NATÜRLICHE SPRACHEN

Natürliche Sprachen legen ihre Struktur durch

- → die Regeln einer **Grammatik**
- → und eine Menge von **erlaubten Worten** ($\triangleq$ Strings gebildet aus Buchstaben)

fest.

Allerdings müssen syntaktisch korrekte Sätze einer natürlichen Sprache keinen Sinn tragen:

- → Wiesbaden wohnt weiterhin weich
- → Der bissige Student jagt die verschlafene Mensa

$\Rightarrow$ syntakisch korrekte Sätze müssen keinen Sinn ($\triangleq$ Semantik) tragen.

Wie kann man diese Beobachtungen in der Informatik ausnutzen?

FORMALE REGELN ZUR ERZEUGUNG EINER SPRACHE

Der Linguist Noam Chomsky hatte folgende Idee:

Korrekte Sätze einer (natürlichen) Sprache sollen durch ein **(endliches System) von formalen Regeln** erzeugt werden.

Bis heute ist diese Idee

→ in der Linguistik umstritten, aber

→ extrem bedeutsam in der Informatik.

Basis für z.B. alle Programmiersprachen / Compilerbau, Auszeichnungssprachen (SGML, XML, HTML, . . .).

Ähnlich sind die sogenannten **(Semi) Thue Systeme**, die heute z.B. in Spezialformen in der Computergraphik Bedeutung erlangt haben.

EINIGE GRUNDLEGENDE BEGRIFFE

Eine endliche Menge Σ heißt **Alphabet**. Die **Elemente** von Σ werden **Buchstaben** genannt. Eine Folge von Buchstaben nennt man **Wort** (über Σ). Eine beliebige Menge von Worten über Σ nennt man dann eine **(formale) Sprache**.

Beispiel (arithmetische Ausdrücke)

Sei $\Sigma = \{), (, +, -, *, /, x\}$ und EXPR die Menge aller korrekten arithmetischen Ausdrücke. Damit gilt

- $\rightarrow$ $(x - x) \in$ EXPR
- $\rightarrow$ $((x + x) * x)/x \in$ EXPR
- $\rightarrow$ $))(x-) * x \notin$ EXPR

EXPR ist eine Menge von Worten über Σ, also kann man EXPR als **formale Sprache** (über $\{), (, +, -, *, /, x\}$) bezeichnen.

WEITERE BEISPIELE FÜR FORMALE SPRACHEN (II))

Beispiel (Wortmengen über $\{a, b\}$)

Sei $\Sigma = \{a, b\}$, dann sind die folgenden Mengen auch formale Sprachen über Σ:

- BRACKET $= \{ab, aabb, aaabbb, aaaabbbb, \ldots\}$
- UODD $= \{a, aaa, aaaaa, aaaaaaa, aaaaaaaaa, \ldots\}$
- $\Sigma^* =$ ALL $= \{\epsilon, a, b, aa, ab, ba, bb, aaa, aab, aba, abb, baa, \ldots\}$

GRAMMATIKEN UND AUTOMATEN

(Formale) Sprachen enthalten meist unendlich viele Wörter

→ Wir brauchen endlich viele **Erzeugungsregeln**, um (algorithmisch) mit formalen Sprachen umgehen zu können. Die Rolle der Regeln übernehmen **Grammatiken**.

→ Weiterhin werden **Erkenner** benötigt, die entscheiden, ob ein **Wort zu einer Sprache gehört**. Die Rolle der Erkenner spielen die **Automaten**, die wir in dieser Vorlesung studieren.

TEIL EINER NATÜRLICHEN SPRACHE

Beispiel (Eine Grammatik)

$$
\begin{aligned}
\textbf{Satz} &\rightarrow \textbf{Subjekt Prädikat Objekt} \\
\textbf{Subjekt} &\rightarrow \textbf{Artikel Attribut Substantiv} \\
\textbf{Artikel} &\rightarrow \epsilon \mid \text{der} \mid \text{die} \mid \text{das} \\
\textbf{Attribut} &\rightarrow \epsilon \\
\textbf{Attribut} &\rightarrow \textbf{Adjektiv} \\
\textbf{Attribut} &\rightarrow \textbf{Adjektiv Attribut} \\
\textbf{Adjektiv} &\rightarrow \text{kleine} \mid \text{bissige} \mid \text{verschlafene} \\
\textbf{Substantiv} &\rightarrow \text{Student} \mid \text{Katze} \\
\textbf{Prädikat} &\rightarrow \text{jagt} \mid \text{betritt} \\
\textbf{Objekt} &\rightarrow \textbf{Artikel Attribut Substantiv}
\end{aligned}
$$

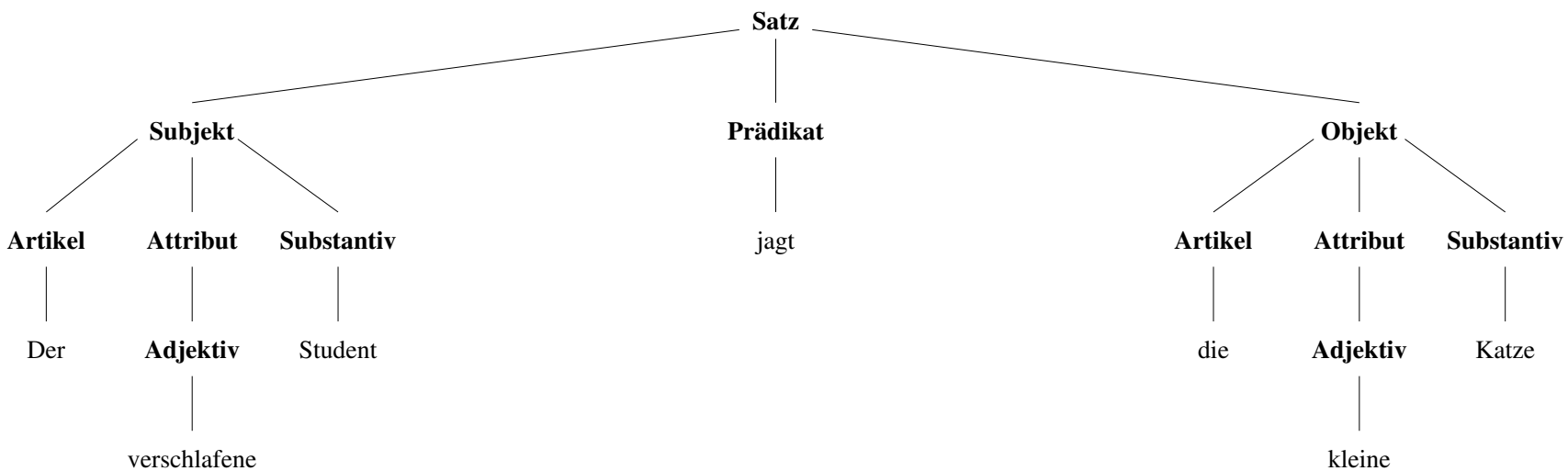
Das Symbol „|“ markiert eine Alternative, d.h. $\mathbf{A} \rightarrow \mathbf{B} \mid \mathbf{C}$ ist Abkürzung für die beiden Regeln $\mathbf{A} \rightarrow \mathbf{B}$ und $\mathbf{A} \rightarrow \mathbf{C}$

TEIL EINER NATÜRLICHEN SPRACHE (II)

Durch Anwendung der Regeln und Ersetzung der fett gedruckten Wörter können z.B. die folgenden Sätze gebildet werden:

→ Der kleine bissige Student betritt die verschlafene Mensa
→ Der verschlafene Student jagt die kleine Katze

Mit **Syntaxbäumen** man man die **Ableitungschritte graphisch** verdeutlichen:



L-SYSTEME

- → Die **L-Systeme** wurden 1968 durch **Aristid Lindenmeyer** als mathematisches Modell des Pflanzenwachstums eingeführt.
- → L-Systeme werden heute in der Computergraphik benutzt, um natürlich wirkende Pflanzen schnell generieren zu können.
- → Hier betrachten wir die einfachste Klasse von L-Systemen, die so genannten DOL-System.
  - → Die Regeln sind deterministisch, d.h. für jeden Buchstaben gibt es **genau eine Regel**.
  - → Die Regeln sind kontextfrei, d.h. **Ersetzungen hängen nicht** von den **umgebenden Buchstaben** (≙ Kontext) ab.

GRUNDLEGENDE BEGRIFFE UND EIGENSCHAFTEN

Definition (0L-Systeme)

- Mit Σ^* bezeichnen wir die Menge **aller Wörter** über Σ.
- Ein **0L-System** G ist ein Tripel $G = (\Sigma, \omega, P)$, wobei
  - Σ das **Alphabet**, ω das **Axiom** und
  - $P \subseteq \Sigma \times \Sigma^*$ die Menge der **Produktionen**.
- Eine Produktion $(a, \chi) \in P$ wird als $a \to \chi$ geschrieben. Der Buchstabe a heißt **Vorgänger** und χ **Nachfolger** dieser Produktion.
- Für **jeden** Buchstaben $a \in \Sigma$ **existiert eine Produktion** $(a, \chi) \in P$.
- Ein $0L$-System heißt **deterministisch**, wenn es für jeden Buchstaben $a \in \Sigma$ nur **genau eine** Produktion $(a, \chi) \in P$ gibt.

DOL-SYSTEME (II)

Definition

Deterministische OL-Systeme heißen **DOL**-Systeme.

Definition (Ableitung)

Sei $\mu = a_1 \dots a_m$ ein beliebiges Wort über Σ, dann kann $\nu = \chi_1 \dots \chi_m$ aus μ **abgeleitet** werden, wenn

- → **für alle** $i = 1, \dots, m$ $(a_i, \chi_i) \in P$ gilt, wobei
- → man $\mu \vdash\!\!\!- \nu$ schreibt.
- → Ein Wort ν heißt **von** G **generiert**, wenn es in **endlich** vielen Schritten aus dem Axiom abgeleitet werden kann.

DOL-SYSTEME (III)

Geben wir aus **Bequemlichkeitsgründen** für einen Buchstaben a keine Produktion an, dann gilt **implizit** $(a, a) \in P$.

Achtung: Alle Regeln aus P werden **gleichzeitig** angewendet.

Wird ein Wort ν von $G = (\Sigma, \omega, P)$ generiert, dann können wir also

$$\omega \vdash \mu_1 \vdash \mu_2 \vdash \ldots \vdash \mu_n = \nu$$

schreiben (kurz: $\omega \overset{*}{\vdash} \nu$).

EIN BEISPIEL

Sei $G = (\Sigma, \omega, P)$, wobei

- $\Sigma = \{a, b, c\}$,
- $\omega = abc$ und
- $P = \{a \to aa, b \to bb, c \to cc\}$.

Mit Hilfe dieses DOL-Systems können Worte der Form

$$a^{2^n} b^{2^n} c^{2^n}$$

für $n \geq 0$ abgeleitet werden.

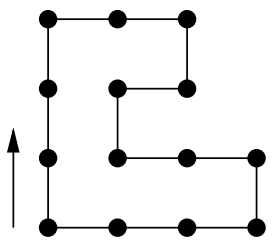
Bemerkung: a^n ist die Abkürzung für $\underbrace{aaa \ldots a}_{n-\text{mal}}$

TURTLE-GRAPHIK

Sei δ **ein beliebiger Winkel**, dann werden die Buchstaben `F`, `f`, $+$ und $-$ wie folgt interpretiert:

`F`	**Bewege** den **Stift** um die Länge d und **zeichne** eine **Linie**
`f`	**Bewege** den **Stift** um die Länge d und **zeichne keine Linie**
$-$	**drehe** um δ Grad nach **rechts**
$+$	**drehe** um δ Grad nach **links**

Mit $\delta = 90°$ wird `FFF-FF-F-F+F+FF-F-FFF` in die Graphik



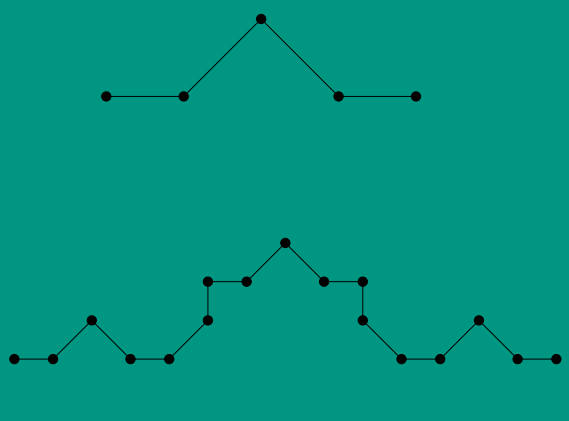
umgesetzt.

EIN BEISPIEL

Beispiel (Kochsche Schneeflocke)

Gegeben sei $G = (\Sigma, \omega, P)$ mit Alphabet $\Sigma = \{\mathtt{F}, +, -\}$, Axiom $\omega = \mathtt{F}$ und der Menge der Produktionen $\{\mathtt{F} \to \mathtt{F}+\mathtt{F}--\mathtt{F}+\mathtt{F}\}$
Wir legen $\delta = 45°$ fest. Für die Anzahl der Schritte n ergibt sich:

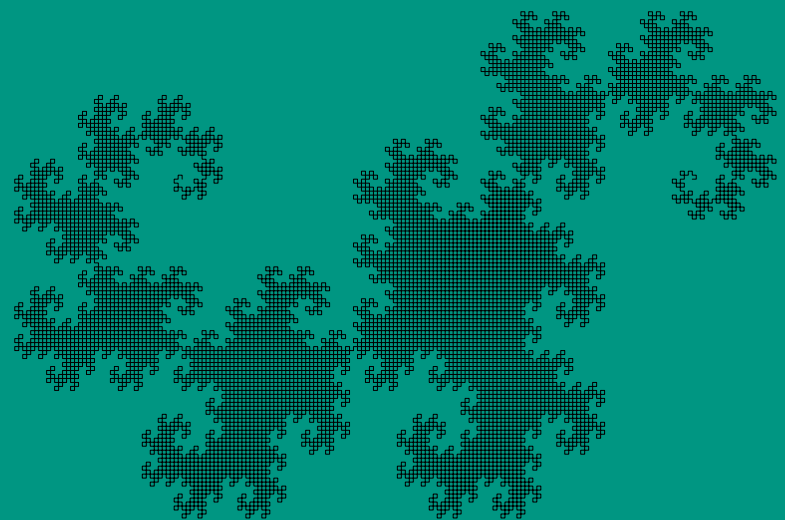
$n = 1$	$\mathtt{F}+\mathtt{F}--\mathtt{F}+\mathtt{F}$
$n = 2$	$\mathtt{F}+\mathtt{F}--\mathtt{F}+\mathtt{F}+\mathtt{F}+\mathtt{F}$ $--\mathtt{F}+\mathtt{F}--\mathtt{F}+\mathtt{F}--$ $\mathtt{F}+\mathtt{F}+\mathtt{F}+\mathtt{F}--\mathtt{F}+\mathtt{F}$



EIN ZWEITES BEISPIEL

Beispiel (Drachenkurve)

Sei $\delta = 90°$ und das L-System $G = (\{\mathtt{F_r}, \mathtt{F_l}, +, -\}, \mathtt{F_l}, \{\mathtt{F_l} \to \mathtt{F_l} + \mathtt{F_r}+, \mathtt{F_r} \to -\mathtt{F_l} - \mathtt{F_r}\})$, dann ergibt sich



Sowohl $\mathtt{F_l}$ als auch $\mathtt{F_r}$ werden als „Bewege den Stift einen Schritt der Länge d und zeichne eine Linie“ interpretiert.