Lehrstuhl für Softwaretechnik und Programmiersprachen
Professor Dr. Michael Leuschel

Compilerbau - Wintersemester 2021/22

Praktisches Übungsblatt 3

Besprechung der Aufgaben am 12.11.21 um 16:30 Uhr in 25.12.02.55 und gleichzeitig online per BBB
Fragen an Lukas.Lang@hhu.de
Die Bearbeitung ist freiwillig.
Bringen Sie Ihre Lösung mit zum Übungstermin

Aufgabe 3.1

Implementieren Sie in Java einen nichtdeterministischen endlichen Automaten. Sie erhalten einen JUnit Testcase für zwei Automaten, die den Ausdruck $(a|b)^*abb$ kodieren.

1. Implementieren Sie dazu eine Klasse Node, die einen Zustand des Automaten repräsentiert und das folgende Interface implementiert:

```
import java.util.Set;

public interface INode {
    public void addTransition(INode dest, char c);
    public Set<INode> getNext(char c);
    public String getName();
    public boolean isFinal();
    public Set<IEdge> getEdges();
    public Set<INode> epsilonClosure();
}
```

   Beachten Sie, dass die Methode getNext(char c) null zurückliefern soll genau dann, wenn es keine Transition für c gibt.

2. Schreiben Sie eine Klasse NFA, in der der Automat kodiert ist, sowie eine Klasse Edge welche die Kanten modelliert.

```
public interface IEdge {
    public INode getStart();
    public INode getEnd();
    public char getChar();
}
```

   Die Kodierung des Automaten soll im Konstruktor der Klasse erfolgen. Dem Konstruktor der Klasse wird hierzu eine Menge von Knoten und Kanten sowie ein Startknoten übergeben.

```
import java.util.Set;
import java.util.HashSet;
import java.util.ArrayList;
import java.util.List;

public class NFA {
    private Set<Character> alphabet;
    private Set<INode> nodes;
    private Set<IEdge> edges;
    private INode start;


    public NFA(Set<Character> alphabet, Set<INode> nodes,
         INode start, Set<IEdge> edges)
    {
        // TODO
    }

    public Set<List<INode>> allPaths(String word) {
        // TODO
    }

    String accept(String word) {
        // TODO
    }

}
```

3. Die Klasse soll eine Methode accept(String word) haben. In dieser Methode werden der Reihe nach die besuchten Knoten ausgegeben und anschließend entweder ablehnen oder akzeptieren. Die gesamte Ausgabe muss in einer Zeile stehen und zwischen der Knotenliste und dem Ergebnis muss ein Leerzeichen sein.
4. Schreiben Sie eine Methode *public DFA toDFA()* welche den NFA in einen DFA konvertiert. Erhalten Sie für beide Automaten im Testcase den selben DFA?