



Informatik III

Übung 3

Silke Wagner

Minimierung von Automaten

● Wozu?

Minimierung von Automaten

● Wozu?

Beantwortung der Frage, ob 2 Beschreibungen regulärer Sprachen dieselbe Sprache repräsentieren (bzw. ob 2 DEAs dieselbe Sprache akzeptieren)

Minimierung von Automaten

- Wozu?

Beantwortung der Frage, ob 2 Beschreibungen regulärer Sprachen dieselbe Sprache repräsentieren (bzw. ob 2 DEAs dieselbe Sprache akzeptieren)

- Wie?

Minimierung von Automaten

- Wozu?

Beantwortung der Frage, ob 2 Beschreibungen regulärer Sprachen dieselbe Sprache repräsentieren (bzw. ob 2 DEAs dieselbe Sprache akzeptieren)

- Wie?

Zusammenfassung äquivalenter Zustände

Minimierung von Automaten

- Wann sind 2 Zustände $p, q \in Q$ eines Automaten $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ äquivalent?

Minimierung von Automaten

- Wann sind 2 Zustände $p, q \in Q$ eines Automaten $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ äquivalent?

Wenn sie bzgl. ihres Akzeptanz-Verhaltens nicht zu unterscheiden sind, d.h.: Die Abarbeitung eines bel. Wortes $w \in \Sigma^*$ aus p heraus endet genau dann in einem (Nicht-)Endzustand, wenn sie es auch aus q heraus tut.

Minimierung von Automaten

- Wann sind 2 Zustände $p, q \in Q$ eines Automaten $\mathcal{A} = (Q, \Sigma, \delta, s, F)$ äquivalent?

Wenn sie bzgl. ihres Akzeptanz-Verhaltens nicht zu unterscheiden sind, d.h.: Die Abarbeitung eines bel. Wortes $w \in \Sigma^*$ aus p heraus endet genau dann in einem (Nicht-)Endzustand, wenn sie es auch aus q heraus tut.

Formal:

$$\delta(p, w) \in F \Leftrightarrow \delta(q, w) \in F$$

Minimierung von Automaten

- Beachte:
es wird nicht verlangt, dass es sich bei $\delta(p, w)$ und $\delta(q, w)$ um dieselben Zustände handelt, sondern nur, dass entweder beide oder keiner von beiden Endzustände sind.

Minimierung von Automaten - Vorgehen

- Entfernung überflüssiger Zustände (Tiefensuche)
- sukzessives Aufspalten der Zustandsmenge mittels Zeugen für Nichtäquivalenz:
 - ε ist Zeuge für Nichtäquivalenz von Zuständen aus F mit Zuständen aus $Q \setminus F$
 - anschl. werden für $n = 0, 1, \dots$ alle Wörter der Länge n betrachtet:
 - ist ein Wort Zeuge für Nichtäquivalenz, so wird die entsprechende Menge von Zuständen weiter aufgespalten
 - ist für ein $n \in \mathbb{N}$ KEIN Wort der Länge n Zeuge für Nichtäquivalenz, so kann das Verfahren abgebrochen werden

Minimierung von Automaten - Vorgehen

- die entstandenen Mengen von Zuständen entsprechen den Zuständen des Äquivalenzklassen-Automaten
- da alle Zustände in einer Ä-Klasse das gleiche "Verhalten" haben, lassen sich die Übergänge aus dem ursprünglichen Automaten "übertragen"

Pumping-Lemma

Sei L eine reguläre Sprache und \mathcal{A} ein DEA mit n Zuständen, der sie erkennt. Sei $w \in L$ ein Wort mit mehr als $n - 1$ Buchstaben.

Idee des Pumping-Lemmas:

- bei der Abarbeitung von w im DEA wird min. ein Zustand zweimal erreicht (da der DEA sonst $|w| + 1 > n$ Zustände haben müsste)

Pumping-Lemma

Sei L eine reguläre Sprache und \mathcal{A} ein DEA mit n Zuständen, der sie erkennt. Sei $w \in L$ ein Wort mit mehr als $n - 1$ Buchstaben.

Idee des Pumping-Lemmas:

- bei der Abarbeitung von w im DEA wird min. ein Zustand zweimal erreicht (da der DEA sonst $|w| + 1 > n$ Zustände haben müsste)
- es wird also eine Schleife gelaufen

Pumping-Lemma

Sei L eine reguläre Sprache und \mathcal{A} ein DEA mit n Zuständen, der sie erkennt. Sei $w \in L$ ein Wort mit mehr als $n - 1$ Buchstaben.

Idee des Pumping-Lemmas:

- bei der Abarbeitung von w im DEA wird min. ein Zustand zweimal erreicht (da der DEA sonst $|w| + 1 > n$ Zustände haben müsste)
- es wird also eine Schleife gelaufen
- diese Schleife kann man beliebig oft laufen, bevor man das restliche Wort abarbeitet und in einem Endzustand landet

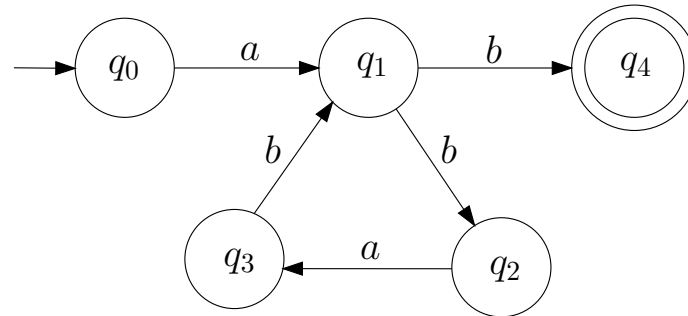
Pumping-Lemma

Sei L eine reguläre Sprache und \mathcal{A} ein DEA mit n Zuständen, der sie erkennt. Sei $w \in L$ ein Wort mit mehr als $n - 1$ Buchstaben.

Idee des Pumping-Lemmas:

- bei der Abarbeitung von w im DEA wird min. ein Zustand zweimal erreicht (da der DEA sonst $|w| + 1 > n$ Zustände haben müsste)
- es wird also eine Schleife gelaufen
- diese Schleife kann man beliebig oft laufen, bevor man das restliche Wort abarbeitet und in einem Endzustand landet
- das wiederholte Laufen der Schleife entspricht einem Aufblasen ("pumping") von w an der entsprechenden Stelle

Pumping-Lemma: Beispiel



$w = ababb$ hat mehr als $n - 1 = 4$ Buchstaben \Rightarrow bei der Abarbeitung muss eine Schleife gelaufen werden ($q_1 \rightarrow q_2 \rightarrow q_3$); diese kann beliebig oft gelaufen werden, bevor man zum Endzustand q_4 "weiterläuft"

\rightsquigarrow Wortteil bab kann bel. oft wiederholt (oder ganz weggelassen) werden und das entstehende Wort $w' = a(bab)^i b$ ($i \in \mathbb{N}_0$) ist auch in L .