# 1. Reguläre Sprachen und endliche Automaten

**Einleitung**

Die Zerlegung eines Textes in Wörter (oder Tokens) heisst **scannen**.
Die Erkennung der aus den Tokens aufgebauten Strukturen heisst **parsen**.

**Kleene-Abschluss**

Darunter versteht man einfach den "*" bei regulären Ausdrücken

**Reguläre Ausdrücke**

- Ein regulärer Ausdruck besitzt gleich viele öffnende wie schliessende Klammern.
- Ein echtes Endstück eines regulären Ausdruckes ist entweder eine Folge von * oder sonst besitzt das Endstück mehr schliessende als öffnende Klammern.
- Ein echtes Endstück eines regulären Ausdruckes ist kein regulärer Ausdruck

Für einen regulären Ausdruck Φ gilt immer genau eine der folgenden Aussagen:

- Φ besteht aus genau einem Zeichen
- Φ ist von der Form (σρ).
- Φ ist von der Form (σ|ρ)
- Φ ist von der Form σ*.

**Reguläre Sprachen**

Eine Sprache, die durch einen regulären Ausdruck beschrieben werden kann, heisst reguläre Sprache.

**Deterministische endliche Automaten (DFAs)**

Mit Hilfe eines deterministischen endlichen Automaten, kann entschieden werden, ob ein String zu einer regulären Sprache gehört oder nicht.

Nachdem ein Stream von Symbolen aus einem Alphabet eingegeben wurde, steht der Automat entweder auf einem akzeptierenden (wahr), oder einem nicht akzeptierenden Zustand (falsch). Dabei besitzt der deterministische endliche Automat eine endliche Zahl von inneren Zuständen.

**Schwarzes Loch**

Ein Zustand Z eines Automaten heisst schwarzes Loch, wenn die beiden folgenden Eigenschaften erfüllt sind:

- Z ist **kein** akzeptierender Zustand
- Alle Symbole im Alphabet führt direkt wieder zurück auf Z

**Nichtdeterministische endliche Automaten (NFAs)**

Diese Art von Automaten können Pfeile von einem Zustand auf sich selbst aufweisen. Ansonsten gibt es keine Änderungen gegenüber den deterministischen Automaten.
Anders ausgedrückt kann nicht immer bereits im Voraus determiniert werden, welches der nächste Zustand sein muss.

**Nichtdeterministische endliche ϵ-Automaten**

Diese Art von Automaten entspricht den nichtdeterministischen endlichen Automaten, jedoch sind so genannte ϵ-Übergänge möglich. Dies sind Übergänge, ohne dass ein Input nötig ist.
Mit anderen Wort wird als Input das Alphabet Σ und zusätzlich noch ϵ akzeptiert.

**Umwandlung eines nichtdeterministischen Automaten**

Jeder nichtdeterministische Automat kann in einen deterministischen umgewandelt werden, indem die Übergangsfunktion auf die jeweilige Menge der möglichen Zustände angewandt wird.

**Abschlusseigenschaften**

Beweis, dass die Menge aller Sprachen, die von endlichen Automaten erkannt werden, abgeschlossen ist unter Booleschen Operation sowie unter Konkatenation und Kleene-Abschluss.

**Produktautomaten**

Ein Produktautomat fügt die jeweiligen Zustände zweier Automaten zusammen.
Ein UND-Produktautomat von A1 und A2 erkennt den Durchschnitt L1 ∩ L2.
Ein ODER-Produktautomat von A1 und A2 erkennt die Vereinigung L1 U L2.
Ein XOR-Produktautomat erkennt die Wörter, die zu einer der Sprachen L1 **oder** L2 gehören, zu der anderen aber nicht.

**Äquivalenz von regulären Sprachen und endlichen Automaten**

Jede reguläre Sprache wird von einem endlichen Automaten erkannt.
Siehe auch Abschlusseigenschaften.

**Erweiterte nichtdeterministische endliche Automaten**

Ein solcher Automat liegt vor wenn:

- Kein Pfeil im Startzustand endet
- Es gibt genau einen akzeptierenden Zustand und es existiert kein Pfeil, der bei diesem Endzustand startet
- Der Startzustand ist nicht der akzeptierende Zustand
- Es gibt genau einen Pfeil zwischen dem Startzustand und dem Endzustand

Diese vier Bedingungen sind keine wirklichen Einschränkungen.
Jede verletzte Bedingung kann durch geeignete Massnahmen umgangen werden.

# 2. Kontextfreie Sprachen und Pushdown-Automaten

**Einführung**

Mit regulären Sprachen lassen sich keine geschachtelten Konstrukte beschreiben. Dies ist jedoch notwendig zu Beschreibung von Programmiersprachen.
Zu diesem Zweck gibt es die kontextfreien Sprachen. Im Gegensatz zu den regulären Sprachen steht hier nicht die Frage im Vordergrund, ob ein Wort zur Sprache gehört, sondern man interessiert sich für die Struktur des Wortes. Die Erkennung dieser Strukturen heisst parsen.
Kontextfreie Sprachen werden durch kontextfreie Grammatiken, oder durch so genannte Pushdown Automaten beschrieben.
Kontextfreie Sprachen sind eine Erweiterung der regulären Sprachen.

**Kontextfreie Grammatiken (CFGs)**

Eine kontextfreie Grammatik ist eine Struktur, die aus vier Komponenten besteht:

- Σ ist eine endliche Menge, die Menge der Terminale (Blätter)
- N ist eine endliche Menge, die Menge der Nichtterminale (Knoten)
- S E von N ist das Startsymbol
- P ist eine endliche Menge von Produktionen. Dies ist ein Paar, bestehend aus einem Nichtterminal und einem String R, der aus Terminalen und Nichtterminalen besteht.
  Eine Produktion wird mit L → R bezeichnet.

Dabei ist zu beachten, dass ein Terminal kein Nichtterminal sein kann, oder umgekehrt.
Eine Sprache L heisst kontextfrei, wenn L die Menge aller Wörter ist, die von einer kontextfreien Grammatik erzeugt werden.
Was man jedoch mit kontextfreien Grammatiken nicht erreicht, ist die Kontrolle, ob eine Variable bereits deklariert wurde oder nicht. Dafür benötigt man Kontextsensitive Grammatiken.

**Ableitung**

Möglichkeit um die von einer kontextfreien Grammatik erzeugten Wörter zu beschreiben. Dabei wird jeweils vom Startzustand ausgehend immer eine Produktion auf einem Nichtterminal angewandt, bis nur noch lauter Terminale, spricht Buchstaben aus dem vorgegebenen Alphabet vorhanden sind.

**Parsbäume**

Dies ist eine weitere Möglichkeit, die von einer kontextfreien Grammatik erzeugten Wörter zu beschreiben. Das Vorgehen dabei ist gleich wie bei der Ableitung, nur wird das ganze als Baum dargestellt. Die dabei einzuhaltenden Regeln sind von der CFG gegeben.

**Reguläre Ausdrücke und kontextfreie Grammatiken**

Zu jedem regulären Ausdruck, gibt es eine kontextfreie Grammatik, die dieselbe Sprache erzeugt.
Um die Umformung vorzunehmen, geht man rekursiv von unten nach oben vor und definiert jeweils neue Nichtterminale, welche dann in Terminale oder die neuen Nichtterminale übergehen, bis man nur noch Übergänge in Terminale hat.

**Endliche Automaten und kontextfreie Grammatiken**

Zu jedem endlichen Automaten kann eine kontextfreie Grammatik konstruiert werden, die die gleiche Sprache erzeugt.

Die Umwandlung ist einfach:

- Das Alphabet wird zur Menge der Terminale
- Die Menge Q der Zustände wird zur Menge der Nichtterminal
- Der Startzustand S ist auch Startsymbol der Grammatik
- Für einen Pfeil von q nach r mit der Beschriftung a, besitzt die Grammatik die Produktion q->ar.
- Für einen ϵ-Pfeil von q nach r besitzt die Grammatik die Produktion q->r
- Zu jedem Endzustand gibt es eine Produktion f -> ϵ

Wird nach der Umwandlung jeweils der Automat mit einem beliebigen Wort aus dem Alphabet durchlaufen und dabei bei jedem Schritt die entsprechenden Produktion angewendet, erhält man gerade die Ableitung, bzw. den Parsebaum für das jeweilige Wort.

**Mehrdeutigkeit**

Eine Mehrdeutigkeit liegt dann vor, wenn eine kontextfreie Grammatik G ein Wort erzeugt, das zwei (oder mehr) verschiedene Parsebäume besitzt. Unterschiedliche Ableitungen sind nicht gleichbedeutend mit einer Mehrdeutigkeit.

**Beseitigung der Mehrdeutigkeit**

Dies wird erreicht durch Operatoren-Präzedenz, oder via Strukturerhöhung. Es gibt jedoch kontextfreie Sprachen, welchen nur mehrdeutige kontextfreie Grammatiken zugrunde liegen. Eine solche Sprache nennt man inhärent mehrdeutig.