Theoretische Informatik 2

- Komplexität (5-7 Wochen)
  - insbesondere Analyse des (Zeit-) Aufwands bei der Ausführung von Algorithmen
- Formale Sprachen & Berechenbarkeitsmodelle
  - Fortführung ThI 1 (Rest des Sem.)
- heute
  - Fallstudie zum Aufwand (nicht im Skript)

Fallstudie: Drachenkurve

- bekannte flächenfüllende Kurve, bekanntes Fraktal
- algorithmische Modellierung als Approximation durch wiederholtes Falten eines Papierstreifens (Linie)
- genauer: als Liniensequenzen in Abhängigkeit von Zahl der Faltungen durch Angabe der Himmelsrichtungen, in die Einheitslinien verlaufen
- Alphabet: $COMPASS = \{N, O, S, W\}$
- sogenannte Kettencodes zur Beschreibung der Liniensequenzen: $u \in COMPASS^*$
- z.B. NWSWSOSWSONOSOSW ↦

dragoncurve

Name der Spezifikation

opns: $dc : \mathbb{N} \to COMPASS^*$
$tb : COMPASS^* \to COMPASS^*$

Deklaration von Operationen (Name, Argument- & Wertebereiche)

vars: $n \in \mathbb{N}$
$u \in COMPASS^*$

Deklaration von Variablen (Name, Typ)

eqns: $dc(0) = N$
$dc(n+1) = dc(n)\, tb(dc(n))$
$tb(\lambda) = \lambda$
$tb(Nu) = tb(u)W$
$tb(Ou) = tb(u)N$
$tb(Su) = tb(u)O$
$tb(Wu) = tb(u)S$

Definition der Operationen durch Gleichungen; Festlegung des Effekts von einzelnen Rechenschritten mit Rekursion über den induktiven Aufbau natürlicher Zahlen und Zeichenketten

Berechnungsbeispiele

$dc(0) = N$

$dc(1) = dc(0)\ tb(dc(0)) = N\ tb(N) \overset{1}{=} NW$

$dc(2) = dc(1)\ tb(dc(1)) = NW\ tb(NW) \overset{1}{=} NWSW$

$dc(3) = dc(2)\ tb(dc(2)) = NWSW\ tb(NWSW) \overset{1}{=} NWSWSOSW$

$dc(4) = dc(3)\ tb(dc(3)) = NWSWSOSW\ tb(NWSWSOSW) \overset{1}{=} NWSWSOSWSONOSOSW$

$\vdots$

1) genauer:

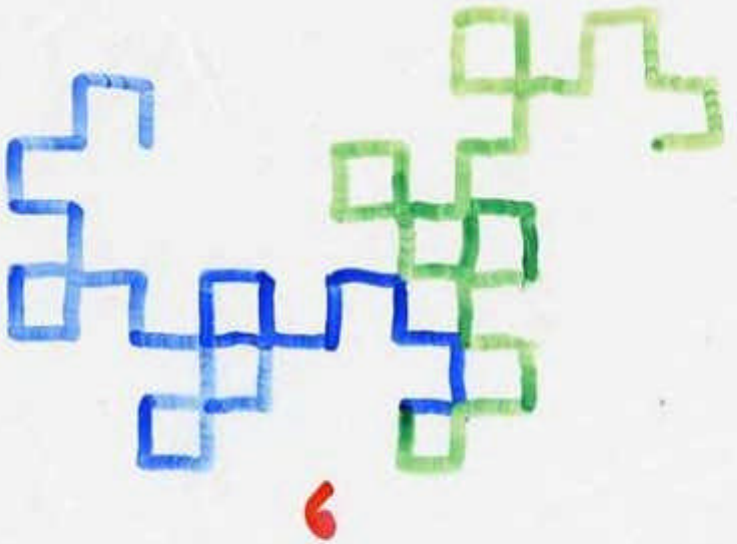
$tb(N) = tb(\lambda)W = W$

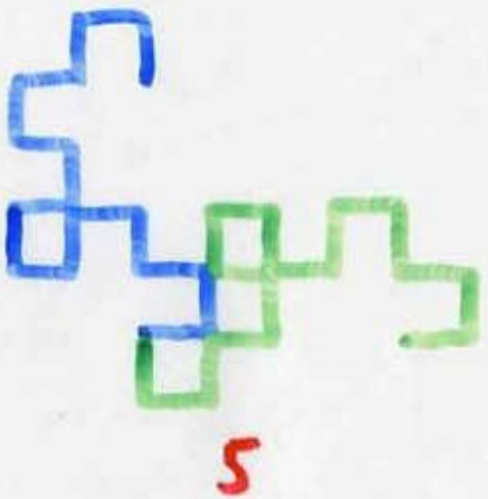
$tb(NW) = tb(W)W = tb(\lambda)SW = SW$

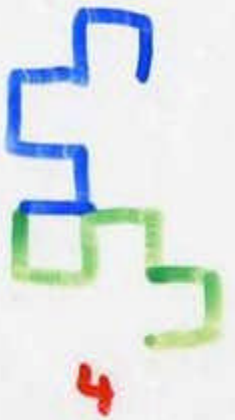
$tb(NWSW) = tb(WSW)W = tb(SW)SW = tb(W)OSW = tb(\lambda)SOSW = SOSW$

$\vdots$

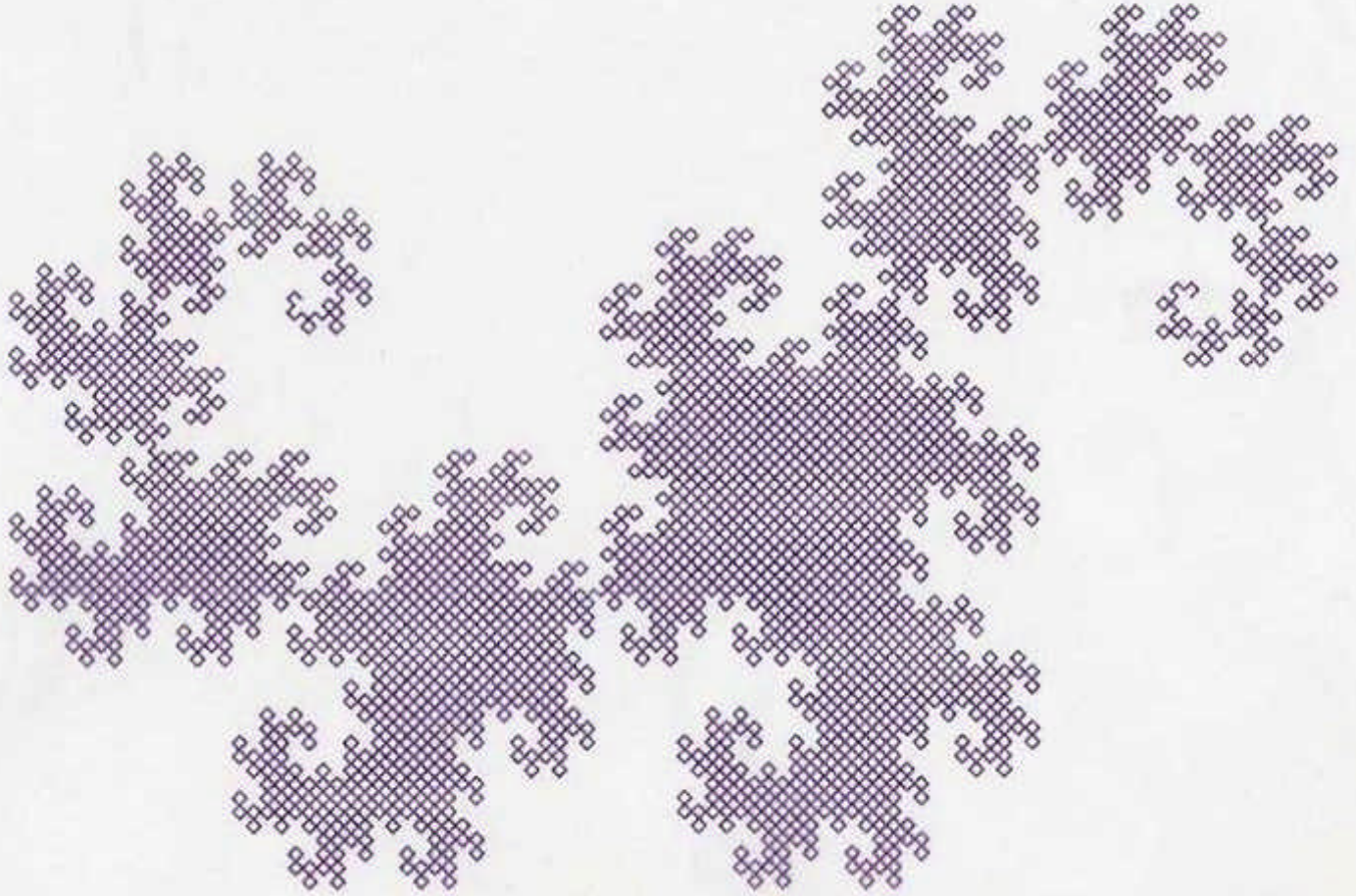
die ersten 6 Approximationen der Drachenkurve







die 12.



Bestimmung des Aufwands von dc

- als Zahl der Rechenschritte bei der Auswertung in Abhängigkeit von der Größe der Eingaben:

$$T^{dc}(n) \quad \& \quad T^{tb}(n)$$

- gemäß 1. und 2. Gleichung gilt:

$$T^{dc}(0) = 1$$

$$T^{dc}(n+1) = 1 + T^{dc}(n) + T^{tb}(m) + T^{dc}(n)$$

(m: Eingabegröße ist Länge von $dc(n)$)

$$= 2 \cdot T^{dc}(n) + 2^n + 2$$

(falls $T^{tb}(m) = m+1$ und $m = 2^n$)

- mit vollständiger Induktion über n ergibt sich daraus:

$$T^{dc}(n) \leq (n+1) \cdot 2^n$$

△ gemäß der Gleichungen 3 bis 7 gilt:

$T^{tb}(0) = 1$

$T^{tb}(n+1) = 1 + T^{tb}(n)$

und damit $T^{tb}(n) = n+1$ (einfache Induktion)

△ ebenfalls mit Induktion ergibt sich für die Länge der Approximationen der Drachenkurve:

$$Length(dc(n)) = 2^n$$

(I.A.: $Length(dc(0)) = Length(N) = 1 + Length(L) = 1 + 0 = 1$;

I.S.: $Length(dc(n+1)) = Length(dc(n)\ tb(dc(n)))$

$= Length(dc(n)) + Length(tb(dc(n)))$

$= 2 \cdot Length(dc(n)) \overset{I.V.}{=} 2 \cdot 2^n = 2^{n+1}$

falls $Length(tb(u)) = Length(u)$ f.a. u)

▷ mit vollständiger Induktion über den Aufbau von Zeichenketten ergibt sich für alle $u \in COMPASS^*$:

$$Length(tb(u)) = Length(u)$$

(I.A.: $Length(tb(\lambda)) = Length(\lambda)$ nach 3. Gleichung;

I.S.: $Length(tb(x \cdot u)) = Length(tb(u)\,\overline{x})$

$= Length(tb(u)) + Length(\overline{x}) \overset{I.V.}{=} Length(u) + 1$

$= Length(x \cdot u)$

wobei $\overline{N} = W$, $\overline{O} = N$, $\overline{S} = O$, $\overline{W} = S$ ist)