Theoretische Informatik I

Sommersemester 2004

Christoph Kreitz / Jens Otten

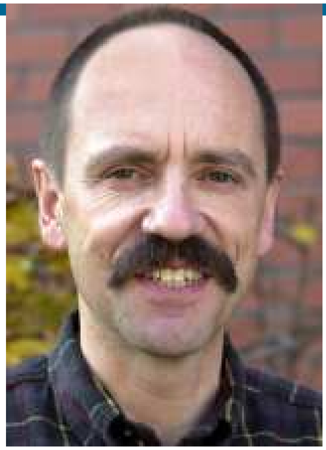
Theoretische Informatik, Raum 1.18, Telephon 3060

{kreitz,jeotten}@cs.uni-potsdam.de

http://www.cs.uni-potsdam.de/ti

1. **Das Team**
2. **Themen der Theoretischen Informatik**
3. **Organisatorisches**
4. **Gedanken zur Arbeitsethik**

Das Team



Christoph Kreitz
Raum 1.18, Telephon 3060
kreitz@cs.uni-potsdam.de

Jens Otten
Raum 1.20, Telephon 3072
jeotten@cs.uni-potsdam.de

Kirstin Peters
peters.kirstin@web.de

Sandro Schugk
sandroschugk@gmx.de

Hermann Schwarting
ti2@knackich.de

Was ist Theoretische Informatik?

Mathematische Analyse von Grundsatzfragen

- **Problemstellungen**
  - Präzisierung: Wie beschreibt man Probleme?
  - Berechenbarkeit: Ist ein Problem überhaupt lösbar?
  - Effizienz: Ist ein Problem schwer oder leicht lösbar?
- **Algorithmen: abstrakte Lösungsmethoden**
  - Was ist Algorithmus und welche Beschreibungsformen gibt es?
  - Welche grundsätzlichen Merkmale und Eigenschaften haben Algorithmen?
- **Programme: konkrete Lösungsvorschriften**
  - Welche grundsätzlichen Arten von Sprachen gibt es?
  - Syntax: Wie kann man Sprachen beschreiben, erkennen und erzeugen?
  - Semantik: wie beschreibt man die Bedeutung von Programmen
- **Maschinen: Ausführung von "Berechnungen"**
  - Welche grundsätzlichen Typen und Merkmale von Maschinen gibt es?
  - Was können bestimmte Maschinentypen leisten?

Typische Theoretische Fragestellungen

- **Sind Korrekheitsgarantien möglich?**
  - Kein Systemabsturz, kein “Aufhängen”, keine falschen Resultate
  - Wie erzeugt man korrekte Programme?
- **Wie effizient kann Software sein?**
  - Wieviel RAM, Plattenplatz, Rechenzeit wird benötigt?
  - Skalierbarkeit: wie groß darf das Problem werden?
- **Wie einfach kann Interaktion gemacht werden?**
  - Wieviel Freiheiten sind in der Formulierung von Programmen möglich?
  - Wie aufwendig wird Syntaxanalyse und Compilierung?
- **Gibt es Grenzen?**
  - Können alle Aufgaben irgendwann von Computern übernommen werden?
  - Werden ineffiziente Programme durch Hardwaresteigerungen handhabbar?
  - Können Computer irgendwann jede natürliche Sprache verstehen?

Themen der Theoretischen Informatik

- **Mathematische Methodik in der Informatik** TI-1
- **Automatentheorie und Formale Sprachen** TI-1
  - Endliche Automaten und Reguläre Sprachen
    - Lexikalische Analyse
  - Kontextfreie Sprachen und Pushdown Automaten
    - Syntaxanalyse und Semantik
  - Die Chomsky Hierarchie
- **Theorie der Berechenbarkeit** TI-2
  - Berechenbarkeitsmodelle
  - Aufzählbarkeit und Entscheidbarkeit
  - Unlösbare Probleme (Unentscheidbarkeit)
- **Komplexitätstheorie** TI-2
  - Komplexitätsmaße und -klassen für Algorithmen und Probleme
  - Nicht handhabbare Probleme ($\mathcal{NP}$-Vollständigkeit)

Organisatorisches

- **Kreditpunkte: 6**
  - Verbindliche Anmeldung bis 5. Mai
- **Vorlesung**
  - Vorstellung und Illustration zentraler Konzepte
  - Wöchentlich Mi 13:30-15:00
- **Übungen**
  - Vertiefung/Anwendung durch Aufgaben, Quiz, Klärung von Fragen
  - 4(+1) Gruppen, wöchentlich je 2 Stunden — Eintrag in Listen erforderlich
- **Sprechstunden**
  - C. Kreitz: Mi 11:00–12:30 . . . , und immer wenn die Türe offen ist
  - J. Otten: Di 14:00–16:00 . . . , und immer wenn die Türe offen ist
  - Tutoren: individuell in Übungsgruppen vereinbaren

Zyklus wird im Winter 2004/2005 umgestellt

Der Lehrstoff

- **Reihenfolge und Notation folgt Leittext**
  - J. Hopcroft, R. Motwani, J. Ullman: *Einführung in die Automaten–theorie, Formale Sprachen und Komplexitätstheorie, Pearson 2002*
  - Vorlesungsfolien (meist im Voraus) auf dem Webserver erhältlich
  - Mitschriften aus früheren Semestern benutzen andere Notationen
- **Lesenswerte Zusatzliteratur**
  - A. Asteroth, C. Baier: *Theoretische Informatik, Pearson 2002*
  - I. Wegener: *Theoretische Informatik, Teubner Verlag 1993*
  - U. Schöning: *Theoretische Informatik - kurzgefaßt, Spektrum-Verlag 1994*
  - K. Erk, L. Priese: *Theoretische Informatik, Springer Verlag 2000*
  - H. Lewis, C. Papadimitriou: *Elements of the Theory of Computation, Prentice-Hall 1998*
- **Vorlesung ist knapp und "unvollständig"**
  - Die Idee (Verstehen) zählt mehr als das Detail (Aufschreiben)
  - Es hilft, die entsprechenden Kapitel des Buchs im Voraus zu lesen
  - Aktive Teilnahme an Übungen ist notwendig für erfolgreiches Lernen

Leistungserfassung

- **Eine Klausur entscheidet die Note**
  - Hauptklausur Mitte Juli (letzten Vorlesungswoche)
  - Nachklausur für Durchgefallene Studenten – maximal 4.0 erreichbar
  - Probeklausur Anfang Juni (geht nicht in Bewertung ein)
- **Zulassung zur Klausur**
  - 50% der Punkte in den Hausaufgaben
    - Gruppen bis 3 Studenten dürfen gemeinsame Lösungen abgeben
  - Teilnahme an Probeklausur
- **Vorbereitung auf die Klausur**
  - 5-Minuten Kurzquiz in Übungsstunde
  - Präsentation eigener Lösungen zu Hausaufgaben + ad hoc Übungsaufgaben
  - Klärung von Fragen in Übung und Sprechstunden
  - Feedback durch Korrektur der Hausaufgaben und der Probeklausur

Fangen sie frühzeitig mit der Vorbereitung an

Nutzen Sie Ihre Chancen!

- **Theorie ist bedeutender als viele glauben**
  - Ist Theorie langweilig? überflüssig? unverständlich? ...eine Plage?
  - Alle großen Softwareprojekte benutzten theoretische Modelle
  - Ohne theoretische Kenntnisse begehen Sie viele elementare Fehler
  - Theorie kann durchaus sehr interessant sein
- **Es geht um mehr als nur bestehen**
  - Das wichtige ist Verstehen
  - Sie können jetzt umsonst lernen, was später teure Lehrgänge benötigt
  - Wann kommen Sie je wieder mit den Besten des Gebietes in Kontakt?
- **Die Türe steht offen**
  - Lernfrust und mangelnder Durchblick ist normal aber heilbar
  - Kommen Sie in die Sprechstunden und stellen Sie Fragen

Vertrauen ist ein kostbares Gut

… mißbrauchen Sie es nicht

- **Abschreiben fremder Lösungen bringt nichts**
  – Sie lernen nichts dabei – weder Inhalt noch Durchhaltevermögen
  – Sie erkennen Ihre Lücken nicht und nehmen Hilfe zu spät wahr
  – Sie werden nie ein echtes Erfolgserlebnis haben
  – Es schadet Ihrer persönlichen Entwicklung
- **Wir vertrauen Ihrer Ehrlichkeit**
  – Benutzen Sie externe Ideen (Bücher/Internet) nur mit Quellenangabe
  – Benutzen Sie keine Lösungen von Kommilitonen (Ausnahme Lerngruppen)
  – Geben Sie keine Lösungen an Kommilitonen weiter
  – **Klausurlösungen sollten ausschließlich Ihre eigenen sein**

  Keine “Überwachung”, aber wenn es dennoch auffliegt …
- **Mehr zur Arbeitsethik auf unseren Webseiten**