Fachhochschule Dortmund
University of Applied Sciences and Arts
we focus on students
Prof. Dr. Robert Preis

Kapitel 2

Reguläre Sprachen

2.2

Minimierung und Äquivalenz von DEAs

Prof. Dr. Robert Preis
Fachbereich Informatik
Fachhochschule Dortmund
Robert.Preis@fh-dortmund.de

Fragen bezüglich Eigenschaften

Es gibt einige Fragen zu Eigenschaften von Automaten bzw. regulären Sprachen, die immer wieder auftauchen:

Frage 1:

Ist die Sprache eines Automaten leer oder wird mindestens ein Wort akzeptiert?

Frage 2:

Ist ein Wort w Element der Sprache eines Automaten?

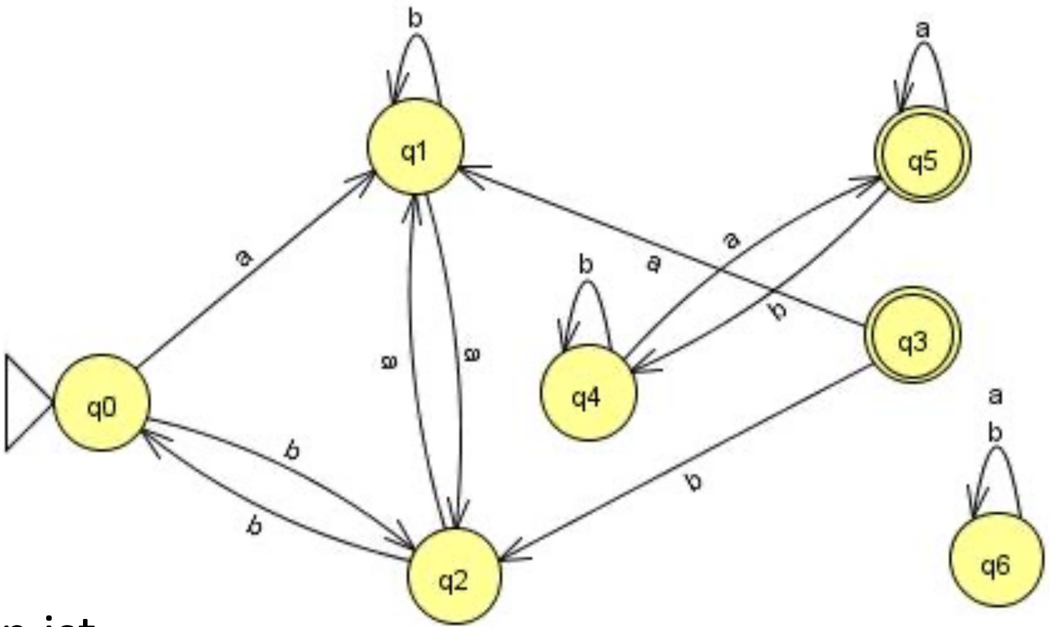
Frage 3:

Sind 2 Automaten äquivalent, d.h. akzeptieren sie dieselbe Sprache?

Frage 4:

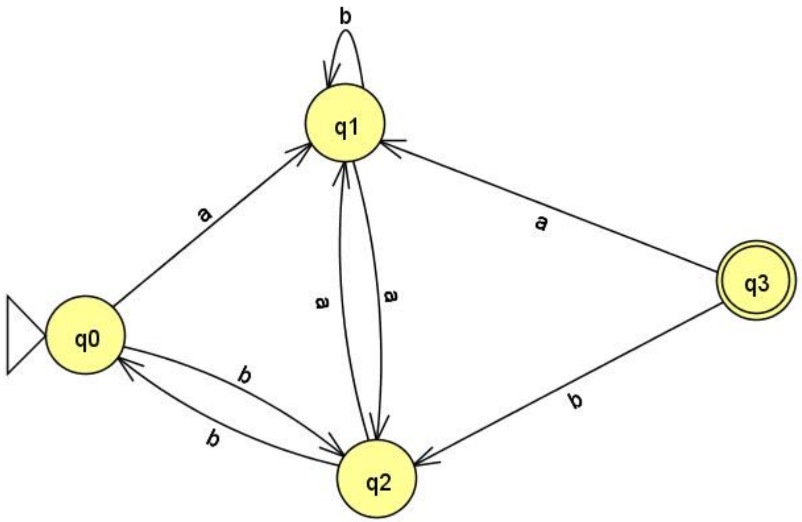
Kann man einen Automaten kleiner machen, d.h. mit noch weniger Zuständen?

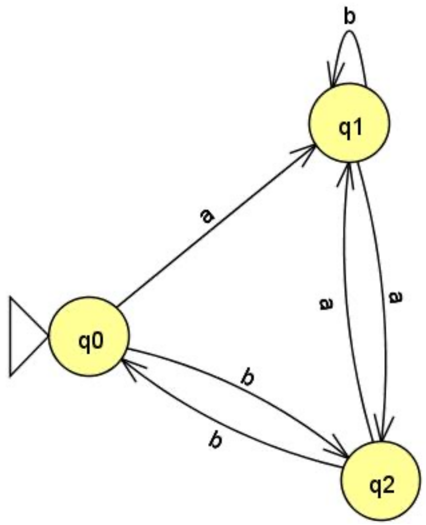
Minimierung von DEAs Teil 1: Erreichbare Zustände

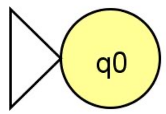


Bei vielen Automaten entdeckt man sofort Möglichkeiten zur Minimierung:

1. Nicht erreichbare Zustände löschen, d.h.
   a) die nicht mit der Start-Komponente verbunden sind.
   b) die vom Startzustand nicht erreichbar sind.
2. Falls es keinen Endzustand gibt, dann ist die Sprache leer und man braucht nur einen Zustand, den Startzustand.





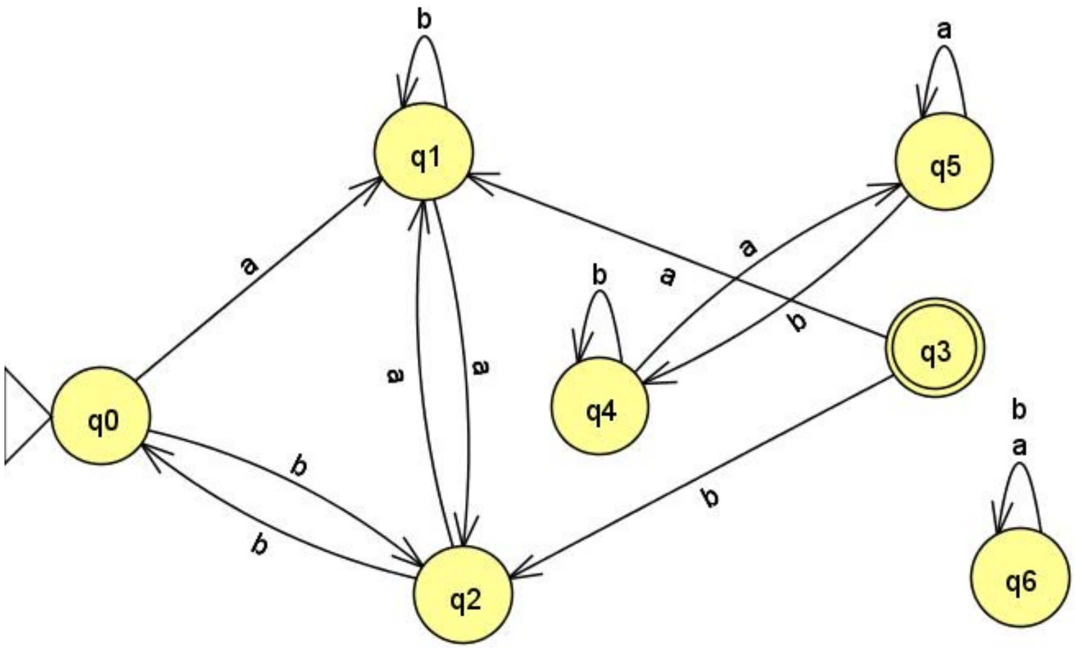


Sprache eines Automaten leer?

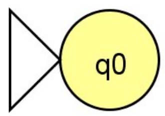
Frage 1:

Ist die Sprache eines Automaten leer oder wird mindestens ein Wort akzeptiert?

- *D.h. gibt es überhaupt einen Endzustand?*
- *D.h. gibt es überhaupt einen akzeptierenden Pfad?*
- *D.h. ist mindestens ein Endzustand vom Start erreichbar?*



Lösung zu Frage 1:
Mit der Minimierung der erreichbaren Zuständen kann man direkt feststellen, ob die Sprache leer ist.



Wortproblem

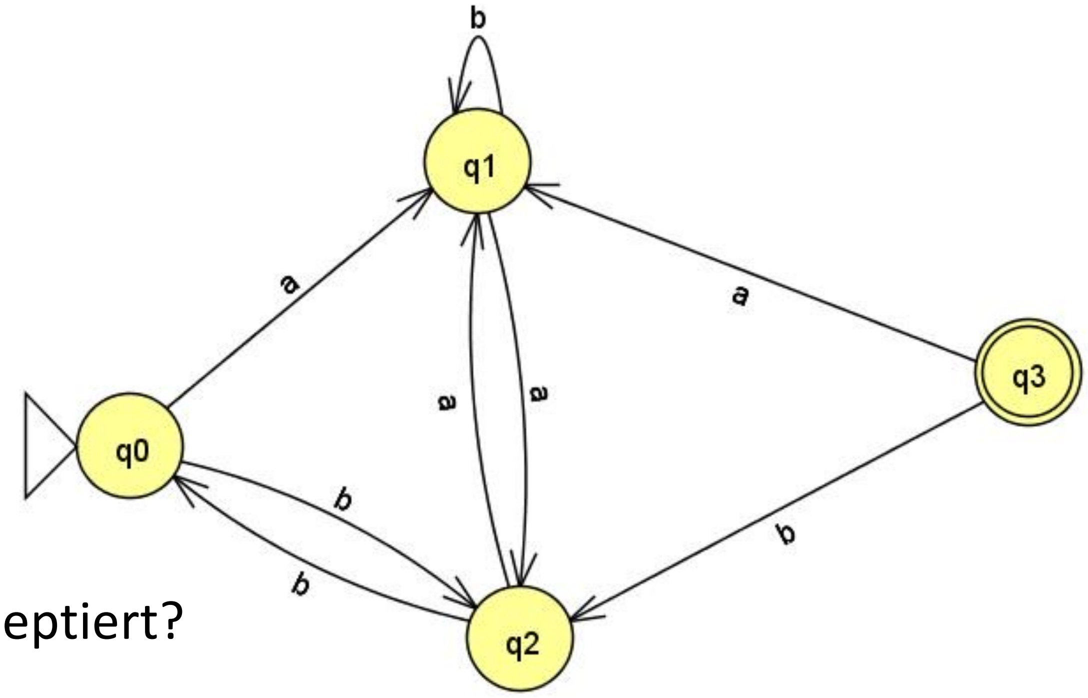
Frage 2:

Ist ein Wort w Element der Sprache eines Automaten?

D.h. gibt es einen akzeptierenden Pfad für das Wort w?

Lösung zu Frage 2:

1. Bestimme $q := \delta^*(q_0,w)$
2. Teste, ob $q \in F$



Beispiel: Wird das Wort w=abba akzeptiert?

Lösung: $\delta^*(q_0,abba) = q_2$

Antwort: Nein, weil q_2 kein Endzustand ist !

Äquivalenz und Minimierung

Frage 3:

Sind 2 Automaten äquivalent, d.h. akzeptieren sie dieselbe Sprache?

Frage 4:

Kann man einen Automaten kleiner machen, d.h. mit noch weniger Zuständen?

Wann akzeptieren zwei Automaten dieselbe Sprache?

- *Wenn für jedes Wort bei beiden Automaten dasselbe Ergebnis eintritt, d.h. wenn es egal ist, ob man das Wort bei dem Startzustand des einen oder des anderen Automaten startet.*

Wann kann man zwei Zustände zusammenfassen?

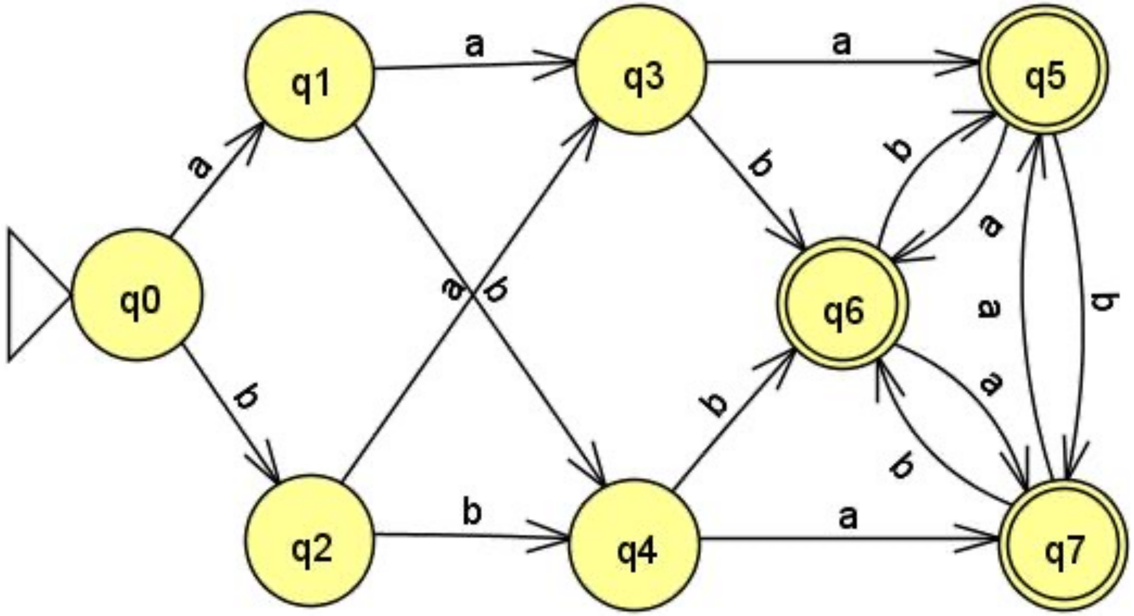
- *Wenn für jedes Wort bei beiden Zuständen dasselbe Ergebnis eintritt, d.h. wenn es egal ist, ob man das Wort von dem einen oder anderen Zustand weiter abarbeitet.*

…die Antworten klingen irgendwie ähnlich…

Äquivalenz von Zuständen

Das Wort abba gestartet auf q_1 und q_2 liefert dasselbe Ergebnis.

Gilt das für alle Wörter, die auf q_1 und q_2 gestartet werden?



Grundlegende Idee:
Bestimme äquivalente Zustände.

Definition: *Zwei Zustände sind äquivalent (nicht unterscheidbar), wenn sie bei allen Wörtern zu dem selben Ergebnis führen - entweder gehen beide in einen Endzustand oder beide in keinen Endzustand.*

Problem: Man muss alle Wörter überprüfen, die in den Zuständen starten.

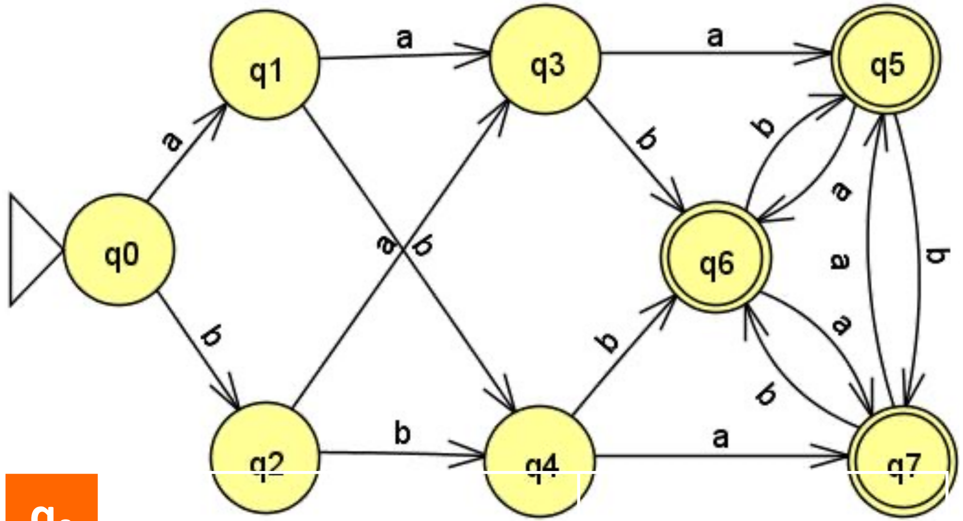
Lösung: Man kann einfacher entscheiden, ob zwei Zustände **NICHT** äquivalent sind. Dazu braucht man nämlich nur Gegenbeispiele.

Table Filling Algorithmus zur Unterscheidbarkeit von Zuständen (1)

1. Generiere Tabelle von Zuständen.
   Die Tabelle ist am Anfang leer, d.h. wir nehmen an, dass alle Zustände äquivalent sind.

Die Tabelle wird mit den kürzesten Gegenbeispielen für Äquivalenz gefüllt. Bei einem „Gegenbeispiel" kommt man von zwei Zustände zu einem unterschiedlichen Ergebnis.

2. Trage alle Gegenbeispiele der Länge 0 (d.h. Wort ε) ein:
   Bei ε kommen Endzustände und Nicht-Endzustände zu einem anderen Ergebnis, d.h. markiere alle solche Kombinationen mit einem ε.



q_0							
q_1							
q_2							
q_3							
q_4							
q_5	ε	ε	ε	ε	ε		
q_6	ε	ε	ε	ε	ε		
q_7	ε	ε	ε	ε	ε		
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

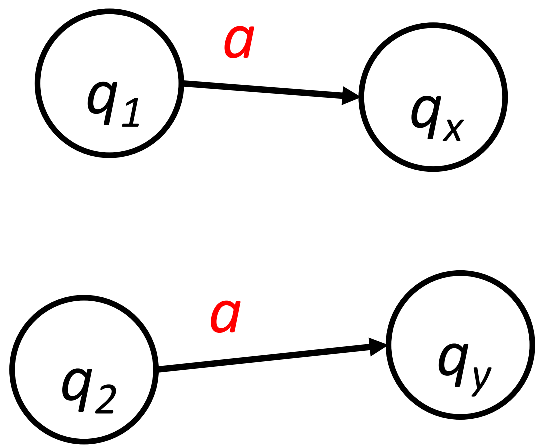
Gegenbeispiel suchen...

Szenario: Ich suche ein Gegenbeispiel für q_1 und q_2 :

Idee: *Falls es ein Gegenbeispiel gibt, dann könnte es ja mit einem „a“ anfangen...*

1. *Wohin gehen q_1 und q_2 mit einem „a“ ?*
   Z.B. geht q_1 nach q_x und q_2 geht nach q_y.

2. *Gibt es ein Gegenbeispiel für q_x und q_y in der Tabelle ?*



Falls Nein:

Probiere dasselbe mit einem „b"

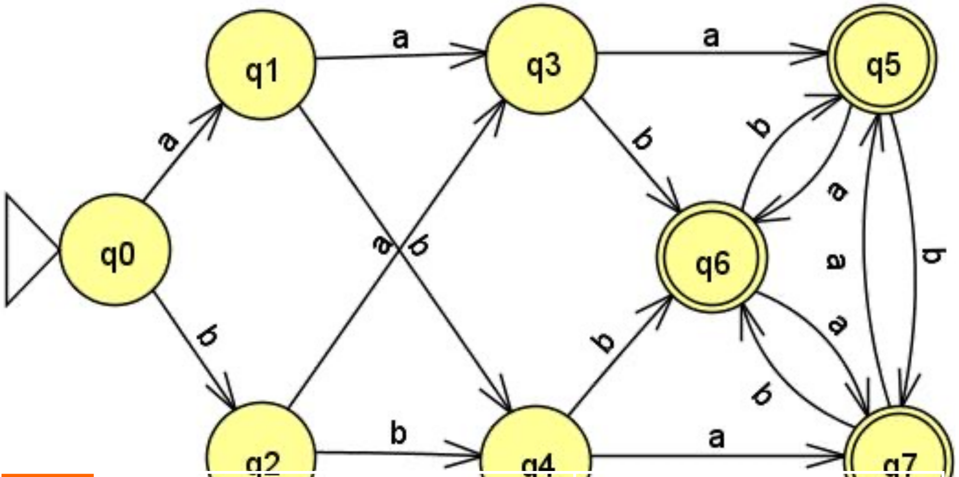
Falls Ja (und das Gegenbeispiel für q_x und q_y ist z.B. „bb“):

Bingo !

Dann ist das Wort „abb“ ein Gegenbeispiel für q_1 und q_2 !

Table Filling Algorithmus zur Unterscheidbarkeit von Zuständen (2)

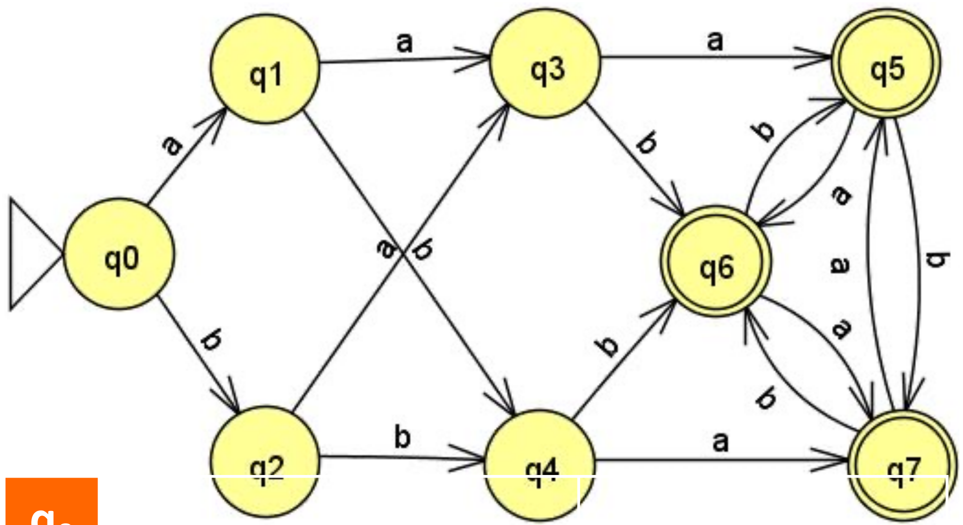
3. Trage alle Gegenbeispiele der Länge 1 ein, d.h. für alle offenen Zustands-Paare p und q:
   Ein Gegenbeispiel für p und q kann nur mit a oder b anfangen.
   Wohin gehen p und q bei a (b)? Probiere alle Buchstaben a, b, … nacheinander aus:
   - Falls p und q mit a (b) zu den Zuständen x und y gehen und für x/y ein Gegenbeispiel „w“ in der Tabelle ist, dann ist „aw“ („bw“) ein Gegenbeispiel für p und q.
   - Falls nicht, dann erst so lassen.



q_0							
q_1							
q_2							
q_3	a	a	a				
q_4	a	a	a				
q_5	ε	ε	ε	ε	ε		
q_6	ε	ε	ε	ε	ε		
q_7	ε	ε	ε	ε	ε		
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

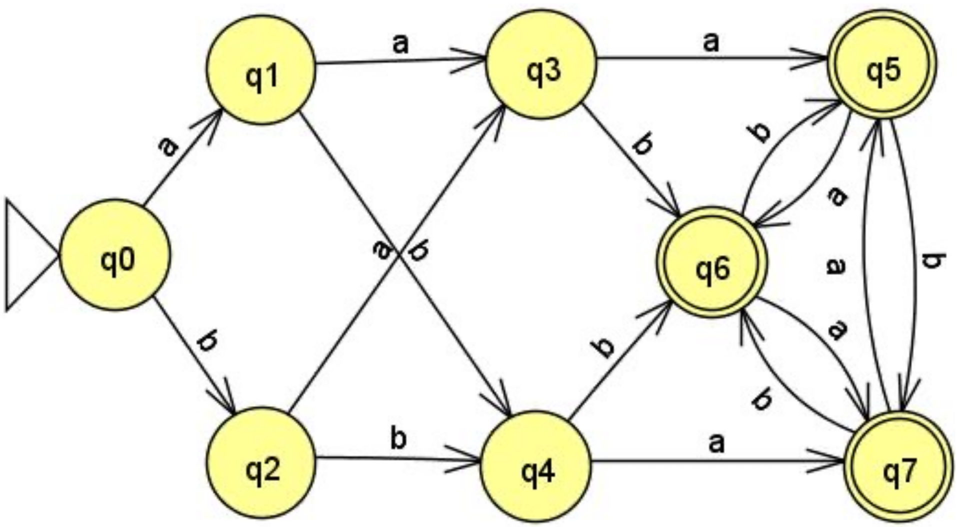
Table Filling Algorithmus zur Unterscheidbarkeit von Zuständen (3)

4. Trage in jeder weiteren Runde alle längeren Gegenbeispiele ein, d.h. für alle offenen p und q:
   Ein Gegenbeispiel für p und q kann nur mit a oder b anfangen.
   Wohin gehen p und q bei a (b)? Probiere alle Buchstaben a, b, … nacheinander aus:
   - Falls p und q mit a (b) zu den Zuständen x und y gehen und für x/y ein Gegenbeispiel „w“ in der Tabelle ist, dann ist „aw“ („bw“) ein Gegenbeispiel für p und q.
   - Falls nicht, dann erst so lassen.
5. Ende, wenn in einer Runde keine neuen Gegenbeispiele gefunden.



q_0							
q_1	aa						
q_2	aa						
q_3	a	a	a				
q_4	a	a	a				
q_5	ε	ε	ε	ε	ε		
q_6	ε	ε	ε	ε	ε		
q_7	ε	ε	ε	ε	ε		
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

Table Filling Algorithmus zur Unterscheidbarkeit von Zuständen (4)



q_0							
q_1	aa						
q_2	aa						
q_3	a	a	a				
q_4	a	a	a				
q_5	ε	ε	ε	ε	ε		
q_6	ε	ε	ε	ε	ε		
q_7	ε	ε	ε	ε	ε		
	q_0	q_1	q_2	q_3	q_4	q_5	q_6

Am Ende bleiben Lücken in der Tabelle.

Paare von Zuständen, in denen eine Lücke bleibt, sind äquivalent!

Jetzt können wir die Äquivalenzklassen der Zustände aufführen:

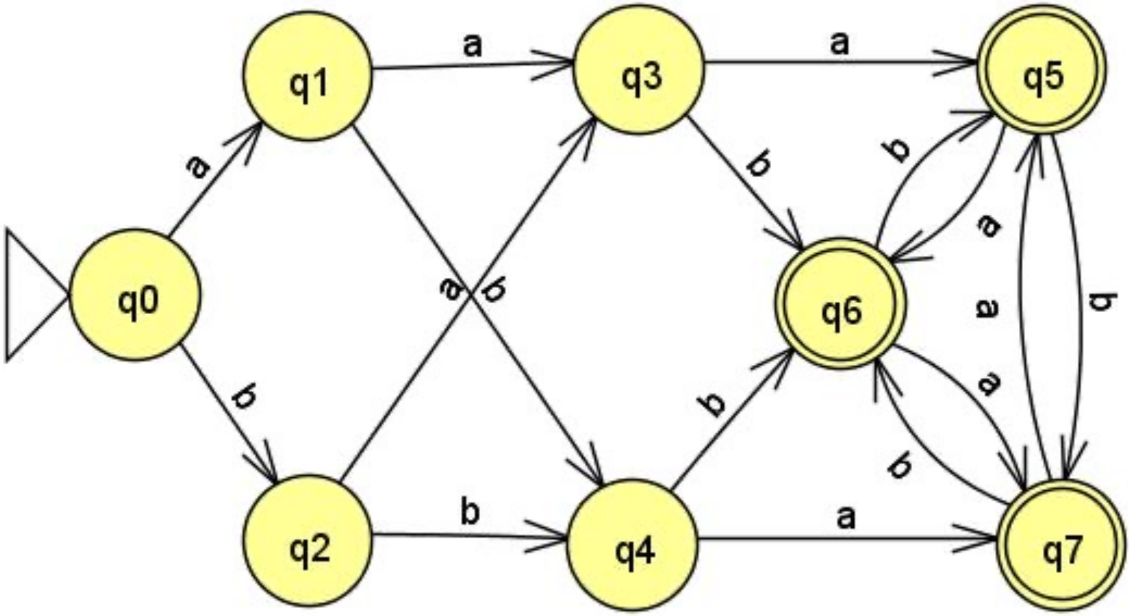
$\{q_0\}, \{q_1,q_2\}, \{q_3,q_4\}$ und $\{q_5,q_6,q_7\}$

Einzelne Zustände bilden dabei ihre eigene Äquivalenzklasse.

Anwendung Table Filling 1: Minimierung von DEAs Teil 2

Frage 4:

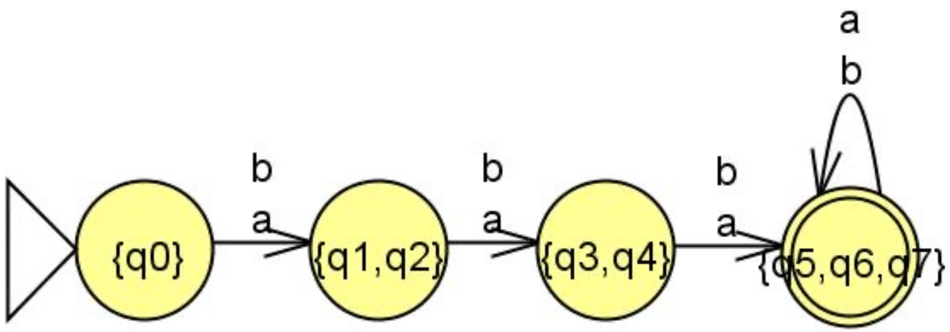
Kann man einen Automaten verkleinern, d.h. dieselbe Sprache mit weniger Zuständen erkennen?



Lösung zu Frage 4:

1. Entferne unerreichbare Zustände.
2. Fasse äquivalente Zustände zu einem zusammen, d.h.
   a) Bestimme Äquivalenzklassen
   b) Jede Äquivalenzklasse wird ein neuer Zustand

{q_0}, {q_1,q_2}, {q_3,q_4} und {q_5,q_6,q_7}



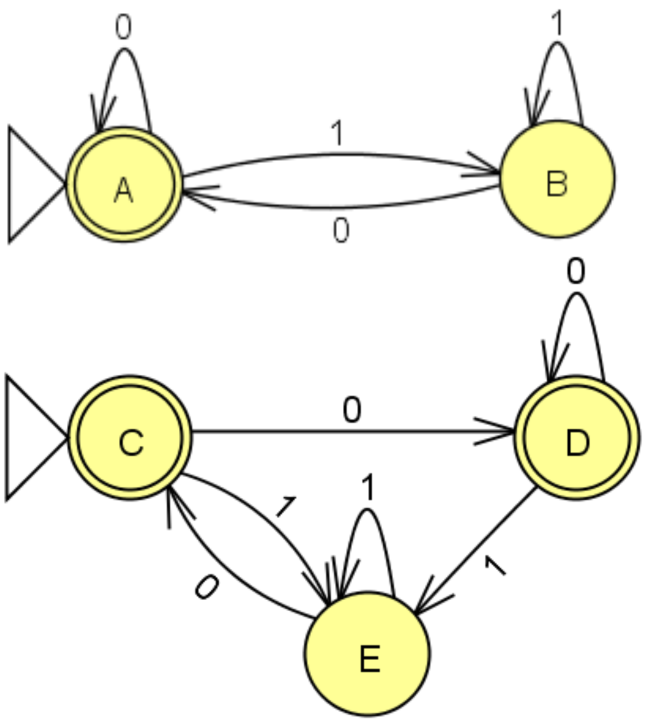
Der so konstruierte Automat ist minimal!

Anwendung Table Filling 2:
Sind zwei Automaten äquivalent?

Frage 3:

Sind 2 Automaten äquivalent, d.h. akzeptieren sie dieselbe Sprache?

Lösung zu Frage 3: Bilde Tabelle mit Zuständen aus beiden Automaten, betrachte aber nur die Teiltabelle mit Zustandspaaren aus unterschiedlichen Automaten. Bestimme dann wieder die äquivalenten Zustände!



C		ε
D		ε
E	ε	
	A	B

Es ist egal, ob ein Wort in A oder C gestartet wird – das Ergebnis ist immer gleich!

Wenn die beiden Startzustände äquivalent sind, dann akzeptieren die Automaten dieselbe Sprache !

Zusammenfassung

- Mit dem **Erreichbarkeitstest** kann man die Menge aller erreichbaren Zustände bestimmen.
- Mit dem **Table Filling Algorithmus** kann man die äquivalenten Zustände bestimmen, d.h. die Äquivalenzklassen.
- Man kann **testen**, ob
  1. die **Sprache eines Automaten leer** ist.
  2. ein **Wort** von einem Automaten **akzeptiert wird**.
  3. **zwei Automaten die selbe Sprache akzeptieren**
     (dann sind die beiden Startzustände äquivalent).
  4. ob ein Automat schon eine **minimale Anzahl von Zuständen** hat
     (dann sind alle Äquivalenzklassen einelementig, ansonsten kann man die Äquivalenzklassen jeweils zu einem Zustand vereinigen).