Übungen 1: Endliche Automaten

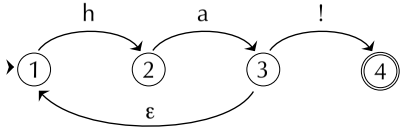
Programmiertechniken in der Computerlinguistik II · Sommersemester 2005

1. Ziemlich leere Sprachen

Wie sieht ein Deterministischer Endlicher Automat aus, der genau die Sprache {ε} bzw. {} akzeptiert? Zeichne ensprechende Zustandsübergangsdiagramme!

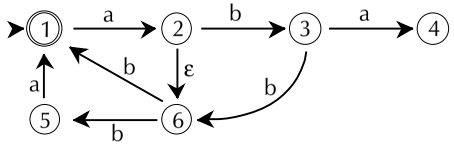
2. Akzeptor für Nicht-Deterministischen Lachautomaten

Adaptiere die Übergangsregeln delta/3 und das Prädikat accept/2 aus der Vorlesung so, dass folgender Nicht-Deterministischer Automat implementiert wird:



3. Nicht-Deterministische Endliche Automaten

Gegeben sei folgender Nicht-Deterministischer Endliche Automat.

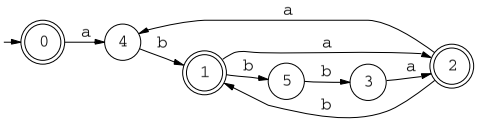


a) Welche der untenstehenden Eingabeketten werden von ihm akzeptiert?

1) aa 2) aba 3) abb 4) ab 5) abab 6) ε

b) Implementiere den Automaten in Prolog und überprüfe deine Akzeptabilitätsurteile!

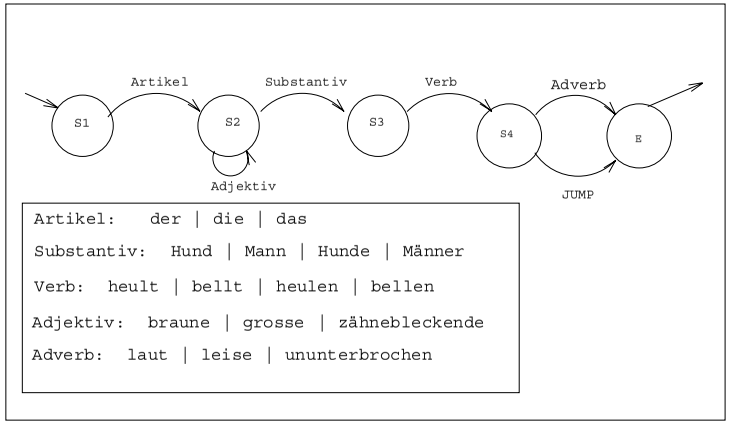
c) Erkennt der folgende deterministische Automat die gleiche Sprache wie obiger nicht-deterministische? Wenn nein, was müsste noch geändert werden?



d) Schreibe einen Regulären Ausdruck, der die gleiche Sprache wie dieser Automat erkennt.

4. Endliche Übergangsnetzwerke

Aus dem Abschnitt "Endliche Automaten und Endliche Übergangsnetzwerke" von ECL I erkennst du sicher die folgende Abbildung mit dem Endlichen Übergangsnetzwerk wieder:



Implementiere diesen Automaten. Überlege dir, wie du delta/3-Klauseln anpasst, damit die Übersichtlichkeit der Endlichen Übergangsnetzwerke erhalten bleibt. Dein Automat sollte dann folgendes Verhalten zeigen:

```
?- init([der,'Mann',bellt]).
yes

?- init(['Hunde',die,heulen]).
no
```