Endliche Automaten

In der ersten Vorlesungswoche wollen wir uns mit *endlichen Automaten* beschäftigen. Um uns diesen zu nähern, betrachten wir zunächst einen einfachen Lichtschalter. Dieser kann 'an' oder 'aus' sein, also in genau einem von zwei *Zuständen*. Zudem sind durch die Aktion 'Schalter betätigen' *Zustandsübergänge* möglich. Abbildung 1 zeigt eine graphische Darstellung. Die Kantenbeschriftung b steht dabei für die Betätigung des Schalters.

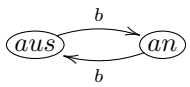

Abbildung 1: Modell eines einfachen Lichtschalters

Gehen wir nun davon aus, dass der Lichtschalter zu Beginn ausgeschaltet ist, so soll der Zustand 'aus' zudem der (eindeutige) *Startzustand* sein. Wollen wir ferner bspw. ein System modellieren, in dem nur jene Aktionsfolgen "gut" sind, bei denen das Licht am Ende wieder ausgeschaltet ist, so modellieren wir den Zustand 'aus' noch als *Endzustand*. Abbildung 2 zeigt das ergänzte System. Der Pfeil in den Zustand 'aus' hinein deutet an, dass 'aus' der Startzustand ist. Der doppelte Kreis um 'aus' bedeutet, dass 'aus' ein Endzustand ist.

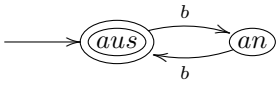

Abbildung 2: Verbessertes Modell eines einfachen Lichtschalters

Damit haben wir bereits alle Komponenten eines *endlichen Automaten* vorliegen. Wichtig ist bei diesem Beispiel, dass

- wir ein System mit einer *endlichen Anzahl von Zuständen* haben,
- von denen *einer* als *Startzustand* und
- eine *beliebige Teilmenge* als *Endzustände* hervorgehoben sind,
- dass Zustands*übergänge* möglich sind und
- dass das System sich stets in *genau einem Zustand befindet*.

Das dynamische Verhalten des Systems lässt sich wie folgt beschreiben: Wir beginnen im Startzustand (hier 'aus'). Dann können genau die Aktionen ausgeführt werden, die mit Kanten aus diesem Zustand herausgehen. In diesem Fall gibt es nur die Aktion b, die uns in den Zustand 'an' überführt. Von dort geht es dann weiter. Führen wir die Aktion b bspw. dreimal hintereinander aus (wir schreiben dafür später $b \cdot b \cdot b$ oder auch einfach bbb), so enden wir im Zustand 'an'. Führen wir b hingegen nur zweimal aus (also $b \cdot b$), so enden wir im Zustand 'aus'. Da 'aus' ein Endzustand ist, sagen wir, dass $b \cdot b$ *akzeptiert* wird (bbb hingegen nicht, da 'an' kein Endzustand ist).

Während die bisher skizzierte Idee eines Systems mit Aktionen recht intuitiv ist (man denke z.B. an einen Fahrkarten- oder einen Süßigkeitenautomaten, die stets in einem Zustand sind und durch Aktionen in andere Zustände überführt werden), wird in der (theoretischen) Informatik meist eine andere Modellvorstellung verwendet. Die beiden Vorstellungen können aber ineinander überführt werden und sind daher im Grunde identisch. Die in der Informatik gängige Vorstellung ist die eines Automaten, der ein *Eingabewort* auf einem *Eingabeband* hat und dieses Wort Buchstabe für Buchstabe liest. Abbildung 3 unten zeigt eine skizzenhafte Darstellung. Dieser Automat besitzt drei Zustände, z_0, z_1, z_2, wobei z_0 der Startzustand und z_2 ein Endzustand ist. An den Kantenbeschriftungen kann man zudem erkennen, dass er as und bs lesen kann. Das Eingabeband ist über dem Automaten notiert. Das Eingabewort ist *abba* (die weiteren leeren Kästchen und die schwarzen Punkte sollen andeuten, dass man auch längere Worte auf diesem Band notieren kann) und der gestrichelte Pfeil zwischen Automat und Eingabeband zeigt an, welcher Buchstabe des Wortes gerade gelesen wird (man spricht hier auch vom *Lesekopf*). Ist der Automat zu Anfang in z_0, so kann er nun das a lesen (da es eine a-Kante aus z_0 heraus gibt). Der Automat wechselt dann in den Zustand z_1 und der Lesekopf wandert ein Feld weiter. Nun können die zwei b gelesen werden, wobei der Automat im Zustand z_1 bleibt. Der letzte Buchstabe (das a) kann dann auch gelesen werden, wobei der Automat in den Zustand z_2 wechselt. Da dies ein Endzustand ist *und* das Wort bis zum Ende gelesen wurde, akzeptiert der Automat das Wort *abba*.

Die Vorstellung eines Eingabebandes ist historisch gewachsen und erscheint heute eher altertümlich. Sie ist aber eine gute Abstraktion und man kann sich recht gut überlegen, dass die Vorstellungen eines Wortes im (Haupt-)Speicher oder einer Folge von Aktionen wie zu Anfang recht ähnlich sind. Das wichtige

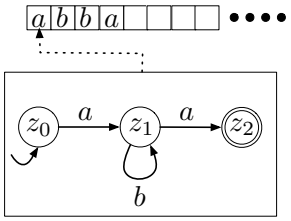

Abbildung 3: Skizze eines endlichen Automaten

ist hier der Automat, der ein Wort *von links nach rechts* liest und dieses letztendlich akzeptiert oder nicht. Nicht akzeptiert wird dabei ein Wort dann, wenn entweder das Wort gar nicht zu Ende gelesen werden kann (dies tritt auf, wenn der Automat in einem Zustand z ist und einen Buchstaben x lesen soll, zu dem es keine Kante aus z heraus gibt; man sagt dann, der Automat *blockiert*) oder aber das Wort zu Ende gelesen werden kann, der Automat dann aber nicht in einem Endzustand ist.

Die Menge aller Wörter, die ein gegebener Automat akzeptiert, bezeichnen wir als seine akzeptierte *Sprache*. Warum diese wichtig ist, werden wir in der Vorlesung genauer behandeln. Im obigen Beispiel aus Abbildung 2 akzeptieren wir z.B. genau die Wörter, die nur aus bs bestehen und die eine gerade Anzahl bs enthalten. Im Beispiel aus Abbidlung 3 werden jene Wörter aus as und bs akzeptiert, die mit genau einem a beginnen und enden und die dazwischen beliebig viele (auch 0) bs haben.

Man kann sich die Frage stellen, ob solche Automaten einen Anwendungsfall haben, da sie recht einfach erscheinen. Erstens sind solche endlichen Automaten die Grundlage für prinzipiell alle weiteren Automatenmodelle, mit denen dann sehr viel komplexere Systeme modelliert werden können. Zweitens sind bereits endliche Automaten für viele Anwendungen von Bedeutung. Bspw. basiert die Worterkennung bei einem Compiler (um bspw. das Schlüsselwort 'if' zu finden) auf endlichen Automaten. Auch Protokolle und viele Anwendungen aus der technischen Informatik lassen sich mit endlichen Automaten (oder leichten Varianten davon) modellieren und dann implementieren (nachdem sich das Modell als hoffentlich fehlerfrei erwiesen hat).

Nichtdeterminismus

Die eben vorgestellten und diskutierten Automaten sind *endlich*, weil die Zustandsmenge endlich ist, und *deterministisch*, weil es zu jedem Paar aus Zustand und Buchstabe immer genau einen Nachfolgezustand gibt. D.h. wenn der Automat in einem Zustand ist und einen Buchstaben liest, dann ist der Nachfolgezustand (der Zustand, in den einen das Lesen dieses Buchstaben überführt) eindeutig bestimmt. Bei einem *nichtdeterministischen*, endlichen Automaten wird nun genau diese Bedingung aufgeweicht.

In Abbildung 4 ist ein *nichtdeterministischer*, endlicher Automat dargestellt. Dieser startet in z_0. Beginnt das zu lesende Wort nun mit einem a, so kann der Automat in den Zustand z_1 *oder* in den Zustand z_2 wechseln. Für beides gibt es einen Übergang.

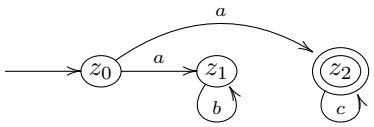

Abbildung 4: Ein nichtdeterministischer, endlicher Automat

Schwierig beim Nichtdeterminismus ist nun die Vorstellung, dass nach Lesen des Buchstabens a der Automat quasi *in beiden Zuständen (z_1 und z_2) gleichzeitig ist*. Erhält der Automat aus Abbildung 4 bspw. das Wort ab als Eingabe, so startet er im Startzustand z_0 mit der Eingabe ab. Durch Lesen von a wechselt er in den Zustand z_1 *und nichtdeterministisch* auch in den Zustand z_2. Als nächstes ist nun das b zu lesen. Im Zustand z_1 kann dies gelesen werden und wir verbleiben dabei im Zustand z_1. Im Zustand z_2 kann dies nicht gelesen werden und der Automat blockiert hier (d.h. diese Rechnung geht nicht weiter und kann wie ein Abbruch interpretiert werden). Er ist somit nur noch im Zustand z_1. Hier ist das Wort zu Ende gelesen, wird aber nicht akzeptiert, da z_1 kein Endzustand ist.

Bei ac als Eingabe wäre der Automat nach Lesen des Buchstabens a zunächst wieder nichtdeterministisch in den Zuständen z_1 und z_2. Da nun ein c kommt, blockiert der Automat in z_1, kann aber in z_2 das Wort zu Ende lesen. Da er dort nun auch in einem Endzustand ist, wird ac akzeptiert. Informal bedeutet dies, dass wenn man eine Möglichkeit finden kann, das Wort zu Ende zu lesen und dann in einem Endzustand zu landen, der nichtdeterministische Automat akzeptiert. In der Vorlesung werden wir hierzu formal den Begriff der *Rechnung* definieren.

Das Konzept des Nichtdeterminismus ist für viele anfangs recht schwierig zu verstehen, mit etwas Übung gewöhnt man sich aber daran und es ist ein Konzept, das in der Informatik sehr weit verbreitet ist und sehr erfolgreich in verschiedensten Kontexten eingesetzt wird. Wir werden diesem Konzept in der Vorlesung mehrmals begegnen und versuchen, ein Verständnis dafür aufzubauen.

Spannend ist nun die Frage, ob der nichtdeterministische Automat Dinge kann, die der deterministische nicht kann? Oder ob es eine Transformation gibt? Letzteres wäre praktisch, da es viele Fälle gibt, die leicht mit einem nichtdeterministischem Automaten modelliert werden können. Könnte man diesen dann automatisch in einen deterministischen Automaten umwandeln, kann man sich u.U. viel Arbeit (und Fehlerquellen) sparen. Wir werden dies in der Vorlesung diskutieren und dort auch zeigen, dass die beiden Automatenmodelle tatsächlich gleichmächtig sind und jeder nichtdeterministische, endliche Automat in einen deterministischen, endlichen Automaten umgewandelt werden kann.

Selbsttest (Lösung auf Seite 6)

1. Wie viele Startzustände kann ein endlicher Automat haben?
2. Wie viele Endzustände kann ein endlicher Automat haben?
3. Wann akzeptiert ein endlicher Automat ein Eingabewort?
4. Wie viele a-Kanten, kann ein deterministischer Automat haben?
5. Wie viele a-Kanten, kann ein nichtdeterministischer Automat haben?
6. Wie würden Sie in Ihren Worten ‘Nichtdeterminismus’ beschreiben?

Die mathematische Seite

Ist $\Sigma = \{a, b\}$ ein Alphabet, so wird mit Σ^* die Menge aller endlichen Worte bezeichnet, die aus a und b gebildet werden können. Mit $\cdot$ wird die *Konkatenation* zweier Worte bezeichnet, also z.B. $ab \cdot ba = abba = a \cdot b \cdot b \cdot a$. Σ^* kann daher auch verstanden werden als die Vereinigung von Σ mit $\Sigma \cdot \Sigma$ und $\Sigma \cdot \Sigma \cdot \Sigma$ usw. Ferner ist in Σ^* noch das *leere Wort* enthalten, d.h. das Wort, das aus 0 Buchstaben besteht. Wir notieren dies als ϵ oder λ. Wir können nun den endlichen Automaten definieren.

Definition 1. *Ein* deterministischer, endlicher Automat *(kurz DFA, vom englischen 'finite') ist ein* 5*-Tupel* $A = (Z, \Sigma, \delta, z_0, Z_{end})$ *mit*

- *Der endlichen Menge von* Zuständen Z*.*
- *Dem endlichen Alphabet* Σ *von* Eingabesymbolen*.*
- *Der* Überführungsfunktion $\delta : Z \times \Sigma \to Z$*.*
- *Dem* Startzustand $z_0 \in Z$*.*
- *Der Menge der* Endzustände $Z_{end} \subseteq Z$*.*

Bei einem nichtdeterministischen*, endlichen Automaten ist die Überführungsfunktion* $\delta : Z \times \Sigma \to 2^Z$*. Zudem gibt es statt eines Startzustandes eine Menge* $Z_{start} \subseteq Z$ *von Startzuständen.*

Erhält ein DFA ein Eingabewort $w \in \Sigma^*$, so beginnt er im Startzustand z_0. Beginnt w mit dem Symbol $x \in \Sigma$, so wird der Nachfolgezustand nun durch $\delta(z_0, x)$ bestimmt. Dies wird dann in dem nun aktuellen Zustand und mit dem Restwort fortgeführt. Formal lässt sich dies (zunächst nur für den DFA) wie folgt präzisieren:

Definition 2. *Sei* $A = (Z, \Sigma, \delta, z_0, Z_{end})$ *ein DFA. Die* erweiterte Überführungsfunktion $\hat{\delta} : Z \times \Sigma^* \to Z$ *wird für alle* $z \in Z$*,* $x \in \Sigma$ *und* $w \in \Sigma^*$ *rekursiv definiert durch*

$$\begin{aligned} \hat{\delta}(z, \lambda) &:= z \\ \hat{\delta}(z, xw) &:= \hat{\delta}(\delta(z, x), w) \end{aligned}$$

Alternativ können wir auch mit Konfigurationen arbeiten: Eine Konfiguration *eines DFA* A *ist ein Tupel* $(z, w) \in Z \times \Sigma^*$ *mit der Bedeutung, dass* A *im Zustand* z *ist und noch das Wort* w *zu lesen ist. Ein* Konfigurationsübergang *ist dann* $(z, w) \vdash (z', v)$ *gdw.* $w = xv$*,* $x \in \Sigma$ *und* $\delta(z, x) = z'$ *ist. Eine* Rechnung *auf dem Wort* $w \in \Sigma^*$ *ist eine Folge von Konfigurationsübergängen, die in* (z_0, w) *beginnt. Endet die Rechnung in* (z', λ) *und ist* $z' \in Z_{end}$*, so ist dies eine* Erfolgsrechnung *und* w *wird akzeptiert.*

Zuletzt ist die von einem DFA A akzeptierte Sprache *die Menge*

$$L(A) := \{w \in \Sigma^* \mid \hat{\delta}(z_0, w) \in Z_{end}\} = \{w \in \Sigma^* \mid (z_0, w) \vdash^* (z_e, \lambda), z_e \in Z_{end}\}$$

Lösungen zum Selbsttest auf Seite 4

1. **F:** Wie viele Startzustände kann ein endlicher Automat haben?

   A: Ein *deterministischer* Automat hat immer genau einen Startzustand. Ein *nichtdeterministischer* Automat kann auch mehr als einen Startzustand haben. Hier wird dann zu Beginn nichtdeterministisch ein Startzustand gewählt, d.h. ähnlich wie wir in dem Beispiel aus Abbildung 4 nach Lesen des a in zwei Zuständen sind, sind wir bei einem nichtdeterministischem Automaten mit mehreren Startzuständen zu Anfang in mehreren Zuständen.

2. **F:** Wie viele Endzustände kann ein endlicher Automat haben?

   A: Hier ist jede Zahl zwischen 0 und und der Anzahl der Zustände möglich, d.h. ein Automat kann keinen Endzustand haben (dann wird allerdings auch kein einziges Wort akzeptiert) oder jede beliebige Teilmenge der Zustände kann zu Endzuständen gemacht werden. Tatsächlich ist es nicht unüblich mehr als einen Endzustand zu haben.

3. **F:** Wann akzeptiert ein endlicher Automat ein Eingabewort?

   A: Wenn das Eingabewort bis zum Ende gelesen wird *und* sich der Automat dann in einem Endzustand befindet, bzw. bei einem nichtdeterministischem Automaten, wenn es *mindestens eine* Möglichkeit gibt, das Eingabewort bis zum Ende zu lesen und dann in einem Endzustand zu sein.

4. **F:** Wie viele a-Kanten kann ein deterministischer Automat haben?

   A: Aus jedem Zustand kann nur genau eine a-Kante herausgehen (sonst wäre der Automat nicht mehr deterministisch). Damit kann es maximal so viele a-Kanten geben, wie es Zustände gibt.

5. **F:** Wie viele a-Kanten kann ein nichtdeterministischer Automat haben?

   A: Aus jedem Zustand kann zu jedem anderen Zustand (inkl. dem Zustand selbst!) eine a-Kante führen. Sei n die Anzahl der Zustände, dann gibt es aus einem Zustand also maximal n a-Kanten. Da dies für jeden der n Zustände gilt, haben wir insgesamt maximal $n \cdot n = n^2$ a-Kanten.

6. **F:** Wie würden Sie in Ihren Worten 'Nichtdeterminismus' beschreiben?

   A: In unserem Kontext vielleicht so: Nichtdeterminismus ist die Möglichkeit, bei gleichen Voraussetzungen (hier: in einem bestimmten Zustand zu sein und einen bestimmten Buchstaben zu lesen) verschiedene Dinge tun zu können (hier: in verschiedene Zustände wechseln zu können) ohne das festgelegt wäre, wie diese 'Wahlmöglichkeit' nun aufgelöst wird. (Bei uns kommt nun noch hinzu, dass wir quasi alle Möglichkeiten ausprobieren und dann später die für uns günstigste (hier: die, bei der das Wort akzeptiert wird) auswählen.)