

Übungsaufgabe II

1) Sehen Sie sich folgenden regulären Ausdruck an: $[?+ i g]$.

- Beschreiben Sie informell die Sprache, die dieser Ausdruck denotiert. Das „?“ steht hier für *any*. (1 Punkt)
- Lesen Sie diesen Ausdruck mit dem Befehl *read regex* in *xfst* ein. Schauen Sie sich die Übergangstabelle mit dem Befehl *print net* an. Leiten Sie nun die Zeichenreihe „schaurig“ mit der Übergangsfunktion δ^* für einen DEA ab. (3 Punkte)

Ein Beispiel dazu finden Sie in Hopcroft et al. (2001:2.2.4) oder in den Vorlesungsnotizen.

2) Sehen Sie sich folgende Übergangstabelle an.

	a	b	c	d	e	f	g
$\rightarrow s_0$	{s1}	{s2}	{s2}	{s3}	\emptyset	\emptyset	\emptyset
s1	\emptyset	{s2}	{s2}	{s3}	\emptyset	\emptyset	\emptyset
s2	\emptyset	\emptyset	{s2}	{s3}	\emptyset	\emptyset	\emptyset
s3	\emptyset	\emptyset	\emptyset	\emptyset	{s4}	{s5}	\emptyset
*s4	\emptyset						
s5	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	\emptyset	{s0}

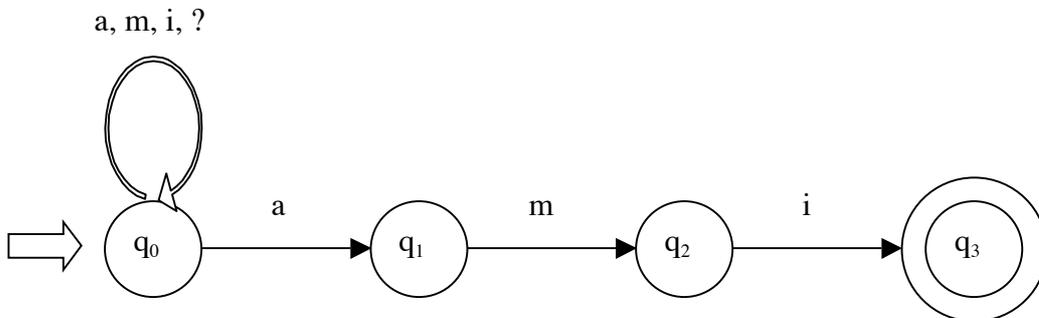
Der Startzustand ist durch „ \rightarrow “ gekennzeichnet, der Endzustand durch „*“.

- Von welchem Typ ist der dazugehörige Automat? Warum? (1 Punkt)
- Beschreiben Sie informell, welche Sprache er definiert. (1 Punkt)
- Definieren Sie den Automaten. Die Funktion δ brauchen Sie dabei nicht noch einmal explizit anzugeben.
- Zeichnen Sie den Automaten. (1 Punkt)
- Geben Sie die Sprache mit einem regulären Ausdruck wieder (nach der Notation von Beesley/Karttunen 2003). (2 Punkte)
- Geben Sie den regulären Ausdruck in *xfst* ein und überprüfen Sie seine Korrektheit anhand von Beispielen mit den Befehlen *apply up* oder *apply down*. Die folgenden Sequenzen sollten akzeptiert werden: de; ade; abccccce; abcde; abcdfgabcde; abcdfgabcdfgabcde.
Die folgenden Sequenzen sollten nicht akzeptiert werden: ab; e; abcdfg. (2 Punkte)
- Geben Sie die Sprache mit einer regulären Grammatik in *lexc* wieder. Benutzen Sie dabei beliebige Ausdrücke für die Kategorien. (2 Punkte)
- Erstellen Sie ein *xfst*-Skript. Lesen Sie in diesem Skript ihre *lexc*-Datei ein. Führen Sie anschließend in dem Skript folgende Ersetzungen mit Regeln durch:

a = all+
 b = the+
 c = old+, young+, small+, tall+
 d = men+, students+
 e = listened, slept
 f = say+, think+, believe+
 g = that+

Verbinden Sie nun das Lexikon mit den Regeln durch .o.
 Erstellen Sie eine Datei mit 5 erlaubten und 2 unerlaubten Kombinationen.
 Lesen Sie diese in Ihrem Skript mit *apply up < testdatei* oder *apply down < testdatei* ein. (5 Punkte)

3) Sehen Sie sich folgenden Automaten an.



- Warum ist dieser Automat ein NFSA? (1 Punkt)
- Beschreiben Sie informell, welche reguläre Sprache dieser Automat denotiert. Das „?“ wird hier als *unknown* interpretiert (Beesley/Karttunen 2003:2.3.4). (1 Punkt)
- Geben Sie die Definition dieses Automaten an. Geben Sie dabei die Funktion δ dabei mit einer Übergangstabelle an. (4 Punkte)
- Wandeln Sie diesen NFSA in einen DFSA durch Teilmengenbildung um. Gehen Sie dabei analog zum Beispiel aus der Vorlesung vor: Erstellen Sie eine erweiterte Übergangstabelle, benennen Sie die Teilmengen um, lassen Sie die unerreichbaren Zustände weg und erstellen Sie eine verkleinerte Übergangstabelle für den DFSA. (5 Punkte)
- Zeichnen Sie den DFSA. (1 Punkt)
- Geben Sie an, wieviele Zustände und Übergänge jeweils der NFSA und der DFSA haben. (1 Punkt)

Das Verfahren zur Umwandlung eines NFSA in einen DFSA wird auch in Hopcroft et al. (2001:2.3.5) beschrieben.

Abzugeben bis zur nächsten Sitzung am 15.05.2007.

Vorbereitung zu nächsten Sitzung:

- a) Jurafsky/Martin 2000:3.3.2-3 – *Finite-State Morphological Parsing* und *Combining FST Lexicon and Rules*
- b) Karttunen (1991:1-2.2) – reguläre Relationen und phonologische Regeln
- c) Karttunen (2001) – Überblick über FST
- d) Roche/Schabes 1997:I – Definition von Transducern
- e) Beesley/Karttunen 2003:II-III
- f) *Kaplan/Kay 1994