Theoretische Informatik 2

Sabine Kuske
Linzer Str. 9a, OAS 3005
Tel.: 2335, 8794
kuske@informatik.uni-bremen.de
www.informatik.uni-bremen.de/theorie

7. April 2008

Termine

- Vorlesung: 2 SWS
  - Mo von 10:00 - 12:00 GW2 B1410
- Tutorien: 2 SWS
  - Mo von 15:00 - 17:00 MZH 7250 (Diedrich Wolter),
  - Di von 08:00 - 10:00 MZH 7220
  - Di von 08:00 - 10:00 MZH 7050 (Sabine Kuske)
  - Di von 10:00 - 12:00 MZH 7220 (Sabine Kuske)
  - Di von 10:00 - 12:00 MZH 7210 (Diedrich Wolter)
  - Mi von 08:00 - 10:00MZH 7050 (Sabine Kuske)

Scheinkriterien

Es gibt zwei Möglichkeiten, den Leistungsnachweis zu erwerben:

1. Regelmäßige Bearbeitung von Übungsaufgaben und Fachgespräch
   - Übungsblätter werden in Gruppen bearbeitet; die Gruppengröße soll 3 nicht überschreiten.
   - Jedes Blatt muss mindestens zu 50% richtig bearbeitet werden. Ein Blatt darf nachgebessert werden.
   - Das Fachgespräch dauert ca. 10 Minuten pro Person und dient der Überprüfung der individuellen Leistungsfähigkeit. Es findet i.d.R. gegen Ende der Vorlesungszeit statt.

2. Mündliche Prüfung

- Eine mündliche Prüfung dauert 20-30 Minuten.

Weitere Infos unter

http://studienzentrum.informatik.uni-bremen.de/

Literatur

1. Skript (`www.informatik.uni-bremen.de/theorie/teach/thi2/`)
2. John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie. Addison-Wesley, 2002.
3. Uwe Schöning. *Theoretische Informatik – kurzgefaßt* (4. Auflage). Spektrum Akademischer Verlag, 2003.
4. Gottfried Vossen and Kurt-Ulrich Witt. Grundlagen der Theoretischen Informatik mit Anwendungen (4. Auflage). Vieweg, Braunschweig, 2006.

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Was kann ein Computer (nicht) berechnen?

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Wie lange muss man höchstens oder mindestens auf ein Ergebnis warten?

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Für welche Probleme existieren bisher nur viel zu langsame Lösungen?

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Macht es Sinn, für ein gegebenes Problem eine Lösung zu entwickeln?

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Welches Modellierungswerkzeug ist für welchen Zweck geeignet?

Theoretische Informatik ist...

wichtig

und

interessant,

denn sie beantwortet Fragen, wie z.B.

Wie erhält man möglichst fehlerfreie Lösungen?

Themen dieses Kurses

1. Komplexitätstheorie
   - Wie lange brauchen Ausführungen von Algorithmen?
   - Wieviel Speicherplatz brauchen Ausführungen von Algorithmen?
   - Welche Berechnungszeiten sind noch akzeptabel?
   - Welche Probleme sind zu aufwendig?
   - Wie schätzt man Zeit- oder Platzverbrauch ab?

2. Formale Sprachen

- Welche Grammatiktypen gibt es, die mehr als die kontextfreien Sprachen erzeugen können?
- Welche Sprachen werden von diesen allgemeineren Grammatiken erzeugt?
- Welche Automatenmodelle erkennen diese allgemeineren Sprachklassen?
- Für welche Sprachklassen ist das Wortproblem (noch) lösbar und mit welchem Aufwand?

3. Berechenbarkeit

- Wie funktionieren Turing-Maschinen und was können sie berechnen?
- Können deterministische Turing-Maschinen genauso viel wie nichtdeterministische?
- Welche Sprachen können durch Turing-Maschinen erkannt werden?
- Wie hängen verschiedene Berechenbarkeitsmodelle zusammen? Sind sie ineinander überführbar?

Wörter (Wiederholung)

- Beispiele
  - Programmiersprachen:
    Namen, Ausdrücke, Datentypen, Programme,. . .
  - Systemmodellierung:
    mögliche Abläufe, Ereignisfolgen, verbotene Folgen,. . .
  - Natürlche Sprachen:
    Wörter, Sätze, Texte, . . .
  - Textsysteme:
    Schreiben und Lesen, Verändern und analysieren: Cut & Paste,
    Zeichen zählen, Vergleichen, Zusammensetzen, . . .

Erzeugung von Wörtern

- A : Alphabet (Menge von Zeichen)
- A^*: Menge aller Wörter über A.

Rekursive Definition von A^*

- $\lambda \in A^*$ (leeres Wort, enthält keine Zeichen)
- mit $x \in A$ und $v \in A^*$ ist auch $xv \in A^*$ (Linksaddition)

Konvention: $v\lambda = v$ für alle $v \in A^*$

Beispiel

$A = \{bei, spiel\}$

Erzeugungsprozess

1. λ,
2. bei $(= bei\lambda)$, $spiel$ $(= spiel\lambda)$,
3. $beibei$,$spielbei$, $beispiel$,$spielspiel$
4. $beibeibei$, $spielbeibei$, $beispielbei$, $spielspielbei$, $beibeispiel$, $spielbeispiel$, $beispielspiel$,$spielspielspiel$
5. ...

Induktionsprinzip

- Induktionsanfang (IA):
  Zeige THEOREM für $v = \lambda$.
- Induktionsvoraussetzung (IV):
  Nimm THEOREM für v an.
- Induktionsschluss (IS):
  Zeige THEOREM für xv mit $x \in A$.

Konkatenation von Wörtern

1. für $v = \lambda$ gilt $\lambda \cdot w = w$,
2. für $v = xu$ mit $x \in A$ gilt $(xu) \cdot w = x(u \cdot w)$.

Beispiel

$$aba \cdot cd = a(ba \cdot cd) = a(b(a \cdot cd)) = a(b(a(\lambda \cdot cd))) =$$
$$a(b(a(cd))) = a(b(acd)) = a(bacd) = abacd$$

Satz (Assoziativität)
Für alle Wörter u, v, w gilt $(uv)w = u(vw)$.

Länge

1. $length(\lambda) = 0$
2. $length(xv) = length(v) + 1$ für $x \in A, v \in A^*$

Beispiel

$length(aba) = 1 + length(ba) = 1 + 1 + length(a) = 2 + 1 + length(\lambda) = 3 + 0 = 3$

Satz Für alle $u, v \in A^*$ gilt:

$$length(uv) = length(u) + length(v).$$

Zeichen zählen

1. $count(x, \lambda) = 0$
2. $count(x, yv) = \begin{cases} count(x, v) + 1, & \text{falls } x \equiv y \\ count(x, v) & \text{sonst} \end{cases}$

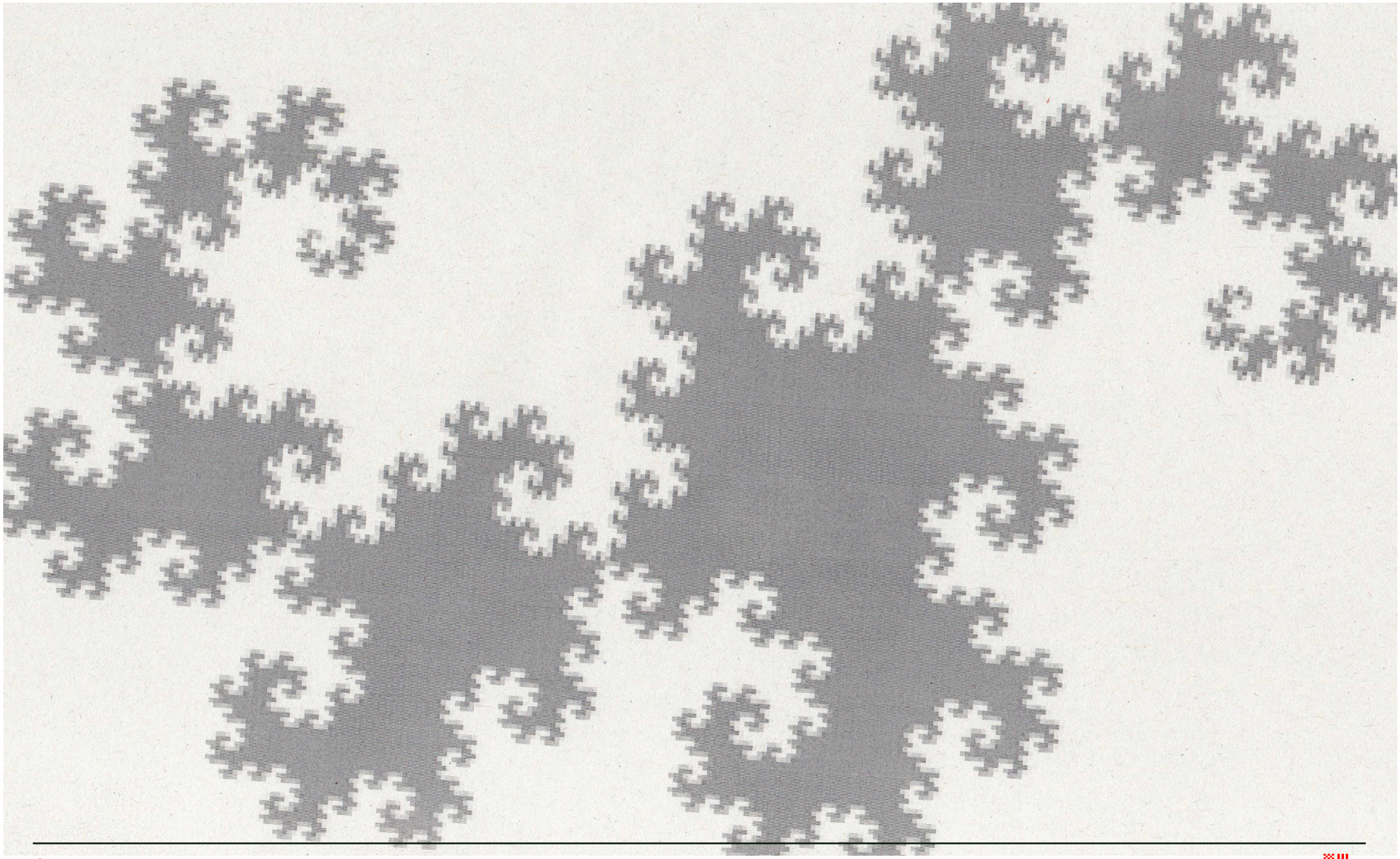
Satz

Für alle $x \in A$, $u, v \in A^*$ gilt:

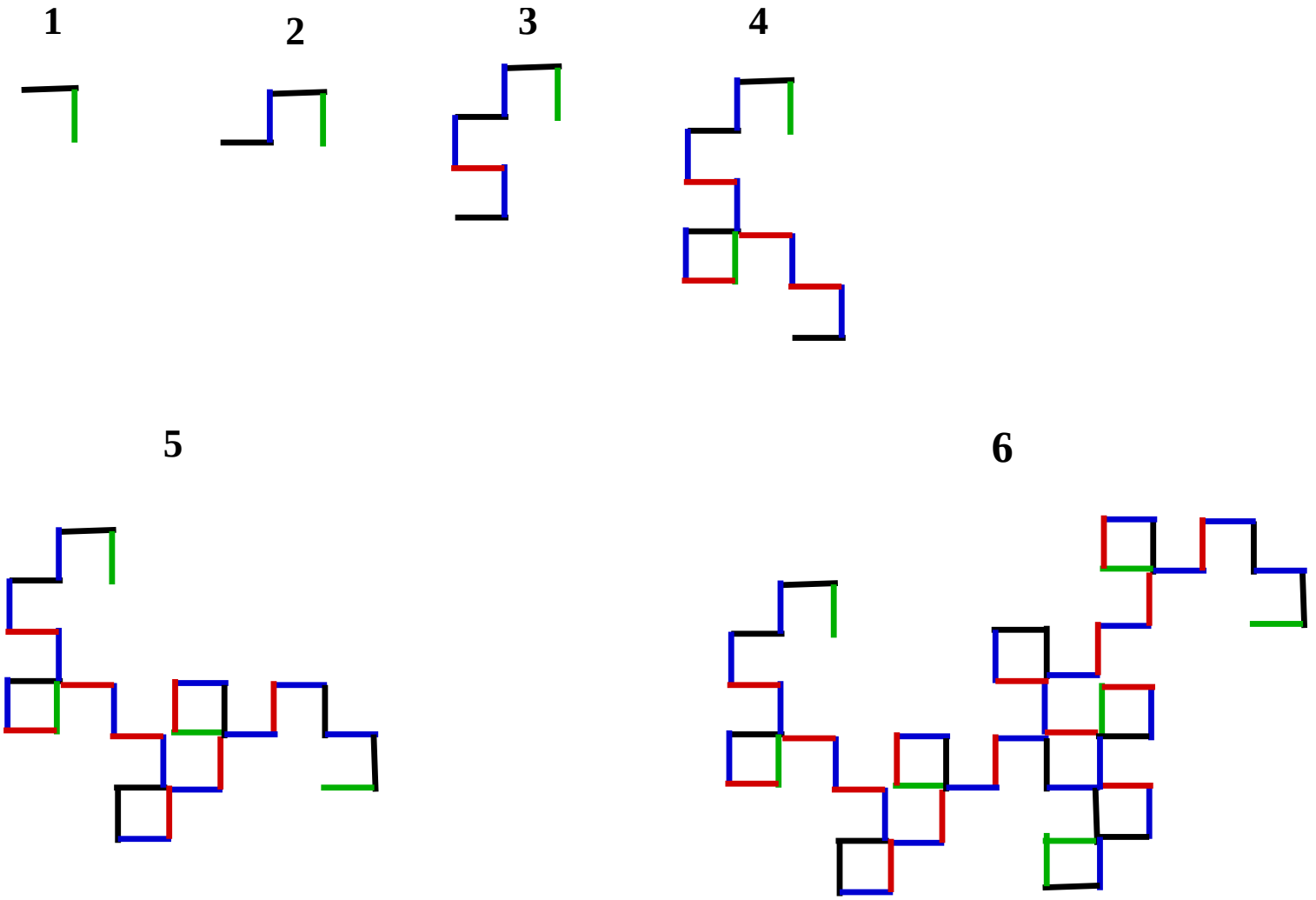
$$count(x, uv) = count(x, u) + count(x, v).$$

Fallstudie "Drachenkurve"

- Bekannte flächenfüllende Kurve, bekanntes Fraktal
- Algorithmische Modellierung als Approximation durch wiederholtes Falten eines Papierstreifens (Linie)
- Genauer: als Liniensequenzen in Abhängigkeit von der Zahl der Faltungen durch Angabe der Himmelsrichtungen, in die Einheitslinien verlaufen



Die Approximationen 1 bis 6

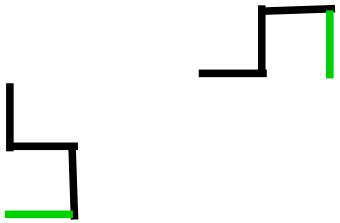


Erzeugung der 4. Approximation aus der 3.

- 3. Aproximation

- Kopieren

- Im Uhrzeigersinn drehen (90 Grad)



- Endpunkte zusammenfügen