Prof. T. Nipkow
K. Kappelmann, J. Rädle, L. Stevens **ABGABE 11.05.2020, 23:59 (Σ 4 P.)** Technische Universität München
Lehrstuhl für Logik und Verifikation

Einführung in die Theoretische Informatik

Sommersemester 2020 – Übungsblatt 2

Ablauf Hausaufgabenabgabe

Für die Korrektur Ihrer Programmierabgaben wird TUMjudge(https://judge.in.tum.de/theo/public/) verwendet. Details finden Sie auf den Folien des Praktikums Algorithms for Programming Contests von 2017(https://www7.in.tum.de/um/courses/theo/ss2018/materials/judge.pdf).

AUFGABE 2.1. (*Automata Tutor*) 1,5 Punkte

Im Automata Tutor finden Sie wieder bepunktete Hausaufgaben. Beachten Sie, dass Sie für diese Aufgaben weiterhin nur fünf Versuche haben. Die Aufgaben finden Sie unter den Kategorien "English to Regular Expression" und "Regular Expression to NFA". Beachten Sie, dass der AutomataTutor automatisch $c \mid \varepsilon$ als $c?$ abkürzt.

AUFGABE 2.2. (*Suffixsprache auf regulären Ausdrücken*) 1 Punkt

In dieser Aufgabe implementieren Sie eine rekursive Prozedur auf regulären Ausdrücken.

- Lesen Sie sich die PDF-Angabe zu der Aufgabe 'SuffixLanguage' durch (zu finden unter 'course/problems' auf TUMjudge). **Achtung**: Lassen Sie sich nicht von den automatisch angehängten Ein-/Ausgabepaaren verwirren, da dort Unicode-Zeichen fehlen. Das Archiv für das Codegerüst enthält die richtigen Paare.
- Laden Sie das Codegerüst für die Hausaufgabe auf Moodle herunter.
- Wir stellen ein Haskell Gerüst bereit, in dem reguläre Ausdrücke mit dem Datentypen `Regex` modelliert sind. Implementieren Sie die Funktionen `isEmpty` und `suffix`.
- Wenn Sie stattdessen Java verwenden möchten, betrachten Sie die abstrakte Basisklasse `Regex` und ihre konkreten Kindklassen. Sie sollten die Attribute, insbesondere die dafür gewählten Datentypen, verstehen, bevor Sie fortfahren.
- Implementieren Sie alle mit `TODO` markierten Methoden in den Kindklassen von `Regex`, bzw. die undefinierten Funktionen im Haskell Template.
- Laden Sie dann für Problem A (RegexOperations) alle benötigten Dateien hoch – falls Sie eines unserer Templates benutzen, müssen Sie *alle* Dateien des Templates hochladen, nicht nur die, die Sie verändert haben.

AUFGABE 2.3. (*Von DFA zu regulärem Ausdruck*) 1,5 Punkte

In dieser Aufgabe implementieren sie eine Konvertierung von DFAs zu regulären Ausdrücken.

- Lesen Sie sich die PDF-Angabe zu der Aufgabe 'DFAToRegex' durch (zu finden unter 'course/problems' auf TUMjudge).
- Laden Sie das Codegerüst auf Moodle herunter.
- Wir stellen ein Haskell Gerüst bereit, in dem reguläre Ausdrücke mit dem Datentypen `Regex` modelliert sind. Implementieren Sie die Funktionen `accepts` und `dfAToRegex`.
- Wenn Sie stattdessen Java verwenden möchten, betrachten Sie die abstrakte Basisklasse `Regex` und ihre konkreten Kindklassen.
- Implementieren Sie alle mit `TODO` markierten Methoden in der Klasse `DFAOperations`, bzw. die undefinierten Funktionen in der Datei `DFAOps.hs`.
- Laden Sie dann für Problem B (DFAToRegex) alle benötigten Dateien hoch – falls Sie eines unserer Templates benutzen, müssen Sie *alle* Dateien des Templates hochladen, nicht nur die, die Sie verändert haben.