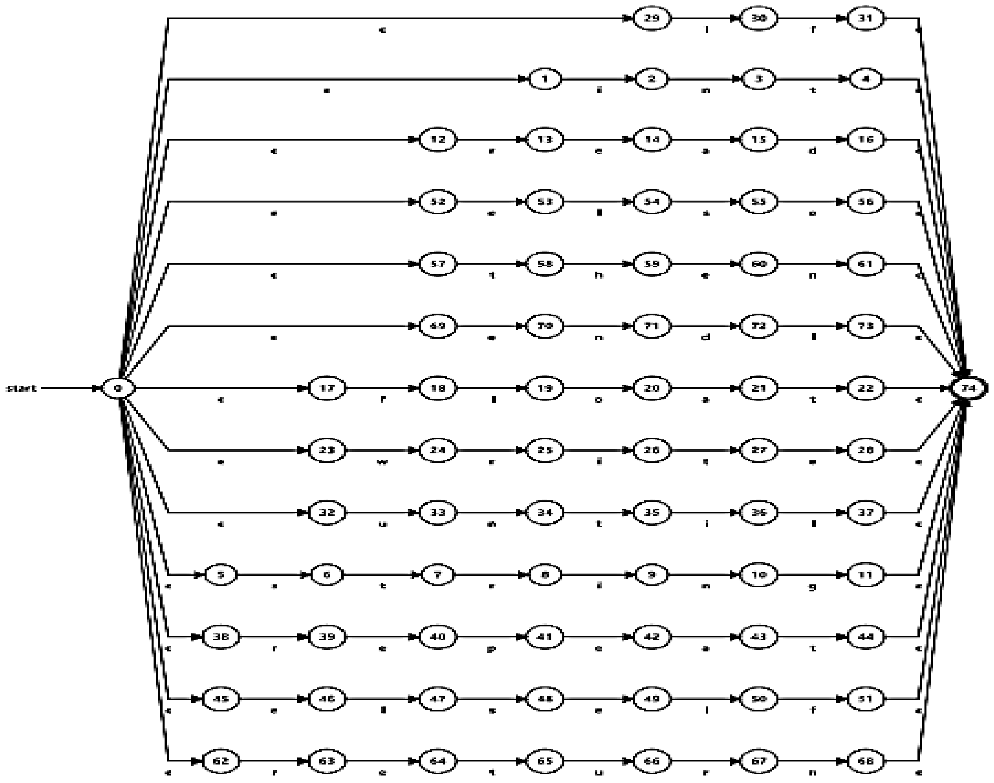


-MIN DFA

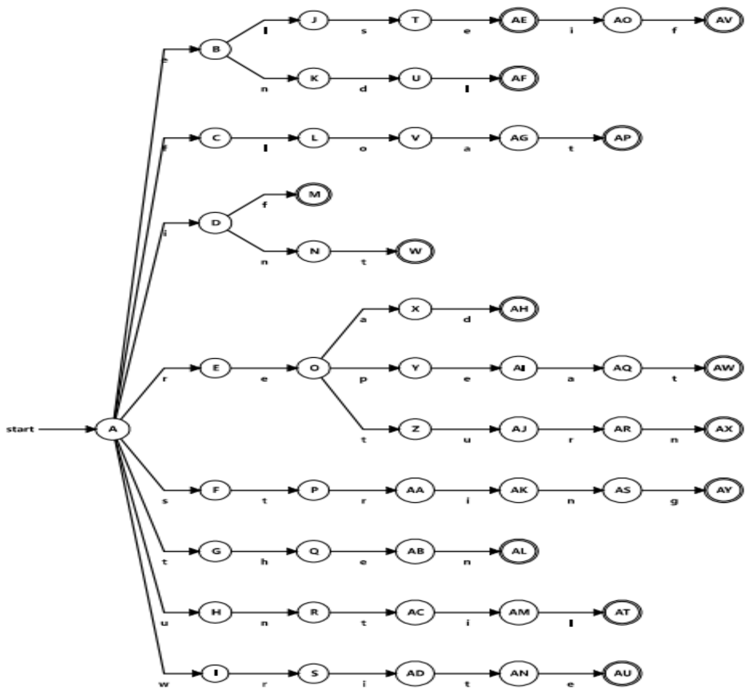


Reserved_Keywords (NFA-DFA-MIN DFA):

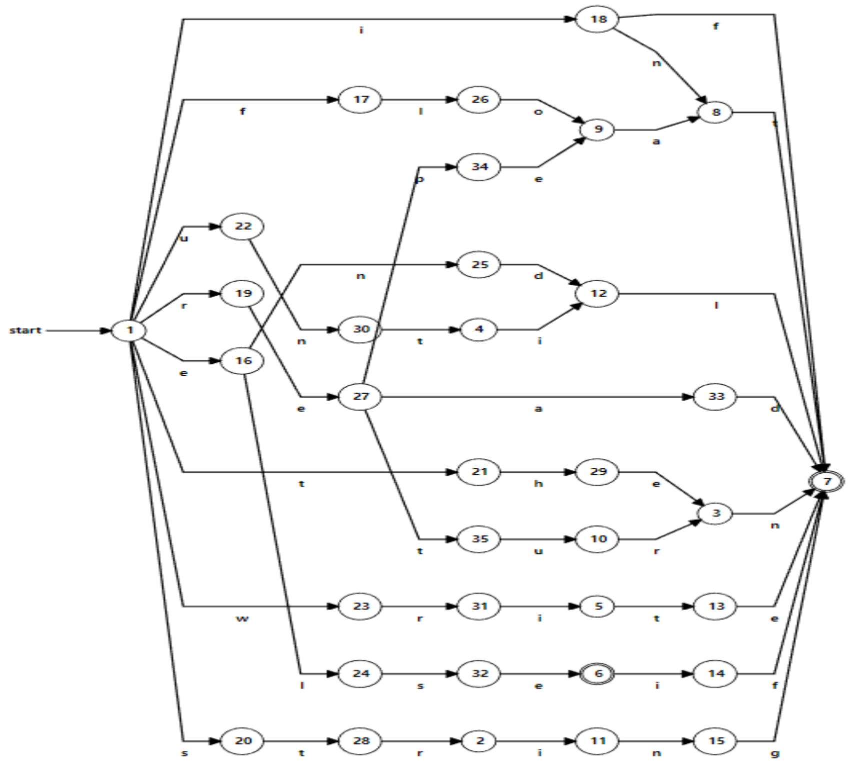
- NFA



-DFA



- MIN DFA



"Thank You"