** **

**DOKUZ EYLUL UNIVERSITY**

**ENGINEERING FACULTY**

**DEPARTMENT OF COMPUTER ENGINEERING**

****

**CME3203 THEORY OF COMPUTATION**

**ASSIGNMENT REPORT**

**By**

**2014510028 Çağrı Anıl Erbey**

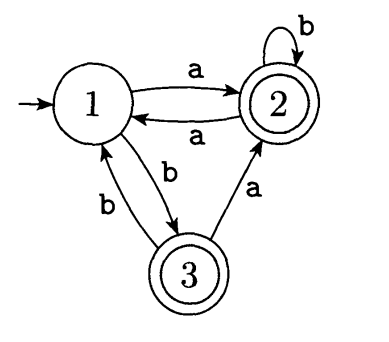
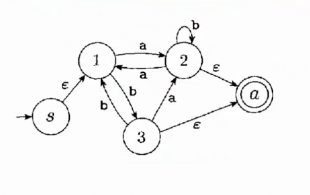
**2013510093 Kemalcan Şimşek**

**January, 2021**

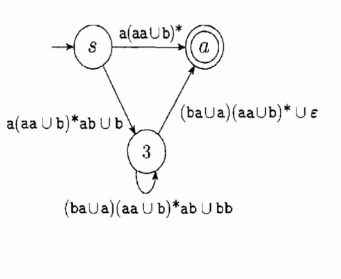
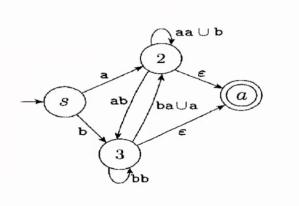
Description

In the theory of computation,a generalized nondeterministic automaton ,also known as generalized nondeterministic finite state machine,is a variation of a nondeterministic finite automaton where each transition is labeled with any regular expression.The GNFA reads blocks of symbols from the input which constitute a string as defined by the regular expression on the transition.There are several differences between a standart finite state machine and a generalized nondeterministic finite state machine.A GNFA must have only one start state and one accept state,and these cannot be the same state,whereas as NFA or DFA both may have several accept states,and the start state can be an accept state.A GNFA must have only one transition between any two states,whereas a NFA or DFA both allows for numerous transitions between states.In a GNFA,a state has a single transition to every state in the macine ,although often it is convention to ignore the transitions that are labelled with the empty set when drawing generalized nondeterministic finite state machines.

GNFA Schema

3 State DFA 5 State GNFA



4 State GNFA 3 State GNFA

//DFA class implementetion

**Psedeu code**

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

public class DFA {

//primary collection of DFA entries

public DFA ( ) {

//Creates DFA from given text datas

}

public DFA( ){

//Creates DFA with null entites

}

public Boolean runDFA(String input) {

//DFA printing method

}

public GNFA converToGNFA ( ){

return GNFA;

}

public String transitionFuncitionToString ( ) {

//converting transition funciton to two dimension array

return transitionFuncitionToString

}

/\*\*

….

Get set methods and returning as orderly string methods.

\*/

}

//GNFA class implementetion

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

public class GNFA {

//GNFA variables

// 5-tuple constructor assuming tuple describes a DFA

public GNFA ( ) {

/\*\*new start and accep state are added, ‘€’ character is added with createNewAlphabetmethod

to alphabet array.\*/ }

public GNFA (DFA dfa) {

//GNFA constructor with given dfa }

public GNFA (DFA dfa) {

//null constructor }

private String[]createNewAlphabet(String[] alphabet){

//adding ‘€’ character to given alphabet }

private Boolean checkForIncomingStartStateTransitions( ){

// Check if we need to create a new start state returns true if we do }

private String[] createStateSetWithStart(String startState, String[] states) {

// the new start and accept states are added to our set of states }

private String[] createStateSetWithoutNewStart(String startState, String[] states){

// do not need to create a new start state so only add a new accept state to our set of states}

private String[][] createNewTransitionFunction(String[][] transitionFunction, String startState){

/\*\* Create the transition function for the GNFA

New transition function needs to create empty transition to new accept state\*/ }

public String createRegex() {

while (statesCopy.size() > 2) {

//Choose qrip as long as it is not the start state and not the accept state

// …

public String[][] convertTransitionFunctionToTransitionTable(

List states, List alphabet, String[][] transitionFunction) {

/\*\* Converts the transition function to a transition table that is index by the states with the

transition symbol as the values\*/ }

//Other print and get methods

…

//Test class to convert and to create.

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

public class dfaTest {

public static void main (String [] args) {

/\* Menu desing

while (true) {

String input = sc.nextLine();

if (input.equals("1")) {//Enter 1 to display the DFA and read strings on the DfA}

else if (input.equals("2")) {//Enter 2 to create and display a GNFA from the DFA}

else if (input.equals("3")) {//Enter 3 to create RE from the GNFA}

else if (input.equals("0")) {break;}

public static DFA createDFAFromFile(String start){

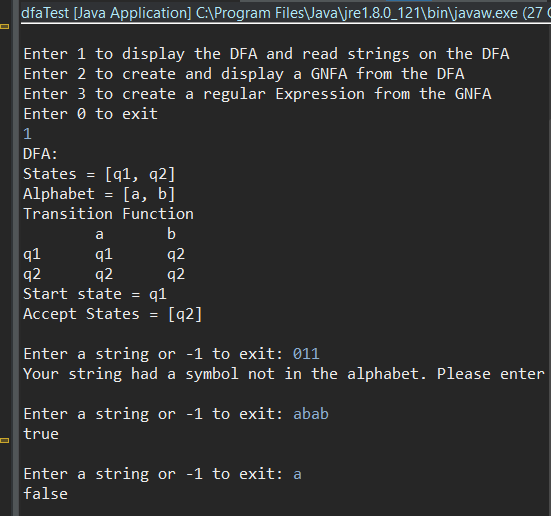
//reading data from text and data parsing

DFA dfa = new DFA(startState, acceptStates , alphabet , states , transitionFunction );

return dfa;

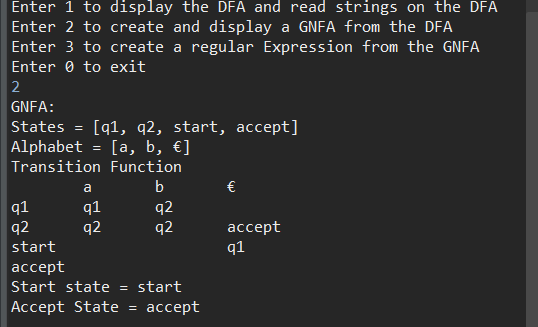
}

}

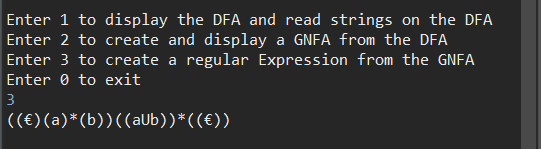


**Sample Screenshots of the Program**

Screenshot 1: Step one; creating DFA and checking string



Screenshot 2: Step two; convert to GNFA from DFA



Screenshot 3: Step three; convert to RE from GNFA