RWTH Aachen **SS 2015**
Lehrgebiet Theoretische Informatik Blatt 3
Rossmanith–Reidl–Sánchez 30.4.2015

Algorithmen für endliche Automaten und kontextfreie Grammatiken

Aufgabe 9

Schreiben Sie eine Methode, um DFAs zu minimieren. Dies ist die letzte grundlegende Funktionalität, die uns bei den Automaten noch fehlt.

Mithilfe dieser und einer weiterer Methode, die feststellt, ob zwei DFAs isomorph sind, können Sie nun auch leicht ein Verfahren programmieren, welches die Sprachäquivalenz zweier Automaten testet. Das heißt, festzustellen, ob zwei Automaten dieselbe Sprache akzeptieren.

Insgesamt, wollen wir folgende Fragen zu einem Automaten M beantworten können:

- Das Leerheitsproblem: $L(M) = \emptyset$?
- Universalität: $L(M) = \Sigma^*$?
- Endlichkeit: $|L(M)| < \infty$?

Und folgende Fragen zu zwei Automaten M_1 und M_2:

- Sprachäquivalenz: $L(M_1) = L(M_2)$?
- $L(M_1) \cap L(M_2) = \emptyset$?
- $L(M_1) \subseteq L(M_2)$?

Es könnte nützlich sein, solche Fragen nicht einfach nur mit ja oder nein zu beantworten, sondern gegebenenfalls ein Gegenbeispiel anzugeben. Gilt, zum Beispiel, $L(M) \neq \emptyset$, dann wäre es nützlich ein Wort $w \in L(M)$ zu erhalten, welches diese Tatsache beweist.

Aufgabe 10

Implementieren Sie eine Klasse für reguläre Ausdrücke. Wir brauchen hier Konstruktoren für die primitiven regulären Ausdrücke, nämlich die leere Sprache, die Sprache, welche nur das leere Wort enthält, und die Sprachen, welche jeweils ein Wort enthalten, das nur aus einem Symbol besteht.

Dann wäre noch Möglichkeit nützlich, aus einem String, der einen regulären Ausdruck beschreibt, einen regulären Ausdruck zu erzeugen. Solch ein String könnte zum Beispiel so aussehen: `(ab)*a+(ba)*`. In diesem String haben die Klammern, der Stern und das + eine Sonderbedeuteung und das Alphabet ist $\{a, b\}$. Es sollte erlaubt sein, beliebige Symbole für die notwendigen Metasymbole (anstatt `(`, `)`, `*` und `+`) festlegen zu dürfen (welche dann nur nicht im zugehörigen Alphabet enthalten sein dürfen).

Nun implementieren Sie auch noch Methoden, um neue reguläre Ausdrücke durch Konkatenation, Vereinigung und Kleene'sche Hülle bilden zu können.

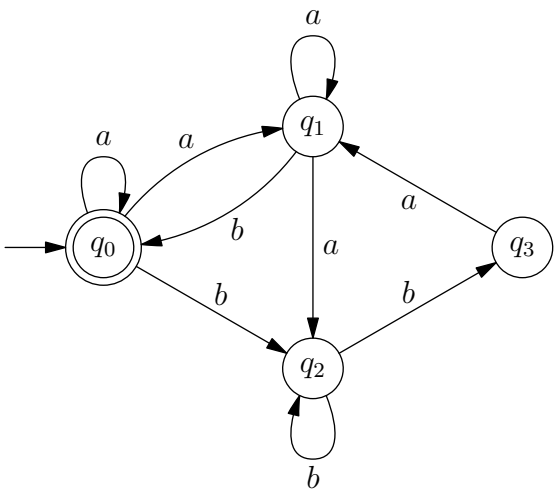
Aufgabe 11

Um die Brücke zwischen regulären Ausdrücken und Automaten zu schlagen, implementieren Sie Methoden, die aus einem regulären Ausdruck einen NFA mit ϵ-Kanten erzeugen, und umgekehrt.

Wie gliedern sich reguläre Ausdrücke in Ihr bestehendes Klassengefüge ein?

Es ist unangenehm, daß reguläre Ausdrücke, die aus einem Automaten erzeugt werden, oft sehr groß werden. Gehen Sie auf dieses Problem ein und versuchen Sie, möglichst kleine reguläre Ausdrücke zu erzeugen. Beispielsweise können reguläre Ausdrücke manchmal vereinfacht werden. Der Ausdruck $((a + b)^* + \epsilon)^*$ kann leicht durch $(a + b)^*$ ersetzt werden. Das ϵ ist nämlich unnötig und nach seiner Entfernung ist es unsinnig zweimal die Kleene'sche Hülle hintereinander anzuwenden.

Welchen regulären Ausdruck erhalten Sie aus diesem NFA:



Welchen regulären Ausdruck erhalten Sie, wenn Sie den Ausdruck $(aabb)^*aa$ mithilfe Ihrer Programme in einen NFA verwandeln und aus diesen wieder einen regulären Ausdruck erzeugen?