

Vorlesung

Grundlagen der Theoretischen Informatik / Einführung in die Theoretische Informatik I

Bernhard Beckert

Institut für Informatik



Sommersemester 2007

Diese Vorlesungsmaterialien basieren ganz wesentlich auf den Folien zu den Vorlesungen von

Katrin Erk (gehalten an der Universität Koblenz-Landau)

Jürgen Dix (gehalten an der TU Clausthal)

Ihnen beiden gilt mein herzlicher Dank.

– Bernhard Beckert, April 2007

Teil I

Einführung

- 1 **Organisatorisches**
- 2 Motivation, Inhalt der Vorlesung
- 3 Kurzer Überblick: Logik
- 4 Kurzer Überblick: Beweismethoden und Mathematische Konzepte

Bernhard Beckert

Email: beckert@uni-koblenz.de

Webseite: www.uni-koblenz.de/~beckert

Raum: B 218

Sprechstunde:

Donnerstags, 16 Uhr

(sonst auch immer, wenn die Tür offen steht)

Ulrich Koch

Email: koch@uni-koblenz.de

Webseite: www.uni-koblenz.de/~koch

Raum: A 219

<http://www.uni-koblenz.de/~beckert/Lehre/TheoretischeInformatik/>

Alle relevante Information auf der Webseite

- Folien
- Weitere Materialien
- Termine usw.

Termine und Einteilung

Stehen noch nicht endgültig fest

Webseite beachten:

MeToo-Registrierung sobald Termine feststehen!

Übungsblätter

- Wöchentlich, jeden Montag
- Dürfen – müssen aber nicht – abgegeben werden
- Werden in den Übungen der darauffolgenden Woche besprochen
- Kein Einfluss auf Scheinvergabe

Teilklausuren während des Semesters

- Während des Semesters
- drei Teilklausuren (je ca. 30-40 Minuten)
- jeweils Mitte Mai, Mitte Juni, Mitte Juli
- Schein bei 50% der insgesamt in den drei Teilklausuren zu erzielenden Punkte
- Freiversuch gilt für alle Teilklausuren zusammen
- **Wiederholung einzelner Teilklausuren nicht möglich**

Nachklausur zum Ende der Semesters

- Gegen Ende des Semesters (Ende September)
- Nachklausur hat gleichen „Wert“ wie alle Teilklausuren zusammen
- 90 Minuten Dauer
- Schein bei 50% der Punkte in der Nachklausur

Anmeldung zu und Teilnahme an der ersten Prüfung (also den Teilklausuren während des Semesters) ist Voraussetzung für die Teilnahme an der Nachklausur.

Zudem darf an der Nachklausur (wie auch an den Teilklausuren) teilnehmen, wer an einer Prüfung zu „Einführung in die Theoretische Informatik I“ in frühen Semestern teilgenommen und diese nicht bestanden oder einen Freiversuch gesetzt hat.

Buch zur Vorlesung



Erk, Katrin and Priese, Lutz (2002).

Theoretische Informatik: Eine umfassende Einführung.

2. Auflage.

Springer-Verlag.

Weitere Literatur

 J. Hopcroft, R. Motwani, and J. Ullman (2002).

Einführung in die Automatentheorie, Formale Sprachen und Komplexitätstheorie.

Pearson.

 G. Vossen and K.-U. Witt (2004).

Grundkurs Theoretische Informatik.

Vieweg.

 U. Schöning (1994).

Theoretische Informatik: kurzgefaßt.

Spektrum-Verlag.

Teil I

Einführung

- 1 Organisatorisches
- 2 Motivation, Inhalt der Vorlesung**
- 3 Kurzer Überblick: Logik
- 4 Kurzer Überblick: Beweismethoden und Mathematische Konzepte

Grundlegende Konzepte der Informatik

- Probleme und ihre Beschreibung
- Systeme/Automaten/Maschinen, die Probleme lösen
- Lösbarkeit von Problemen
(Entscheidbarkeit/Berechenbarkeit und deren Grenzen)
- Schwierigkeit (Komplexität) der Lösung von Problemen

Teilgebiete

- Formale Sprachen
- Automatentheorie
- Berechenbarkeitstheorie
- Komplexitätstheorie
- (Logik)

Theoretische Informatik ...

- ist die Grundlage
- ist wichtig
(bspw. für Algorithmentechnik, Software Engineering, Compilerbau)
- hilft, weitere Themen/Vorlesungen der Informatik zu verstehen
- veraltet nicht
- macht Spaß

- 1 Terminologie
- 2 Endliche Automaten und reguläre Sprachen
- 3 Kellerautomaten und kontextfreie Sprachen
- 4 Turingmaschinen und rekursiv aufzählbare Sprachen
- 5 Berechenbarkeit, (Un-)Entscheidbarkeit
- 6 Komplexitätsklassen P und NP

Teil I

Einführung

- 1 Organisatorisches
- 2 Motivation, Inhalt der Vorlesung
- 3 Kurzer Überblick: Logik**
- 4 Kurzer Überblick: Beweismethoden und Mathematische Konzepte

Aussagenlogische Operatoren

- \neg Negationssymbol („nicht“)
- \wedge Konjunktionssymbol („und“)
- \vee Disjunktionssymbol („oder“)
- \Rightarrow Implikationssymbol („wenn ... dann“)
- \Leftrightarrow Symbol für Äquivalenz („genau dann, wenn“)

Zusätzliche prädikatenlogische Operatoren

- \forall Allquantor („für alle“)
- \exists Existenzquantor („es gibt“)

Formeln

- Atomare Aussagen sind Formeln
- Seien A, B Formeln, x eine Variable, dann sind

$$\neg A, (A \wedge B), (A \vee B), (A \Rightarrow B), (A \Leftrightarrow B), \forall x A, \exists x A$$

Formeln

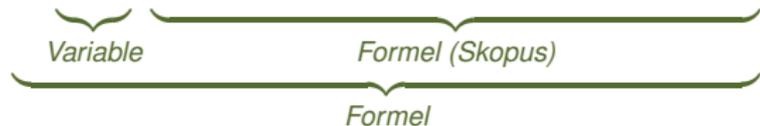
Beispiel 3.1

$$\underbrace{\neg(y \leq x)}_{\text{Atom}} \Rightarrow \underbrace{\exists z(\neg(z \leq x) \wedge \neg(y \leq z))}_{\text{Skopus von } \exists z}$$

Beispiel 3.2

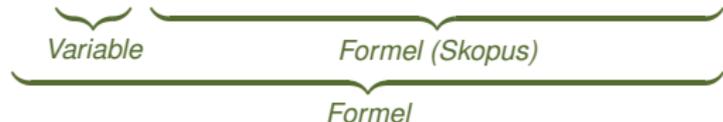
„Alle, die in Koblenz studieren, sind schlau“

$$\forall x \quad (\text{studiertIn}(x, \text{koblenz}) \Rightarrow \text{schlau}(x))$$



„Jemand, der in Landau studiert ist schlau“

$$\exists x \quad (\text{studiertIn}(x, \text{landau}) \wedge \text{schlau}(x))$$



Quantoren gleicher Art kommutieren

$\forall x \forall y$	ist das gleiche wie	$\forall y \forall x$
$\exists x \exists y$	ist das gleiche wie	$\exists y \exists x$

Eigenschaften von Quantoren

Verschiedene Quantoren kommutieren NICHT

$\exists x \forall y$ ist **nicht** das gleiche wie $\forall y \exists x$

Beispiel 3.3

$\exists x \forall y \text{ loves}(x, y)$

*Es gibt eine Person, die jeden Menschen in der Welt liebt
(einschließlich sich selbst)*

$\forall y \exists x \text{ loves}(x, y)$

Jeder Mensch wird von mindestens einer Person geliebt

*(Beides hoffentliche wahr aber verschieden:
das erste impliziert das zweite aber nicht umgekehrt)*

Verschiedene Quantoren kommutieren NICHT

$\exists x \forall y$ ist **nicht** das gleiche wie $\forall y \exists x$

Beispiel 3.4

$\forall x \exists y \text{mutter}(y, x)$

Jeder hat eine Mutter **(richtig)**

$\exists y \forall x \text{mutter}(y, x)$

Es gibt eine Person, die die Mutter von jedem ist **(falsch)**

Dualität der Quantoren

$\forall x \dots$ ist das gleiche wie $\neg \exists x \neg \dots$

$\exists x \dots$ ist das gleiche wie $\neg \forall x \neg \dots$

Beispiel 3.5

$\forall x \text{ mag}(x, \text{eiskrem})$ ist das gleiche wie $\neg \exists x \neg \text{mag}(x, \text{eiskrem})$

$\exists x \text{ mag}(x, \text{broccoli})$ ist das gleiche wie $\neg \forall x \neg \text{mag}(x, \text{broccoli})$

Eigenschaften von Quantoren

\forall distributiert NICHT über \vee

$\forall x (... \vee ...)$ ist **NICHT** das gleiche wie $(\forall x ...) \vee (\forall x ...)$

Beispiel 3.6

$\forall x (eiskrem(x) \vee broccoli(x))$ ist **NICHT** das gleiche wie
 $(\forall x eiskrem(x)) \vee (\forall x broccoli(x))$

\exists distributiert NICHT über \wedge

$\exists x (... \wedge ...)$ ist NICHT das gleiche wie $(\exists x ...) \wedge (\exists x ...)$

Beispiel 3.7

$\exists x (\text{gerade}(x) \wedge \text{ungerade}(x))$ ist NICHT das gleiche wie
 $(\exists x \text{gerade}(x)) \wedge (\exists x \text{ungerade}(x))$

Beispiel 3.8

- „Brüder sind Geschwister“

$$\forall x \forall y (bruder(x, y) \Rightarrow geschwister(x, y))$$

- „bruder“ ist symmetrisch

$$\forall x \forall y (bruder(x, y) \Leftrightarrow bruder(y, x))$$

- „Mütter sind weibliche Elternteile“

$$\forall x \forall y (mutter(x, y) \Leftrightarrow weiblich(x) \wedge elter(x, y))$$

- „Ein Cousin ersten Grades ist
das Kind eines Geschwisters eines Elternteils“

$$\forall x \forall y (cousin1(x, y) \Leftrightarrow \\ \exists p \exists ps (elter(p, x) \wedge geschwister(ps, p) \wedge elter(ps, y)))$$

Beispiel 3.9

Formalisierung von „Bruder, der nicht nur Halbbruder ist“

$$\begin{aligned} \forall x \forall y \text{ bruder}(x, y) \Leftrightarrow & (\neg(x = y) \wedge \\ & \exists m \exists v (\neg(m = v) \wedge \\ & \text{elter}(m, x) \wedge \text{elter}(v, x) \wedge \\ & \text{elter}(m, y) \wedge \text{elter}(v, y))) \end{aligned}$$

Teil I

Einführung

- 1 Organisatorisches
- 2 Motivation, Inhalt der Vorlesung
- 3 Kurzer Überblick: Logik
- 4 Kurzer Überblick: Beweismethoden und Mathematische Konzepte**

Deduktiver Beweis

- Aneinanderkettung von Argumenten/Aussagen

$$A = A_1, A_2, \dots, A_n = B$$

- Zwischenaussagen A_i müssen schlüssig aus dem Vorhergehenden folgen
- Verwendet werden dürfen nur
 - Annahmen aus A
 - mathematische Grundgesetze
 - bereits bewiesene Aussagen
 - logische Schlussfolgerungen

Beispiel 4.1 (Deduktiver Beweis)

Zu beweisen:

Wenn x die Summe der Quadrate von vier positiven ganzen Zahlen ist, dann gilt $2^x \geq x^2$

Beweis in schematischer Darstellung:

Aussage	Begründung
1. $x = a^2 + b^2 + c^2 + d^2$	Gegeben
2. $a \geq 1, b \geq 1, c \geq 1, d \geq 1$	Gegeben
3. $a^2 \geq 1, b^2 \geq 1, c^2 \geq 1, d^2 \geq 1$	(2) und Gesetze der Arithmetik
4. $x \geq 4$	(1), (3) und Gesetze der Arithmetik
5. $2^x \geq x^2$	(4) und Satz aus der Analysis

Beweis durch Kontraposition

Beweise, dass **nicht A** aus der Annahme **nicht B** folgt:

$$\neg B \Rightarrow \neg A \quad \text{logisch äquivalent zu} \quad A \Rightarrow B$$

Widerspruchsbeweis

Beweise, dass aus **A und nicht B** ein Widerspruch folgt:

$$(A \wedge \neg B) \Leftrightarrow \text{false} \quad \text{logisch äquivalent zu} \quad A \Rightarrow B$$

- A kann „leer“ sein
- Widerspruch zu \forall -Aussage gelingt durch Gegenbeispiel

Induktionsbeweise

Standardinduktion:

Gilt $A(0)$ und
folgt $A(i+1)$ aus $A(i)$,
dann gilt $A(n)$ für alle $n \in \mathbb{N}$

Vollständige Induktion:

Gilt $A(0)$ und
folgt $A(i+1)$ aus $A(0) \wedge \dots \wedge A(i)$,
dann gilt $A(n)$ für alle $m \in \mathbb{N}$

Strukturelle Induktion (auf Datentypen wie Listen, Bäumen, Wörtern):

Gilt A für das Basiselement und
folgt die Gültigkeit von A für ein beliebiges Element aus der
Gültigkeit von A für seine Unterelemente,
dann gilt A für alle Elemente.

Grundkonzepte (z.B. aus DAS)

- Elementare **Mengentheorie**
 - $\{x|P(x)\}$
 - \cup, \cap, \setminus
- Bezug zwischen Mengen, Relationen und Funktionen (**wichtig!**)
- Elementare Gesetze der Algebra
- Strukturen wie Listen, (endliche) Folgen, Graphen, Bäume
- Wörter (Strings)

Funktionen

Funktion $f : S \rightarrow S'$: Abbildung zwischen den Grundmengen S und S' ,
nicht unbedingt auf allen Elementen von S definiert

Definitionsbereich D

Wertebereich W

f eine totale Funktion: f für alle Elemente des Wertebereichs W definiert

$$\forall x \in D \exists y \in W (x, y) \in f$$

Injektiv, surjektiv, bijektiv