Lexical Analysis
Programming Languages

Sujit Kumar Chakrabarti

IIITB

Implementation of Lexical Analysis

Finite State Automata

Example

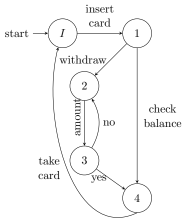
ATM:

Implementation of Lexical Analysis

Finite State Automata

Example

ATM:



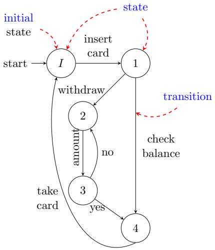
- Alphabet Σ: {insert card, check balance, no, yes, withdraw, amount, take card }
- States, initial state
- Transitions

Implementation of Lexical Analysis

Finite State Automata

Example

ATM:

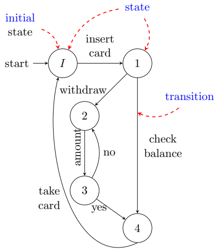


Implementation of Lexical Analysis

Finite State Automata

Example

ATM:



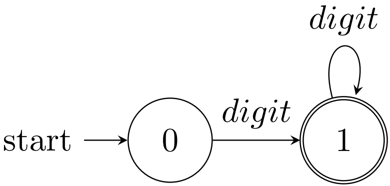
- Alphabet Σ: {insert card, check balance, no, yes, withdraw, amount, take card }
- States, initial state
- Transitions

Implementation of Lexical Analysis

Finite State Automata

Example

NUMBER:

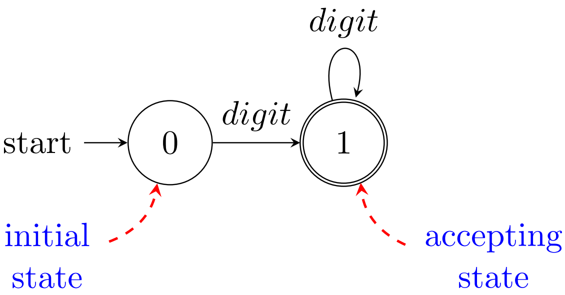


Implementation of Lexical Analysis

Finite State Automata

Example

NUMBER:



Implementation of Lexical Analysis

Finite State Automata as recogniser

A string can be considered accepted if:

- input pointer has reached the end of the string.
- machine is in an accepting state.

Implementation of Lexical Analysis

Finite State Automata as recogniser

A string can be considered accepted if:

- input pointer has reached the end of the string.
- machine is in an accepting state.

A string can be considered rejected if:

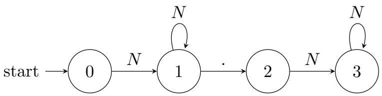
- input pointer has reached the end of the string and machine is not in an accepting state.
- a symbol occurs for which the machine can't make a transition.

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:

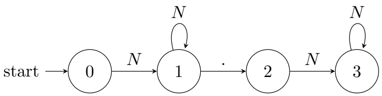


Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



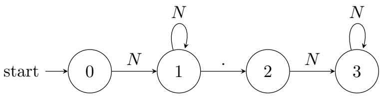
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



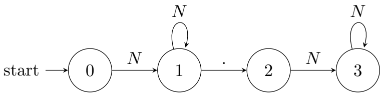
1. $NN.N$ $\qquad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



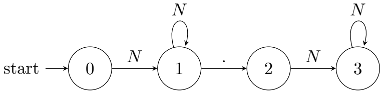
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



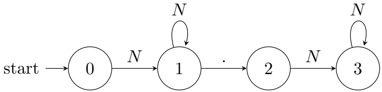
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



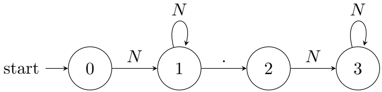
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



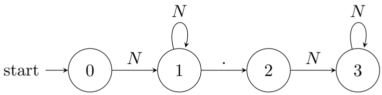
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



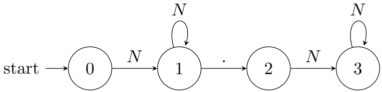
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×
4. $.N$ $\quad 0 \xrightarrow{.}$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



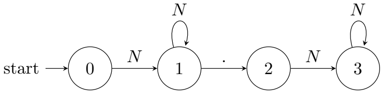
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×
4. $.N$ $\quad 0 \xrightarrow{.}$ ×

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



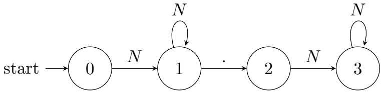
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \; \checkmark$
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3 \; \checkmark$
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \; \times$
4. $.N$ $\quad 0 \xrightarrow{.} \; \times$
5. $N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{\alpha}$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



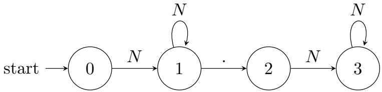
1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×
4. $.N$ $\quad 0 \xrightarrow{.}$ ×
5. $N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{\alpha}$ ×

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:



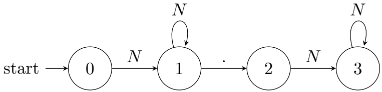
1. $N N.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $N N.N N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $N N.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×
4. $.N$ $\quad 0 \xrightarrow{.}$ ×
5. $N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{\alpha}$ ×
6. $N.N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{\alpha}$

Implementation of Lexical Analysis

Finite State Automata as recogniser

Example

DECIMAL NUMBER:

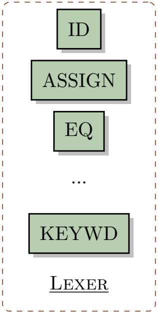


1. $NN.N$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3$ ✓
2. $NN.NN$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{N} 3$ ✓
3. $NN.$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2$ ×
4. $.N$ $\quad 0 \xrightarrow{.}$ ×
5. $N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{\alpha}$ ×
6. $N.N\alpha$ $\quad 0 \xrightarrow{N} 1 \xrightarrow{N} 1 \xrightarrow{.} 2 \xrightarrow{N} 3 \xrightarrow{\alpha}$ ×

Implementation of Lexical Analysis

Finite State Automata as recogniser

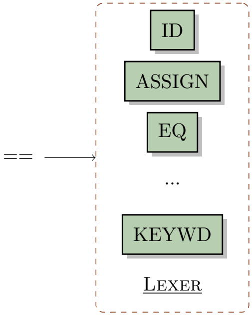
- Each token class T is represented using an FSA F.
- Acceptance of an input string i by F indicates that $i \in T$.
- Lexical analyser consumes i and returns T.



Implementation of Lexical Analysis

Finite State Automata as recogniser

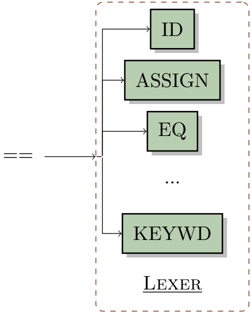
- Each token class T is represented using an FSA F.
- Acceptance of an input string i by F indicates that $i \in T$.
- Lexical analyser consumes i and returns T.



Implementation of Lexical Analysis

Finite State Automata as recogniser

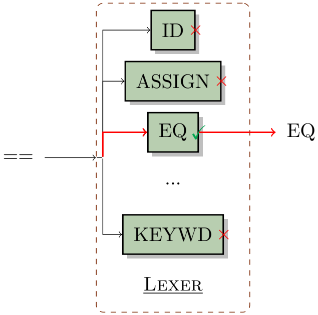
- Each token class T is represented using an FSA F.
- Acceptance of an input string i by F indicates that $i \in T$.
- Lexical analyser consumes i and returns T.



Implementation of Lexical Analysis

Finite State Automata as recogniser

- Each token class T is represented using an FSA F.
- Acceptance of an input string i by F indicates that $i \in T$.
- Lexical analyser consumes i and returns T.



Next

1. DFA and NFA
2. Implementation of FSAs