2 Endliche Automaten

In diesem Kapitel wollen wir den *endlichen Automaten* einführen, das einfachste Automatenmodell, das aber recht nützlich ist und als Grundlage für weitere Automatenmodell dient.

Um uns diesem 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 2.1 zeigt eine graphische Darstellung. Die Kantenbeschriftung b steht dabei für die Betätigung des Schalters.

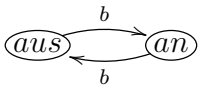

Abbildung 2.1: Modell eines einfachen Lichtschalters

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 2.2 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.

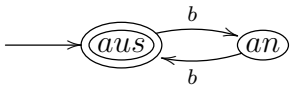

Abbildung 2.2: Verbessertes Modell eines einfachen Lichtschalters

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).

Aus der bisherigen Beschreibung ist der Nutzen von endlichen Automaten und der Bezug zur Informatik vielleicht noch nicht ersichtlich. Tatsächlich sind mit endlichen Automaten zunächst eine Vielzahl von Anwendungen und Problemen *modellierbar*. Beispielsweise werden endliche Automaten benutzt, um Schaltkreise oder Kommunikationsprotokolle zu modellieren. Die Modelle dienen dabei insbesondere zur Kommunikation in Entwicklungsteams und zur Fehlerfindung. Daneben sind endliche Automaten wichtig, um auf ihnen aufbauende kompliziertere Automatenmodelle zu verstehen. Beispielsweise wollen wir später die Turingmaschine einführen, mit der man dann formal argumentieren kann, welche Probleme nicht von einem Computer gelöst werden können.

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 2.3 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 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 2.2 akzeptieren wir z.B. genau die Wörter, die nur aus bs bestehen und die eine gerade Anzahl bs enthalten. Im Beispiel aus Abbidlung 2.3 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).

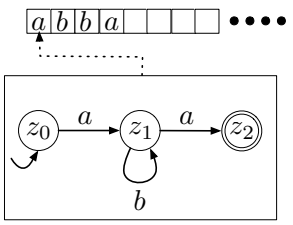

Abbildung 2.3: Skizze eines endlichen Automaten