Pflanzenwachstum als formale Sprache und Anwendung in der Computergrafik

Was sind Lindenmayer-Systeme?

$$L(G) = \{a^n b^n | n \geq 1\}$$

- Entwickelt von Aristid Lindenmayer im Jahre 1968
- Darstellung von Pflanzen und deren Wachstumsprozesse als mathematischen Formalismus
- Grundlagen:
  - Pflanzen DNA
  - Eigenschaften von Zellen und Veränderungsprozesse
  - Formale Sprachen

Was sind Lindenmayer-Systeme?

- Ein Organismus O entwickelt sich aus einer Urzelle
- Die Urzelle hat die gesamte Information für jegliche Wachstumsprozesse und mögliche Zellstrukturen von O (DNA)
- Aus der Urzelle entwickeln sich durch Wachstum, Zellteilung, Transformation, Absterben etc.

$O = \{\mathrm{V}, Urzelle, P\}, wobei\ P = \{Zellteilung, Wachstum, Absterben\}$
$und\ \mathrm{V} = \{\mathrm{Zellen}\}$

Was sind Lindenmayer-Systeme?

Definition eines 0L-Systems Tripel G = (V, w, P)

- V als endliches Alphabet des Systems
- $\mathbf{w} \in V^+$ als nicht leeres Wort (Axiom)
- $P \subset V \times V^*$ als endliches Set an Produktionen in der Form $(a \to \beta)$
- Ist für eine Variable kein P festgelegt → Identitätsproduktion $(a \to a)$
- Gilt: $\mathbf{jedes}\ a \in V\ mit\ exat\ einer\ a \to \beta, mit\ \beta \in V^*$ → deterministisch
- Ein 0L-System berücksichtigt keinen Input zwischen den Zellen
- Parallele Anwendung der Produktionen auf die Zeichenkette

0L-Systeme und Chomsky Sprachtypen

0L-Systeme

- Endliches Alphabet Σ
- Produktionen: $A \rightarrow \alpha$
- Restriktionen: $A \in V, \alpha \in (V \cap \Sigma)^*$
- Kontext: Kontextfrei

Formale Sprache (Typ-2)

- Endliches Alphabet Σ
- Produktionen: $A \rightarrow \gamma$
- Restriktionen: $A \in V, \gamma \in (V \cap \Sigma)^*$
- Kontext: Kontextfrei

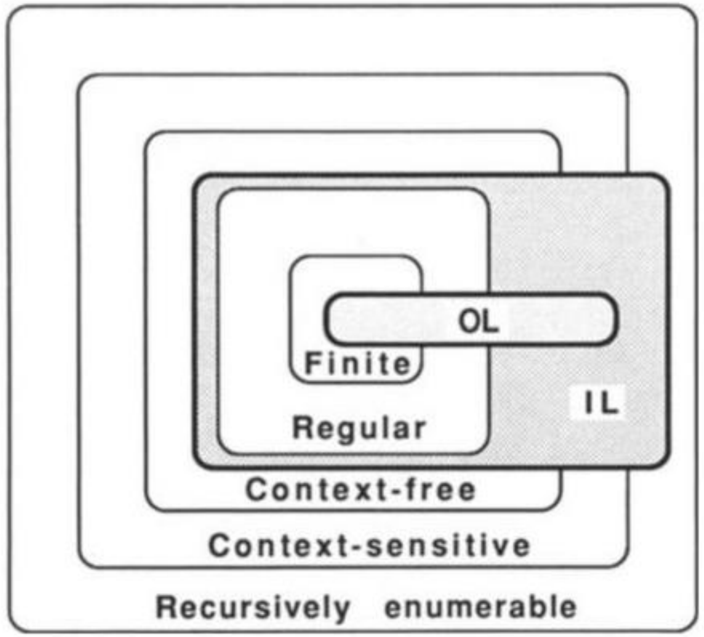
Unterschiede:

- Parallele Anwendung der Produktionen auf die gegebene Zeichenkette
- 0L-Systeme verwenden keine Terminale; Theoretisch ist die Zeichenkette nie abgeschlossen

0L-Systeme und Chomsky Sprachtypen

0L-Systeme

- Endliches Alphabet Σ
- Produktionen: $A \rightarrow \alpha$
- Restriktionen: $A \in V, \alpha \in (V \cap \Sigma)^*$
- Kontext: Kontextfrei



Unterschiede:

- Parallele Anwendung der Produktionen auf die gegebene Zeichenkette
- 0L-Systeme verwenden keine Terminale; Theoretisch ist die Zeichenkette nie abgeschlossen

Wie funktionieren 0L-Systeme? Beispiel

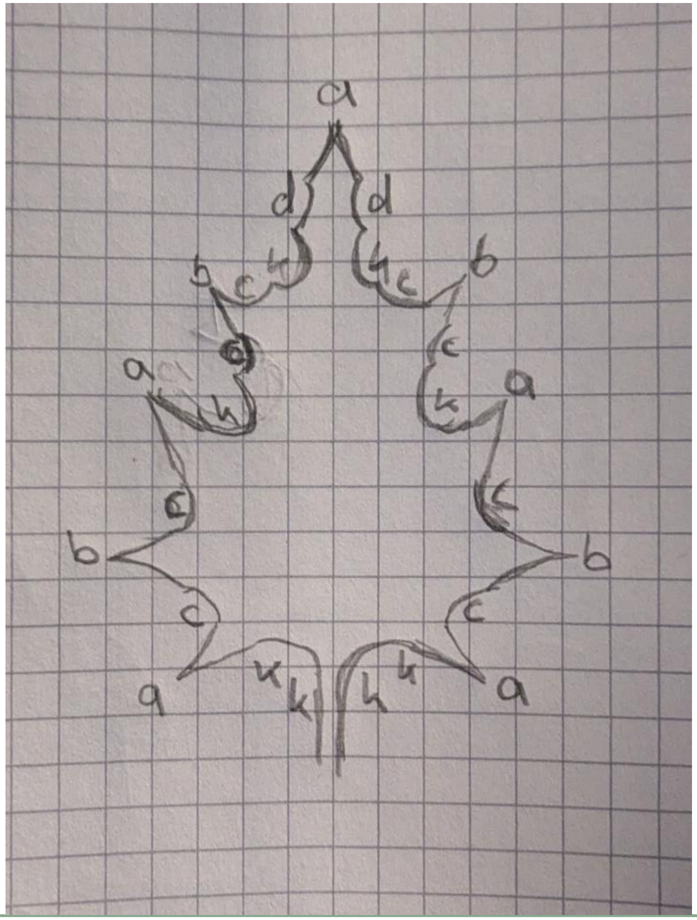
Gegeben ist:

- Alphabet {a,b,c,d,k}
- Produktionen:
  $\{(a \rightarrow cdc), (b \rightarrow dad),$
  $(c \rightarrow k), (d \rightarrow a), (k \rightarrow k)\}$
- Axiom: a

→ Erzeugt eine Zeichenkette; Länge abhängig von der #Generationen

Generation	Wort w
1	a
2	cbc
3	kdadk
4	kacbcak
5	kcbckdadkcbck
6	kkdadkkacbcakkdadkk
7	kkacbcakkcbckdadkcbckkacbcakk
8	kkcbckdadkcbckkkdadkkacbcakkcbckkkcbckdadkcbckk

Wie funktionieren 0L-Systeme? Beispiel

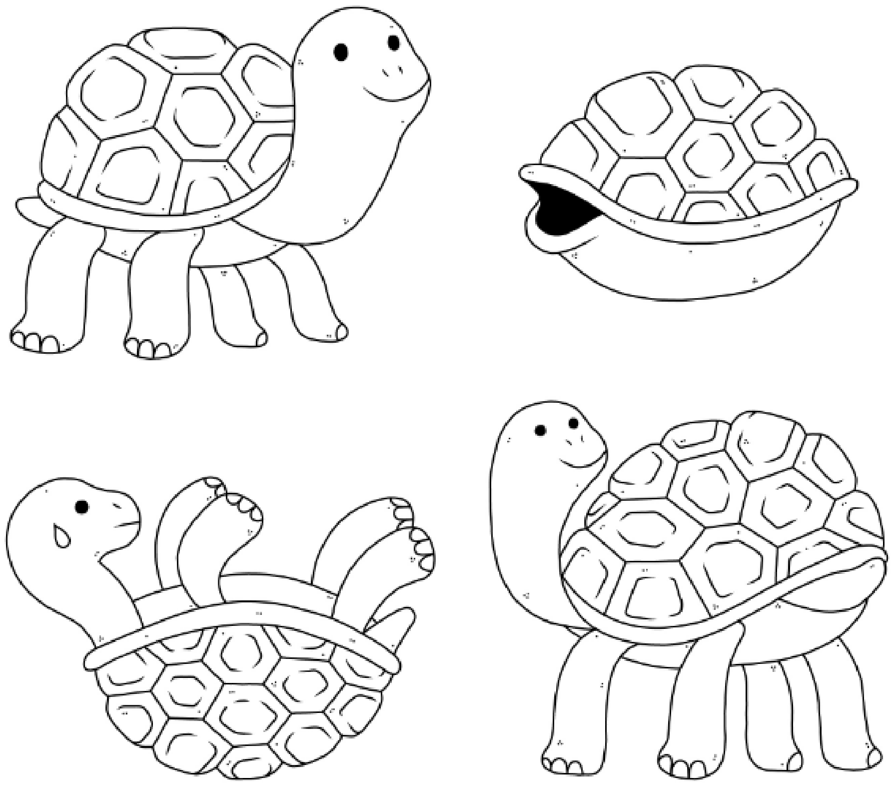


7	kkacbcakkcbckdadkcbckkacbcakk

Visuelle Interpretation:

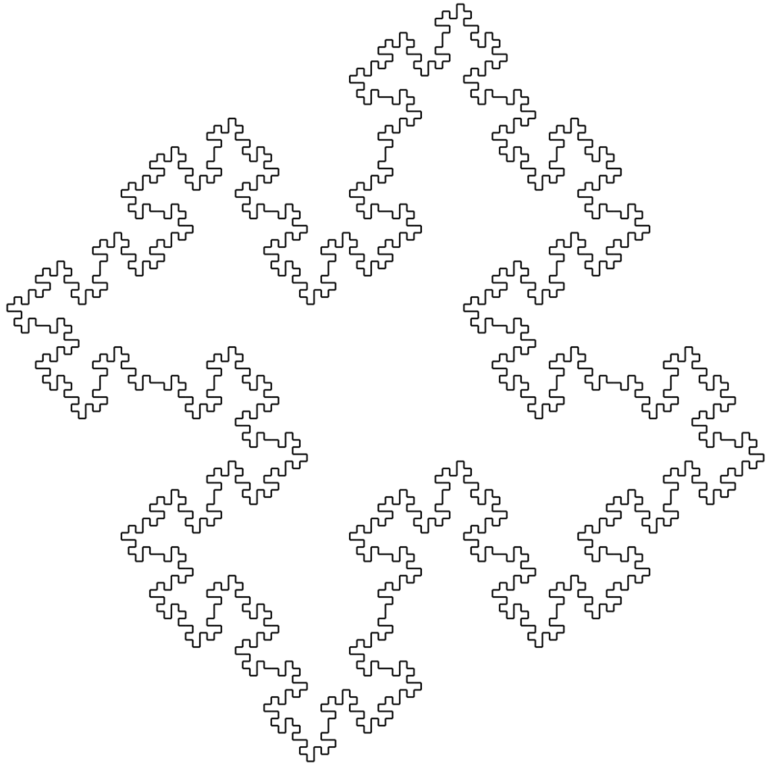
- a,b: spitz zulaufende Dächer
- c,d: leicht nach innen gewölbten Linien
- k: Starke Vertiefungen nach Innen
- Je öfter ein Buchstabe hintereinander, desto ausgeprägter

Computergrafik: Turtle Interpretation



- Standardisieren der Interpretationsmöglichkeiten
- Turtle ist durch einen Zustand (x, y, α) beschrieben
- X,y sind kartesische Koordinaten und geben die Position der Turtle an
- α ist der Blickwinkel der Turtle und beschreibt die Richtung
- d ist die Schrittweite
- Winkel-Inkrement δ für Drehungen

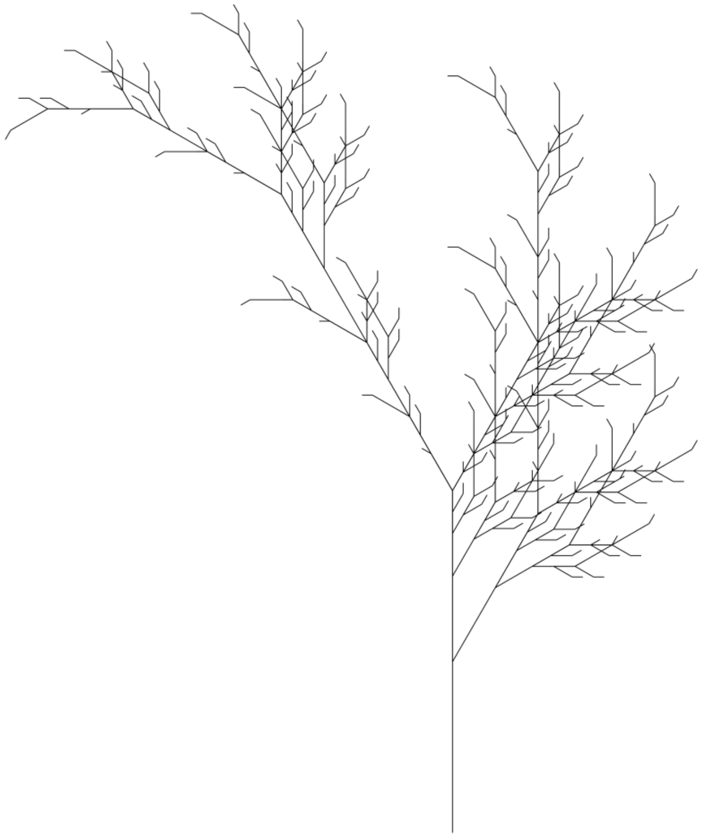
Computergrafik: Turtle Interpretation



- F = Bewegung nach vorne der Länge d. Eine Linie zwischen (x,y) und (x‘,y‘) wird gezeichnet
- f Bewegung nach vorne der Länge d ohne zeichnen einer Linie
- + Drehung nach links um den Winkel δ
- - Drehung nach rechts um den Winkel δ

0L-System Definition:
Axiom: F-F-F-F
P: F → F-F+F+FF-F-F+F
α = 0°
δ = 90°

Computergrafik: Turtle Interpretation Erweiterungen



Verzweigung (Segmentierung der Zeichenkette)

Verbindung der Turtle mit einem Pushdown-Stack

- [= Schreibe den aktuellen Zustand in den Stack
-] = Lese den letzten Eintrag im Stack, Setze diesen Zustand als aktuellen Zustand und zeichne keine Linien

Stack ermöglich auch:

- Farben, dicke der Linie

0L-System Definition:

Axiom: X

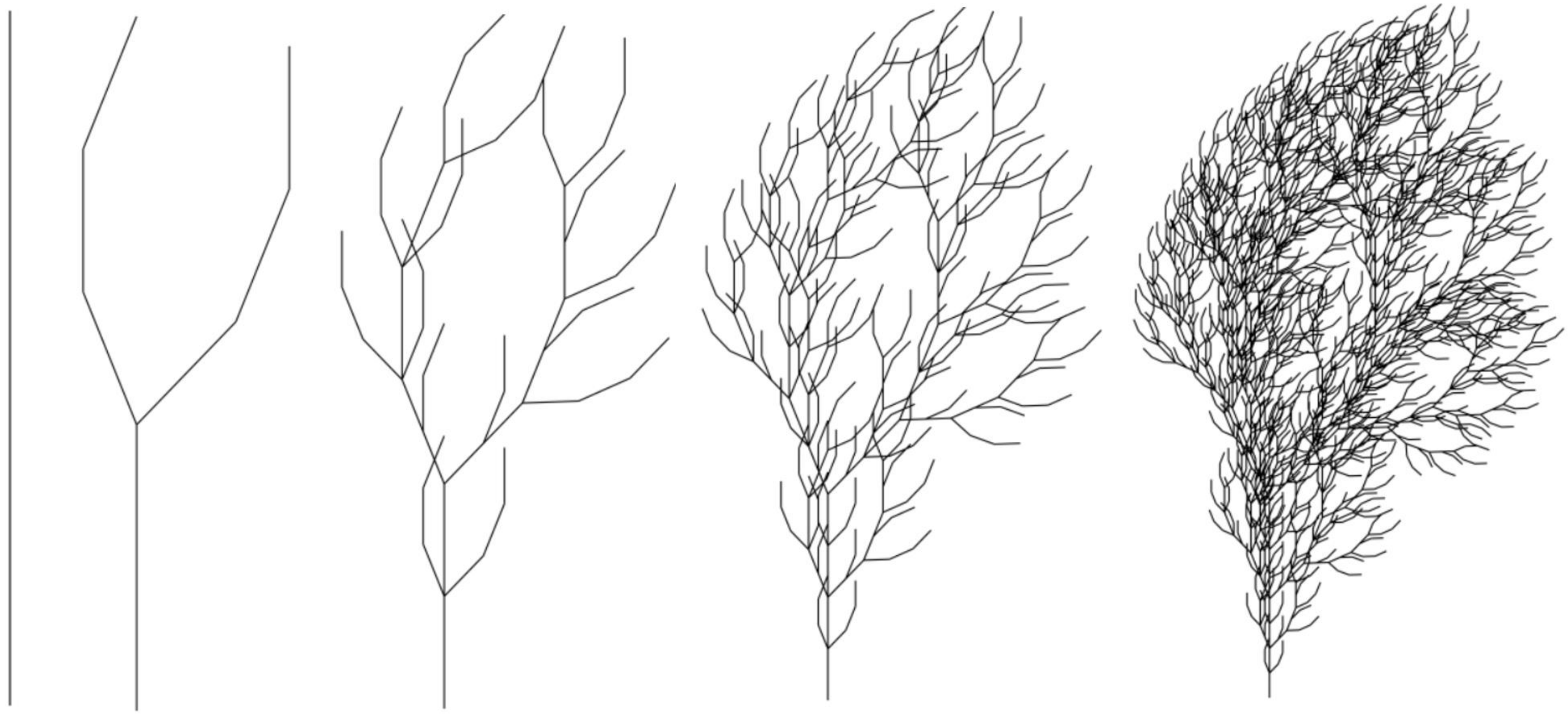
X → F[-[X]+X]F[+FX]-X

F → FF

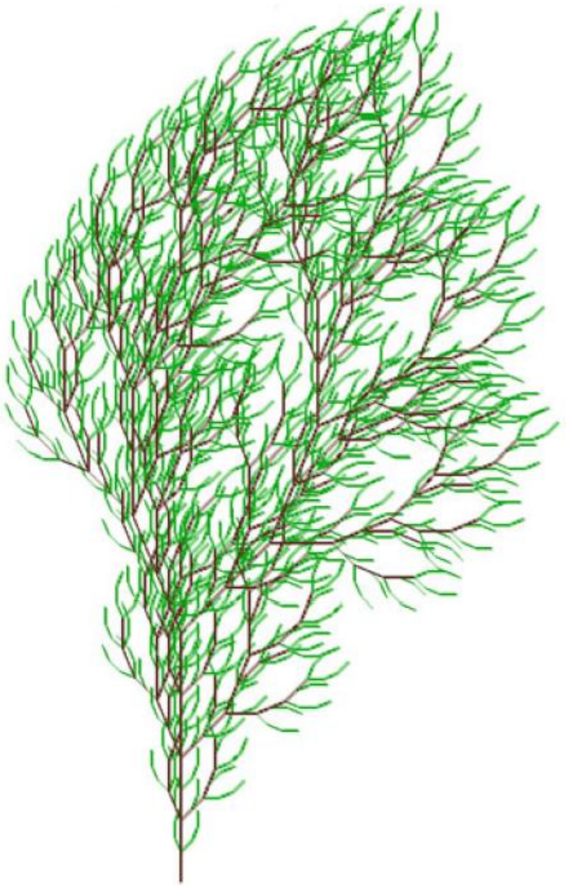
α = 90°

δ = 30°

Computergrafik: Turtle Interpretation Erweiterungen



Computergrafik: Turtle Interpretation Erweiterungen



Farbkodierung

- Kleinbuchstaben als Farbcode in der Interpretation hinterlegen
- Bsp. n = braun, g = grün

0L-System Definition:
Axiom: gF
F → nFF-g[-F+F+F]+g[+F-F-F]
α = 90°
δ = 22.5°

Computergrafik: Turtle Interpretation Erweiterungen

$$R_U(\alpha) = \begin{bmatrix} \cos\cdot\alpha & \sin\cdot\alpha & 0 \\ -\sin\cdot\alpha & \cos\cdot\alpha & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$R_L(\alpha) = \begin{bmatrix} \cos\cdot\alpha & 0 & -\sin\cdot\alpha \\ 0 & 1 & 0 \\ \sin\cdot\alpha & 0 & \cos\cdot\alpha \end{bmatrix}$$

$$R_H(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & \cos\cdot\alpha & -\sin\cdot\alpha \\ 0 & \sin\cdot\alpha & \cos\cdot\alpha \end{bmatrix}$$

Dreidimensionaler Raum

Verwenden von Rotationsmatrizen zur Drehung

- + Linksdrehung um den Winkel δ Rotationsmatrix $R_U(\delta)$
- - Rechtsdrehung um den Winkel δ Rotationsmatrix $R_U(-\delta)$
- & Höhe nach unten verschieben Rotationsmatrix $R_L(\delta)$
- ∧ Höhe nach unten verschieben Rotationsmatrix $R_L(-\delta)$
- \ Bewegung nach links; Rotationsmatrix $R_H(\delta)$
- / Bewegung nach rechts; Rotationsmatrix $R_H(-\delta)$
- | Drehung um 180° $R_U(180)$

Computergrafik: Turtle Interpretation Erweiterungen

$n=5, \delta=18°$

ω : **plant**
p_1 : **plant** → **internode** + [**plant** + **flower**] − − //
[− − **leaf**] **internode** [+ + **leaf**] −
[**plant flower**] + + **plant flower**
p_2 : **internode** → F **seg** [// & & **leaf**] [// ∧ ∧ **leaf**] F **seg**
p_3 : **seg** → **seg** F **seg**
p_4 : **leaf** → [' { +f−ff−f+ | +f−ff−f }]
p_5 : **flower** → [& & & **pedicel** ‘ / **wedge** //// **wedge** ////
wedge //// **wedge** //// **wedge**]
p_6 : **pedicel** → FF
p_7 : **wedge** → [‘ ∧ F] [{ & & & & −f+f | −f+f }]

Verschiedene Arten von L-Systemen

- <m,n>L-Systeme oder parametrische L-Systeme sind in der Lage kontextabhängig zu agieren
- Dadurch wird unter anderem Zellsterben und Terminierung von Zeichenketten möglich
- Komplexer und nicht mehr mit Typ-2 Sprachen vereinbar
- War nicht Bestandteil der Untersuchungen

Fazit

- Findet Anwendung im wissenschaftlichen Bereich für computergestützte Botanik
- Berücksichtigt keinerlei Eigenschaften in Bezug auf Material und Physik (Wie viel Last kann ein Ast eines Baumes tragen)
- Schnell und einfach implementiert, um mit L-Systemen zu experimentieren
- Geringer Speicheraufwand: Alphabet und Produktionen → Rest kann abgeleitet werden
- Als Sprachtyp-2 können L-Systeme im Compilerbau für Pflanzenwachstumsprozesse verwendet werden
- Eigenschaften von Typ-2 Sprachen können abgeleitet werden für das Erstellen von effizienten Algorithmen (Keller Automaten)
- Kein Industriestandard, weil fehlende Schnittstellen, Weiterentwicklungen etc.

Fazit - Alternativen