Fragen, Antworten und Kommentare zur aktuellen Vorlesung

Bitte nehmen Sie an der anonymen Lehrevaluation unter https://forms.gle/G7XQPDffkZsP1R3v9 teil. Die von mir kommentierten Ergebnisse stehen im letzten Fragen&Antworten-Dokument des Semesters.

Das Video zur Lösung der Aufgaben 43 und 44 finden Sie unter: https://youtu.be/24Fxx1cETtI (Aufgabennummern im Video stimmen nicht.)

Das Video zur Lösung der Aufgaben 45 und 46 finden Sie unter: https://youtu.be/lvN-Bp1uTUs

Frage: zur Sicherheit, zum Ausdruck a(b)* gehört auch das Wort a.

Antwort: Genau, da der Kleene-Stern auch für „hoch 0“ also ε steht. Außerhalb des Hilfstools zur Überprüfung kann dann auch ab* geschrieben werden, der Kleene-Stern bindet stärker.

Frage: Gibt es noch ein Beispiel zum Pumping Lemma?

Antwort: Schauen wir uns $L = \{a^n b^p c^m \mid n<m, p > 0\}$ an. Der erste Versuch ist immer einen Automaten anzugeben, der die Sprache akzeptiert. Wenn der nicht gefunden wird, wird mit dem Pumping-Lemma versucht zu zeigen, dass es keinen Automaten gibt. Dazu muss als Gegenbeispiel nur ein nicht aufpumpbares oder/und schrumpfbares Wort gefunden werden. Im Beispiel ist es das Wort $a^n bc^{n+1}$. Dann muss argumentiert werden, dass es kein Teilwort x w=vxy zum Aufpumpen/Schrumpfen gibt. Hier:

x=a (oder mehrere a) verstößt irgendwann beim Aufpumpen gegen n<m.

x=ab (oder mehrere b) verstößt gegen die Struktur, da z. B. das Teilwort abab nicht erlaubt ist, nicht zu Wörtern der Sprache gehört

x=b, pumpen geht, aber schrumpfen oder weglassen ist nicht erlaubt, da ein b gefordert wird

x=bc (oder mehrere c) verstößt wieder gegen die geforderte Struktur beim Aufpumpen

x=c (oder mehrere c), pumpen geht, aber schrumpfen oder weglassen ist nicht erlaubt, da dann die Anzahl der a und b gleich werden würde

Beweise sehen oft ähnlich auf und basieren darauf, dass Automaten nicht beliebig weit zählen können.

Die Beweisidee klappt nicht, wenn mathematische Konstruktionen bei der Angabe der Sprache erscheinen.

Frage: Gilt das Pumping-Lemma in beide Richtungen?

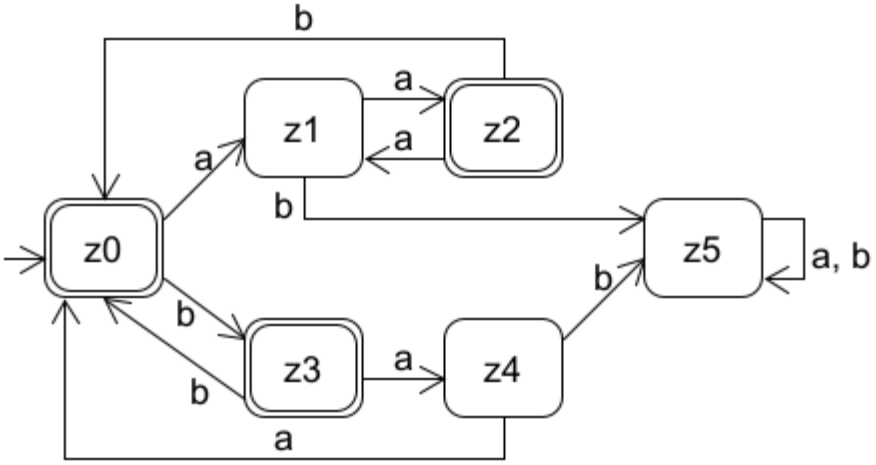
Antwort: Nein, wenn eine Sprache regulär und nicht endlich ist, muss sie aufpumbar sein. Anders herum gilt **nicht**, wenn eine Sprache aufpumpbar ist, muss sie regulär sein. Ein konkretes Gegenbeispiel ist die Sprache $L = \{c^m a^n b^n \mid m,n >=0\} \cup \{a^m b^n \mid m,n >= 0\}$. Hat ein Wort c’s am Anfang können diese aufgepumpt und geschrumpft werden. Für Worte der Form ohne c am Anfang, also $a^n b^n$ können die a’s aufgepumpt werden und das Wort liegt in der zweiten Teilmenge. Die zweite Teilmenge ist trivial aufpumpbar. Die Sprache ist anschaulich argumentiert nicht regulär, da bei

einem Automaten nach c's bei einem ersten a die Anzahl der a gezählt werden muss, was Automaten nicht können.

Frage: Wie geht nochmal der zweite Schritt bei der Äquivalenzberechnung der Zustände, nachdem ich die End- und Nichtendzustände getrennt habe.

Antwort: Der 2. geht wie jeder folgende n + 1. Schritt. Sie haben ein Zustandspaar (qi,qj) was aktuell nach n Schritten noch äquivalent ist. Dann berechnen Sie für jedes Zeichen des Automaten, z. B. z das Ergebnis ueber(qi,z)=ri und ueber(qj,z)=rj. Sie prüfen dann, ob (ri,rj) in der bisherigen Tabelle äquivalent sind, wenn ja, passiert nichts, wenn nicht, wird dies eingetragen.

Achtung beim 2. Schritt ist das die Prüfung ob ri und rj beide Endzustände oder beide Nicht-Endzustände sind. Die Idee ist aber ab dem 3. Schritt nicht hilfreich, da Sie jetzt nicht mehr auf erreichte End- oder Nicht-Endzustände prüfen können, da jetzt ja Worte der Länge 2 also 2 Schritte betrachtet werden.

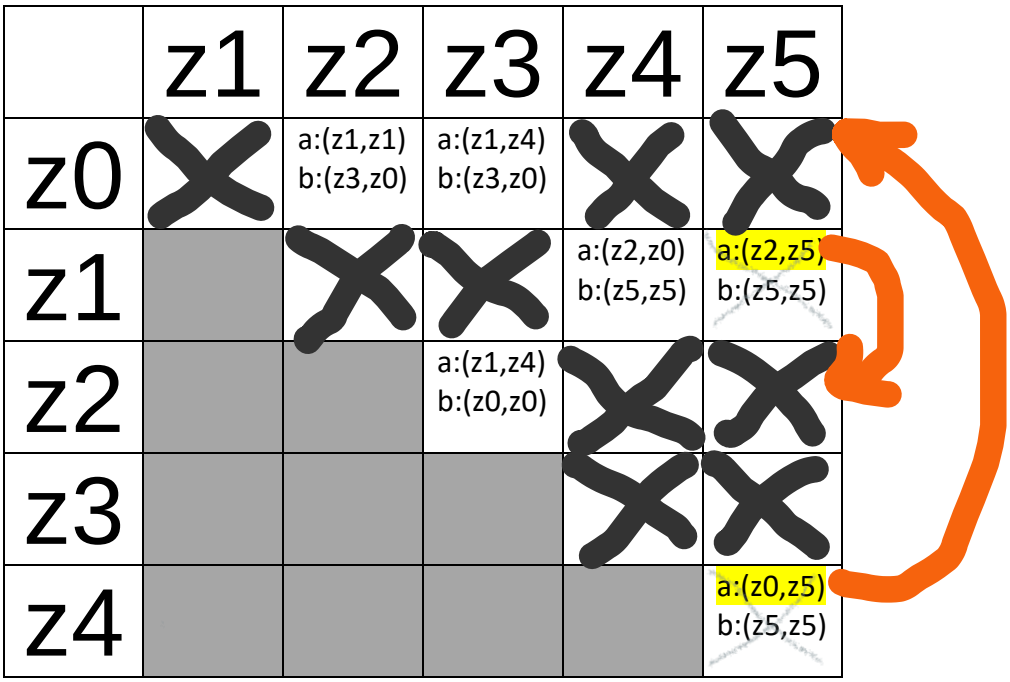


	z1	z2	z3	z4	z5
z0	X	a:(z1,z1) b:(z3,z0)	a:(z1,z4) b:(z3,z0)	X	X
z1		X	X	a:(z2,z0) b:(z5,z5)	a:(z2,z5) b:(z5,z5)
z2			a:(z1,z4) b:(z0,z0)	X	X
z3				X	X
z4					a:(z0,z5) b:(z5,z5)

Die obige Matrix zeigt die Lösung nach dem ersten Schritt, bei dem nur zwischen End- und Nichtendzuständen unterschieden wird. Formal gibt es ein Wort der Länge 0, mit von einem

Zustand in einen Endzustand und dem anderen Zustand in einen Nichtendzustand kommt, wodurch die Zustände nicht äquivalent sind.

In den Zustandspaaren, die noch äquivalent sein könnten sind für die Zeichen a und b jeweils die Folgezustände eingetragen. Sollten diese nicht äquivalent sein, ist auch das Ursprungspaar nicht äquivalent. Formaler war das Zustandpaar für Werte der Länge n (am Anfang 0) äquivalent, es wurde aber ein Wort der länge n+1 gefunden, so dass die Zustände nicht äquivalent sind.



Die vorherige Abbildung zeigt, dass z1 und z5 nicht äquivalent sein können, da eines der berechneten Ergebnispaare, hier (z2,z5) nach dem vorherigen Berechnungsschritt nicht äquivalent sind. Ähnliches mit (z4,z5) mit (z0,z5).

Im nächsten Schritt wird erkannt, dass alle 4 in der Tabelle stehenden Zustandspaare weiterhin als äquivalent angenommen werden, so dass es für Worte der Länge 2 keine weiteren nicht-äquivalenten Zustände gefunden werden.

Frage: Müssen wir die Äquivalenzmatrix nach dem Verfahren ausfüllen oder können wir direkt die Tabelle füllen und den Automaten angeben?

Antwort: Generell gilt, Sie müssen keines der Verfahren anwenden, solange Sie die geforderten Ergebnisse, hier Matrix/Tabelle und Automat liefern. Sollten Sie aber Fehler machen, kann ich beim Korrigieren bei der Anwendung des vorgeschlagenen Verfahrens besser erkennen ob es sich um einen Flüchtigkeitsfehler oder um einen systematischen bzw. Denkfehler handelt. Das eine kostet einzelne Punkte, das andere die Hälfte oder deutlich mehr Punkte.