**Formale Sprachen und Automaten**

**1. Einführung**

**Berechenbarkeit:** Die Berechenbarkeit beschäftigt sich mit der Frage, welche Probleme von einem Computer berechenbar sind und welche Voraussetzungen dieser erfüllen muss, damit er alle berechenbaren Probleme lösen kann.

**Turing-Berechenbarkeit:** Ein Problem ist berechenbar, wenn es von einer Turing-Maschine berechnet werden kann.

**Church-Turing-These:** Die beiden Begriffe der Berechenbarkeit sind äquivalent, d.h. für jedes intuitiv berechenbare Problem gibt es auch eine Turingmaschine, die das Problem löst.

**Komplexitätstheorie:** Neben der Berechenbarkeit muss auch geprüft werden, ob ein Problem handhabbar ist, d.h. ob es auch für große Eingabewerte in vertretbarer Zeit gelöst werden kann:

* Algorithmen mit polynomialer Laufzeit gelten als handhabbar
* Algorithmen mit exponentieller Laufzeit gelten als nicht handhabbar

**Komplexitätsklassen:**

* P: Problem kann in polynomialer Zeit von einem deterministischen System berechnet werden
* NP: Problem kann in polynomialer Zeit von einem nichtdeterministischen System berechnet werden

**Formale Sprachen:** Formale Sprachen lassen sich als Analogie zu natürlichen Sprachen wie folgt definieren:

* Man wählt eine Menge von Buchstaben als Alphabet und erzeuge Worte durch Aneinanderhängen von Buchstaben aus dem Alphabet.
* Eine Teilmenge der gesamten Wortmenge bezeichnet eine formale Sprache.

**Automaten:** Automaten sind das akzeptierende Konzept: Eine Sprache gehört zu einer Sprachklasse, wenn es einen Automaten des vorgegebenen Automatentyps gibt, der die Sprache akzeptiert.

**Grammatik:** Grammatik ist das erzeugende Konzept: Eine Sprache gehört zur Sprachklasse, wenn es eine Grammatik des vorgegebenen Grammatiktyps gibt, der die Sprache erzeugt.

**2. Formale Sprachen und Grammatiken**

**Grundlegende Begriffe:**

* Alphabet: Eine nichtleere, endliche Menge ∑ von Buchstaben.
* Wörter: Eine geordnete, endliche Folge von Zeichen .
* Länge: ist die Anzahl der Buchstaben von .
* Häufigkeit: ist die Anzahl des Vorkommens von a in .
* Präfix: ist Präfix von , wenn gilt für ein Wort .
* Infix: ist Infix von , wenn gilt für die Wörter .
* Suffix: ist Suffix von , wenn gilt für ein Wort .
* Leeres Wort: bezeichnet das leere Wort.

**Operationen:** Man kann auf Wörtern Rechenoperationen anwenden und so neue Wörter erschaffen. Es gibt folgende Operationen:

* Konkatenation: Die Operation bezeichnet die Verkettung zweier Wörter und .
* Potenzen: bezeichnet die -fache Konkatenation von mit sich selbst.
* Inverses Wort: besteht aus den Buchstaben von in der umgekehrten Reihenfolge.

**Wortmengen und Sprachen:**

* bezeichnet die Menge aller Worte über dem Alphabet .
* Eine Teilmenge heißt formale Sprache über dem Alphabet .

**Operationen über Sprachen:** Auch über Sprachen lassen sich Rechenoperationen ausführen.

* Konkatenation: besteht aus allen Worten, die durch Konkatenation von zwei Wörtern aus und entstehen.
* Potenzen: bezeichnet die mehrfache Konkatenation der Sprache mit sich selbst.
* Kleenesche-Hülle: besteht aus allen Potenzen .
* Inverse Sprache: besteht aus allen inversen Worten von .

**Grammatik:** Eine Grammatik (Typ-0) ist ein Tupel bestehend aus:

* Einem Alphabet .
* Einer Menge von Variablen mit .
* Einer Menge von Produktionsregeln mit .
* Einer Startvariablen .

**Ableitungen:** Mithilfe von Produktionsregeln lassen sich Worte ableiten:

* Ableitungsschritt Die Anwendung einer Produktionsregel auf ein Wort .
* Ableitung Eine endliche Folge von Ableitungsschritten beginnend mit dem Wort .
* Sprache : Besteht aus allen Worten , die man durch eine Ableitung erzeugen kann.

**3. Reguläre Sprachen und endliche Automaten**

**Endlicher Automat:** Ein endlicher Automat besteht aus den folgenden Komponenten:

* Einem Eingabeband, das rechtseitig beschränkt ist. Der Automat kann sequenziell die Zeichen auf dem Band von links nach rechts lesen.
* Einem Zustandsspeicher S, der endlich viele Werte speichern kann. Der Automat kann lesend und schreibend auf den Speicher S zugreifen.
* Einer Übergangsfunktion . Der Automat berechnet mit aus dem aktuellen Wert im Speicher und dem aktuellen Eingabezeichen einen Folgezustand , der in gespeichert wird.

**Arbeitsweise eines Automaten:** Bei der Arbeitsweise eines Automaten gibt es drei Konfigurationen:

* Startkonfiguration: Zu Beginn ist im Speicher der Startzustand und auf dem Band das Eingabewort gespeichert. Der Lesekopf steht über dem ersten Zeichen des Eingabeworts .
* Folgekonfiguration: Der Automat liest den aktuellen Zustand in und das Zeichen unter dem Lesekopf und berechnet mit der Übergangsfunktion den Folgezustand , der in gespeichert wird. Der Lesekopf wechselt auf das nächste Zeichen .
* Endkonfiguration: Der Automat stoppt, wenn er das letzte Zeichen gelesen hat oder keinen Zustandsübergang machen kann. Der Automat akzeptiert das Wort, wenn er das Wort vollständig eingelesen hat und im Speicher ein Endzustand gespeichert ist.

**Zustandsdiagramme:** Endliche Automaten werden meist durch Zustandsdiagramme dargestellt. Dafür gibt es folgende Regeln:

* Ein Kreis im Diagramm stellt einen Zustand des Automaten dar.
* Ein Pfeil im Diagramm symbolisiert einen Zustandsübergang von einem Zustand in seinen Folgezustand. Das Eingabezeichen für den Zustandsübergang steht auf dem Pfeil.
* Der Startzustand wird durch einen eingehenden Pfeil ohne Ausgangszustand gekennzeichnet.
* Endzustände besitzen einen äußeren Kreis.

Bsp.:

Ein Bild, das Kreis, Diagramm, Reihe, Text enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Deterministischer endlicher Automat (DEA):** Ein DEA ist ein System . Hierbei ist:

* ein Eingabealphabet.
* eine endliche Menge von Zuständen.
* eine partielle Funktion der Zustandsübergänge.
* der Startzustand.
* die Menge der Startzustände.

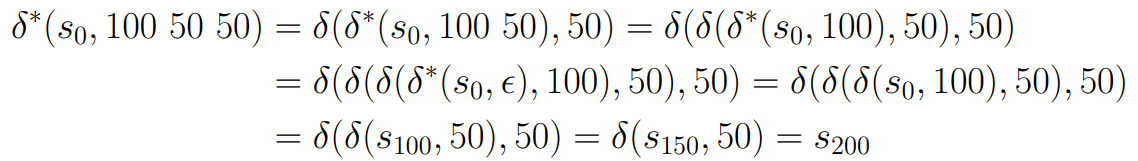
**Sprache des Automaten:** Automaten haben auch eine Sprache. Hierbei gilt:

* Ein Wort wird von dem Automaten akzeptiert, wenn er sich nach dem Einlesen von in einem Endzustand befindet.
* Die Sprache besteht aus allen Worten, die der Automat akzeptiert.
* Zwei Automaten und heißen äquivalent, wenn gilt.

**Erweiterte Übergangsfunktion:** Die erweiterte Übergangsfunktion berechnet den Zustandsübergang eines DEAs von einem Zustand in einen Zustand , wenn er ein Wort einliest. Der Folgezustand ergibt sich eindeutig aus der Übergangsfunktion durch die Folge der Zustandsübergänge der Eingabezeichen :

Die erweiterte Übergangsfunktion wird induktiv definiert durch:

Bsp.:



**Vollständige Automaten:** Ein Automat mit einer totalen Übergangsfunktion heißt vollständiger Automat.

**Vervollständigung:** Gegeben ist ein Automat mit einer partiellen Übergangsfunktion . Zur Vervollständigung wird ein neuer Zustand eingefügt. Nicht definierte Werte von werden in den neuen Zustand überführt.

Bsp.:

Ein Bild, das Text, Diagramm, Reihe, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Minimierung von Automaten:** Gegeben ist ein DEA, gesucht ist ein äquivalenter DEA mit einer minimalen Anzahl von Zuständen. Hierbei bestimmt man alle Zustände, welche nicht vom Startzustand erreichbar sind. Diese sind nutzlos und können entfernt werden.

**Äquivalenzrelationen:** Eine Relation auf einer Menge heißt Äquivalenzrelation, wenn sie folgende Eigenschaften erfüllt:

1. ist reflexiv, falls für alle gilt.
2. ist symmetrisch, falls aus stets folgt.
3. ist transitiv, falls und stets folgt.

Die Menge bezeichnet die Äquivalenzklasse von .

**Äquivalenzklassenautomat:** Vorgegeben sei ein vollständiger DEA ohne nutzlose Zustände. Zwei Zustände und sind äquivalent, falls für alle gilt: .

**Markierungsalgorithmus:** Der Markierungsalgorithmus berechnet die Äquivalenzklassen für den minimalen Äquivalenzklassenautomat. Er geht:

* Bilde eine Tabelle aller Zustandspaare mit und markiere alle Paare mit einem Endzustand und einem Nicht-Endzustand.
* Teste für jedes nicht markierte Zustandspaar und jedes Zeichen , ob die Folgezustände markiert sind. Falls ja, dann wird das Paar markiert.
* Wiederhole Schritt 2, bis es keine Änderungen mehr gibt.
* Nicht markierte Zustandspaare sind äquivalent und werden zusammengefasst.

**Reguläre Sprachen:** Als Reguläre Sprachen gelten Sprachen, welche von einem DEA akzeptiert werden.

**Pumping-Lemma:** Jede reguläre Sprache besitzt die Eigenschaft, dass man alle Worte ab einer bestimmten Wortlänge ‚aufpumpen‘ kann, d.h. es gibt ein Infix des Wortes, dass man beliebig oft wiederholen kann.

**Anwendung Pumping-Lemma:** Das Pumping-Lemma ist nur eine notwendige Bedingung für reguläre Sprachen. Man kann mit ihm daher nur prüfen, ob eine Sprache nicht regulär ist.

**Nichtdeterministische endliche Automaten:** Häufig ist die Konstruktion eines DEAs für eine reguläre Sprache schwierig und benötigt viele Zustände. Diese wird einfacher, wenn wir nichtdeterministische Zustandsübergänge zulassen. Der Automat hat jetzt beim Einlesen eines Zeichens zwei oder mehrere Möglichkeiten für einen Zustandsübergang und wählt nichtdeterministisch einen der Folgezustände.

**Epsilon-Übergänge:** Man kann für endliche Automaten auch Spontanübergänge (-Übergänge) zulassen. Dabei kann der Automat seinen Zustand wechseln, ohne dass er ein Eingabezeichen liest.

**NEA:** Ein nichtdeterministischer endlicher Automat (NEA) ist ein System bestehend aus:

* Einem Eingabealphabet .
* Einer endlichen Menge an Zuständen.
* Einer Übergangsfunktion , dabei ist die Potenzmenge von .
* Einem Startzustand .
* Einer Menge von Endzuständen .

**Erweiterte Übergangsfunktion:** Die -Hülle eines Zustandes ist die Menge aller Zustände, die der Automat ohne Einlesen eines Zeichens erreichen kann. Insbesondere gilt . Die erweiterte Übergangsfunktion eines Wortes wird induktiv definiert durch: .

**Äquivalenz von DEA und NEA:** Zu jedem NEA gibt es einen äquivalenten DEA.

**4. Reguläre Ausdrücke und Typ-3-Grammatiken**

**Reguläre Ausdrücke:** Reguläre Ausdrücke sind Muster zu Beschreibung von regulären Sprachen.

**Induktives Prinzip:** Elementare RA:

* Für jedes ist ein RA und definiert die Sprache .
* ist ein RA und definiert die Sprache .
* ist ein RA und definiert die Sprache .

Operatoren: Für zwei RA und definieren die folgenden Operationen einen RA:

* Selektion: definiert die Sprache .
* Konkatenation: definiert die Sprache .
* Wiederholung: definiert die Sprache .
* Klammern: definiert die Sprache .

**Äquivalenz zu endlichen Automaten:** Zu jedem endlichen Automaten gibt es einen regulären Ausdruck mit . Zu jedem regulären Ausdruck gibt es einen endlichen Automaten mit .

**Umwandlung eines Automaten:** Gegeben ist ein endlicher Automat Man kann voraussetzen, dass genau einen Endzustand hat, der vom Startzustand verschieden ist.

1. Entferne schrittweise alle Zustände bis auf und und ersetze die Beschriftungen durch RA.

Ein Bild, das Text, Kreis, Diagramm, Schrift enthält.

KI-generierte Inhalte können fehlerhaft sein.

1. Aus dem reduzierten Automaten mit den Zuständen und erhält man den regulären Ausdruck.

Ein Bild, das Schrift, Typografie, Text, Kalligrafie enthält.

KI-generierte Inhalte können fehlerhaft sein.

**Umwandlung eines RA:** Bei der Umwandlung eines RA in einen Automaten geht an genau andersherum vor.

1. Wir erstellen einen Automaten mit einem Startzustand und einem Endzustand , der einen Übergang von nach beschriftet mit dem RA besitzt.

Ein Bild, das Kreis, Diagramm enthält.

KI-generierte Inhalte können fehlerhaft sein.

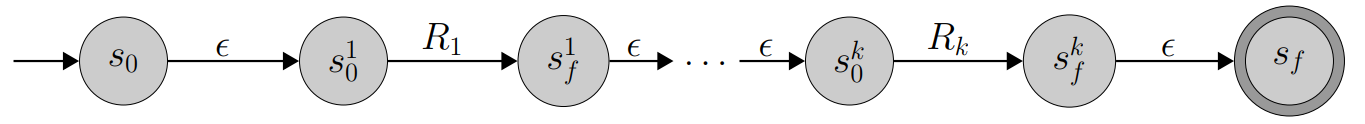
1. Die Operation im RA werden schrittweise nach ihrer Priorität aufgelöst und dabei der Automat wie folgt erweitert:

* Selektion : Ergänze für jeden RA zwei Zustände und , einen Übergang von  nach beschriftet mit und -Übergänge von nach und nach .

Ein Bild, das Kreis, Sport, Langhantel enthält.

KI-generierte Inhalte können fehlerhaft sein.

* Konkatenation : Ergänze für jeden RA zwei Zustände und , einen Übergang von nach beschriftet mit und -Übergängen von nach nach und nach .

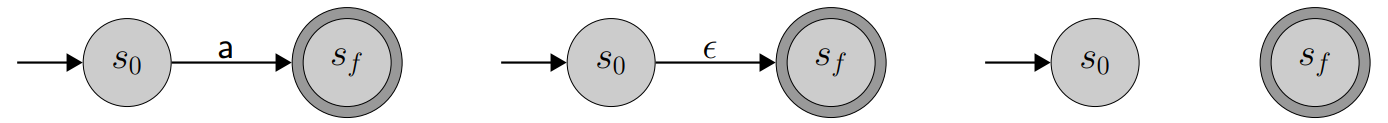


* : Ergänze zwei Zustände und , einen Übergang von nach beschriftet mit und -Übergänge von nach nach nach und nach .

Ein Bild, das Kreis, weiß, Antenne enthält.

KI-generierte Inhalte können fehlerhaft sein.

1. Die elementaren RA und werden durch das Terminalzeichen bzw. durch einen -Übergang ersetzt. Für den RA wird der Übergang entfernt.



**Abgeschlossenheit von regulären Sprachen:** Sind und zwei reguläre Sprachen über dem Alphabet , dann gilt:

1. Konkatenation: ist regulär.
2. Vereinigung: ist regulär.
3. Kleenesche Hülle: ist regulär.
4. Komplement: ist regulär.
5. Durchschnitt: ist regulär.
6. Differenz: ist regulär.
7. Inverse Sprache: ist regulär.
8. Die Sprachen und sind regulär.

Für Teilmengen von regulären Sprachen lassen sich keine Aussagen machen.

**Typ-3-Grammatiken:** Zu den Typ-3-Grammatiken gehören die rechts- und linkslinearen Grammatiken. Bei diesen Grammatiken werden in jedem Ableitungsschritt rechts bzw. links ein Terminalzeichen angehängt.

**Rechtslineare Grammatik:** Eine rechtslineare Grammatik besitzt die Produktionsregeln mit . Für die rechte Seite muss gelten:

1. mit .
2. mit und .
3. .

**Äquivalenz zu endlichen Automaten:** Zu jeder Typ-3-Grammatik gibt es einen endlichen Automaten, der akzeptiert. Zu jedem endlichen Automaten gibt es eine Typ-3-Grammatik, die erzeugt.

**Konstruktion eines Automaten:** Zu einer Typ-3-Grammatik wählen wir ein und definieren den Automaten mit den Zustandsübergängen:

* für Regeln vom Typ i.
* für Regeln vom Typ ii.
* für Regeln vom Typ iii.

**Konstruktion der Grammatik:** Zu einem DEA definieren wir die Grammatik mit

* Den Regeln für Zustandsübergänge .
* Den Regeln für Endzustände .

**Vereinfachung einer rechtslinearen Grammatik:** Jede rechtslineare Grammatik mit kann in eine äquivalente Grammatik umgewandelt werden, die nur Regeln vom Typ ii und eine -Regel besitzt. Gilt , dann wird zusätzlich noch eine -Regel für die Startvariable benötigt.

* Eine Regel der Form mit wird durch die folgenden Regeln ersetzt: .
* Regeln der Form für werden nach dem folgenden Verfahren ersetzt:

1. Es wird eine neue Variable und die Regel hinzugefügt.
2. Man bestimme zu die Regeln , in denen auf der rechten Seite vorkommt.
3. Ergänze für diese Regel eine neue Regel .

**Linkslineare Grammatiken:** In einer linkslinearen Grammatik sind die Regeln des Typen ii von der Form . Jede rechtslineare Grammatik kann in eine äquivalente linkslineare Grammatik transformiert werden und umgekehrt.

**Transformation in eine linkslineare Grammatik:** Gegeben ist eine rechtslineare Grammatik in der vereinfachten Form:

* Es gibt eine Regel mit .
* Es gibt ggf. eine -Regel für die Startvariable .
* Alle übrigen Regeln sind vom Typ ii.

Wir definieren zu die linkslineare Grammatik wie folgt:

* Die Regel wird zur Regel .
* Die Regel , sofern vorhanden, wird zur Regel .
* Jede Regel der Form wird zur Regel .

**Lineare Grammatiken:** Eine lineare Grammatik besitzt Produktionsregeln der Form mit

* mit .
* oder mit und .
* .