CS 410-PROJECT1 DESIGN REPORT
FALL2022

1. Introduction

In the project, implementing a program which will be converted NFAs to DFAs is aimed. The input will be a text file that contains the information of alphabet, states, start state, final state, and transitions. The output, the DFA version of NFA, will have the same format with input file.

In the section 2, the tools that will be used to design and create this project are introduced. In the section 3, the design of the program is described.

2. Tools Used

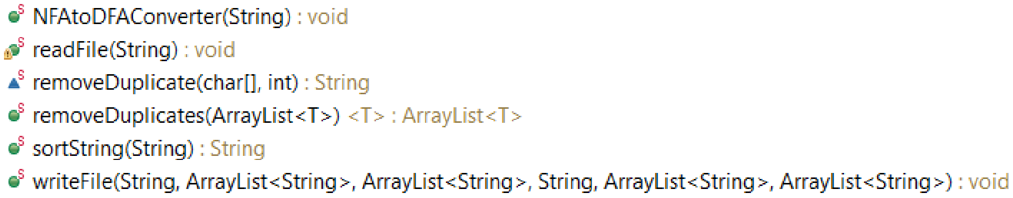
In the project, the tools are used in below.

- Java 8: Used as the software language.
- IntelliJ IDEA Ultimate 2019: Used as the Integrated Development Environment.
- Unified Modeling Language (UML): Used in creating diagram.
- Java Util Package: Used to create ArrayList objects to manipulate converting NFA to DFA.
- Java Io Package: Used to read and write files.
- Creatly: Used website to create diagram.

3. Software Design

In this section, algorithm is explained and activity diagram is introduced.

The algorithm contains functions to make the algorithm simplified. The list of the functions are below:



```
NFAtoDFAConverter(String) : void
readFile(String) : void
removeDuplicate(char[], int) : String
removeDuplicates(ArrayList<T>) <T> : ArrayList<T>
sortString(String) : String
writeFile(String, ArrayList<String>, ArrayList<String>, String, ArrayList<String>, ArrayList<String>) : void
```

Figure 1: List of functions of the NFA to DFA converter

NFAtoDFAConverter: It is the main function that is called to convert NFA to DFA.
readFile: It is used to read NFA file.
removeDuplicate: It is used to remove duplicated chars that exists on the string.
removeDuplicates: It is used to remove duplicated list items that exists on the ArrayList.
sortString: It is used to sort given string.
writeFile: It is used to create and write DFA file.

The algorithm start with reading the NFA file and add the Arraylists and strings necessarily. After that, the transition heads and tails obtained from the ArrayLists. Then, the while loops start and continues length of the states. In the while loop, firstly every possible state and alphabet check to unite the states started with same state and transition. The united states are also added to state ArrayList and checked with other states. After the step, the transition tails and heads are updated, also transition differences between heads and tails will be checked. If a state exists tails but not heads it means that the transition must create to convert DFA. In DFA, every states have to go with every possible alphabet. Then, other while loop starts to remove transition differences.

Finally, after converting successfully, writeFile function runs and DFA(DFAnumber).txt file is created.

```
ALPHABET  //alphabet of the DFA, similar with alphabet of given NFA.
0
1
STATES  //states of the DFA, added new states to the states of given NFA.
A
B
C
BC
AB
ABC
START  //start state of the DFA, similar with start state of given NFA.
A
FINAL  //final state of the DFA, added the new states which contains the old final states.
C
BC
ABC
TRANSITIONS //transitions of the DFA, added the new transitions contained the old final states.
BC 0 ABC
BC 1 B
AB 0 ABC
AB 1 BC
ABC 0 ABC
ABC 1 B
A 0 A
A 1 BC
B 0 BC
C 0 AB
C 1 B
B 1 Ø  // Ø means that the state B goes to dead/ sink state.
END
```

Example 1: Example of converted DFA

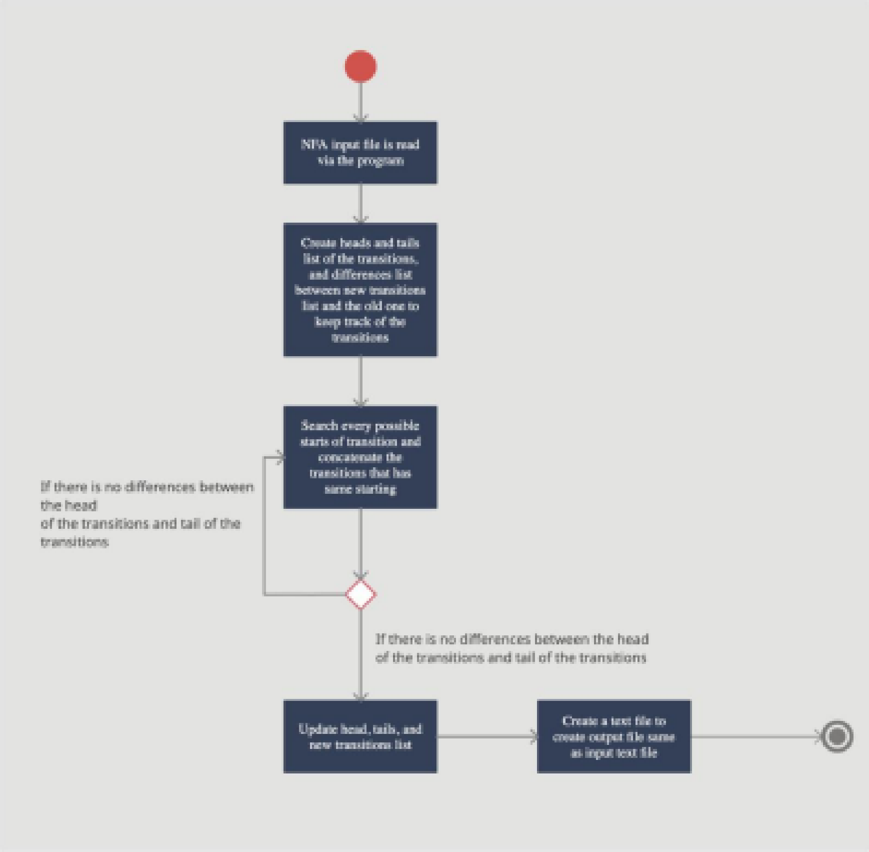

NFA input file is read via the program
Create heads and tails list of the transitions, and differences list between new transitions list and the old one to keep track of the transitions
Search every possible starts of transition and concatenate the transitions that has same starting
If there is no differences between the head of the transitions and tail of the transitions
If there is no differences between the head of the transitions and tail of the transitions
Update head, tails, and new transitions list
Create a text file to create output file same as input text file

Figure 2: Activity diagram of the program