Logik und diskrete Strukturen

Prof. Dr. Heiko Röglin

Institut für Informatik

Abteilung I

Wintersemester 2012/13

Organisatorisches

- **Vorlesung**
  Dienstag und Donnerstag
  10:15 – 11:45 Uhr (HS 1) und 12:30 – 14:00 Uhr (HS 2)
  Vorlesung am Vormittag = Vorlesung am Nachmittag

- **Übungen**
  16 Übungsgruppen
  Anmeldung über Tutorienvergabesystem
  https://puma.cs.uni-bonn.de/
  bis Freitag, den 12.10., um 8:00 Uhr.

universitätbonn

Dozenten und Tutoren

- **Vorlesung**
  Heiko Röglin (Abteilung I)
  Professor für Theoretische Informatik
  E-Mail: roeglin@uni-bonn.de
  Web: http://www.roeglin.org/
  Sprechstunde: Mittwochs 11-12 Uhr
  und nach der Vorlesung
  Büro: Friedrich-Ebert-Allee 144, E.63

- **Übungen**
  Organisation: Tobias Brunsch (Abt. I), Dennis den Brok (Abt. II)
  studentische Tutoren: Katharina Bauer, Linus Boes, Rafel Israels, Pia Kullik, Patrick Loka, Klara Reichard

universitätbonn

Übungsblätter und Studienleistung

Übungsblätter

- Ausgabe: Dienstag in der Vorlesung & auf Webseite
- Abgabe: Dienstag darauf,14 Uhr, Briefkasten Römerstraße
- Besprechung: in der Woche nach der Abgabe

Studienleistung

- insgesamt mind. 50% der möglichen Punkte
  (Abgabe in Gruppen mit max. 3 Studenten möglich)
- Präsentation von 3 (Teil)aufgaben in den Übungen

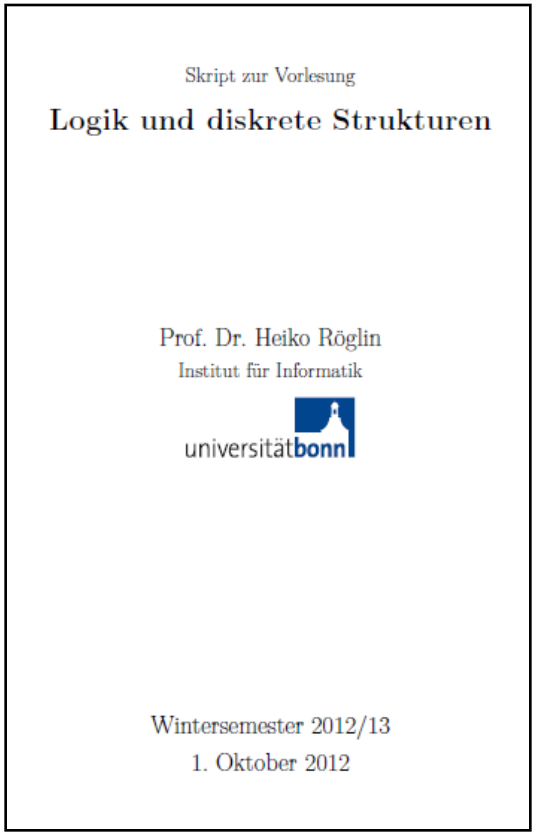
universitätbonn

(Ungebetene) Ratschläge

- Bleiben Sie am Ball und arbeiten Sie die Vorlesung nach.
- Arbeiten Sie kontinuierlich und lernen Sie nicht erst zwei Wochen vor der Prüfung.
- Bearbeiten Sie die Übungszettel selbstständig. (Starke Korrelation mit Klausurerfolg.)
- Trauen Sie sich, Fragen zu stellen. Sprechen Sie bei Problemen direkt mit dem Dozenten oder Ihrem Tutor.
- Gehen Sie mit Interesse an die Vorlesung ran.
- Feedback zur Veranstaltung ist jederzeit erwünscht.

universitätbonn

Literatur


Skript zur Vorlesung

Logik und diskrete Strukturen

Prof. Dr. Heiko Röglin

Institut für Informatik

universitätbonn

Wintersemester 2012/13

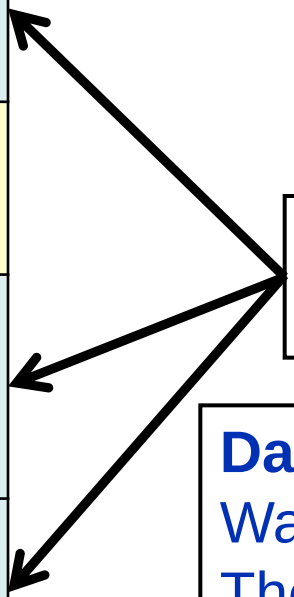
1. Oktober 2012

- Zu dieser Vorlesung gibt es ein Skript: http://www.roeglin.org/teaching/WS2012/LuDS.html
- Dieses wird während des Semesters kontinuierlich erweitert.
- Fallen Ihnen Fehler auf oder haben Sie Anregungen und Verbesserungs-vorschläge zum Skript, so sagen Sie mir bitte Bescheid.
- Sekundärliteratur: siehe Skript

universitätbonn

Vorlesungen über Theoretische Informatik

1.		Logik und diskrete Strukturen V4+Ü2=9 LP
2.	Lineare Algebra V4+Ü2=9 LP	Analysis V4+Ü2=9 LP
3.	Angewandte Mathematik V2+Ü2=6 LP	Algorithmen und Berechnungskomplexität I V4+Ü2=9 LP
4.		Algorithmen und Berechnungskomplexität II V2+Ü2=6 LP



Theoretische Informatik

Danach: Zahlreiche Wahlpflichtmodule in Theoretischer Informatik

Aber: Was ist Theoretische Informatik eigentlich?

Mathematische Grundlagen und Theoretische Informatik (48/94 LP)

universitätbonn

Theoretische Informatik

„Die Theoretische Informatik beschäftigt sich mit der Abstraktion, Modellbildung und grundlegenden Fragestellungen, die mit der Struktur, Verarbeitung, Übertragung und Wiedergabe von Informationen in Zusammenhang stehen.“ (Quelle: Wikipedia)

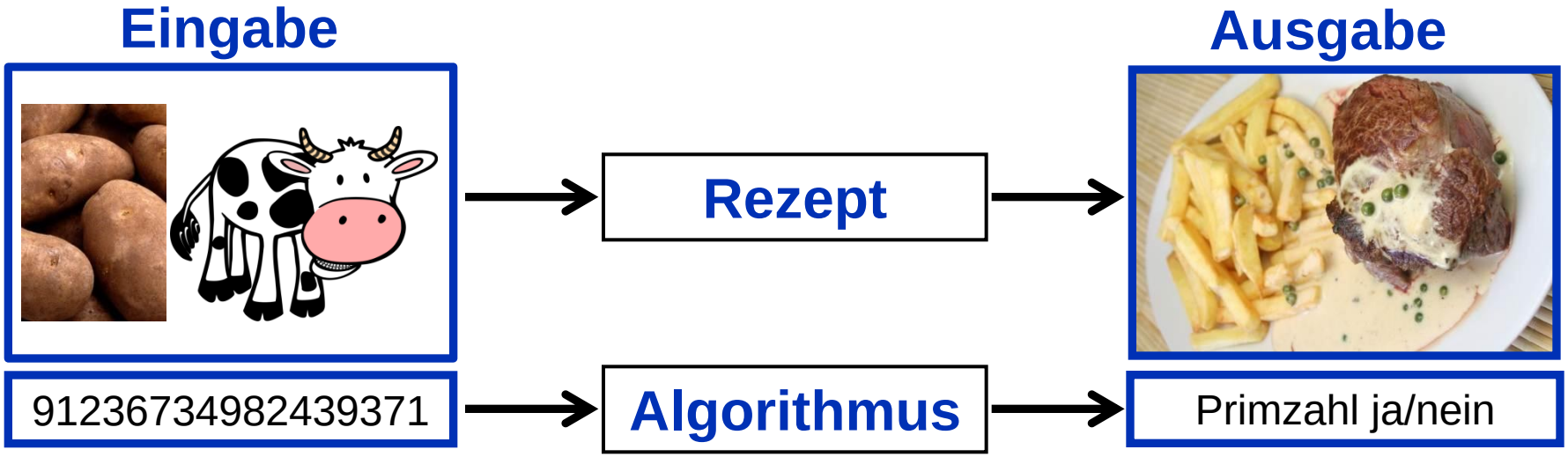
Betrachten wir lieber einige Teilgebiete:

- Algorithmen und Datenstrukturen
- Berechenbarkeitstheorie
- Komplexitätstheorie
- Automatentheorie
- Formale Sprachen
- Logik
- …

universitätbonn

Algorithmik

Algorithmen sind formal definierte Handlungsvorschriften, die von Computern wie ein Rezept zur Lösung eines Problems ausgeführt werden. Dabei wird eine Eingabe in eine Ausgabe transformiert.



Algorithmus zum Primzahltest:
Bei Eingabe n
teste alle Zahlen 2,3,4,…,n-1, ob sie Teiler von n sind.
Ausgabe: Nein falls Teiler gefunden, sonst Ja.

universitätbonn

Algorithmik (2)

Algorithmen begegnen uns täglich.

Navigationsgerät
Was ist der kürzeste Wege von A nach B?

Online Banking, Einkaufen im Internet
Wie werden die Daten ver- und entschlüsselt?

Google

Suchmaschinen
Wie sucht man in einer großen Datenmenge?

Paketdienste, Logistikdienstleister
In welcher Reihenfolge werden Kunden beliefert?

universitätbonn

Datenstrukturen

Ziel: Korrekte und **schnelle Algorithmen**.

Wichtiges Hilfsmittel: **Datenstrukturen**.

Wie sollen Daten gespeichert werden, um schnelle Ausführung wichtiger Operationen zu ermöglichen?

Beispiel: Suchen in einer großen Datenmenge.

→ Algorithmen und Berechnungskomplexität I

universitätbonn

Berechenbarkeitstheorie

Welche Probleme können von Computern gelöst werden?

```
void foo (int a) {
  int i = a;
  while (i != 10) {
    i++;
  }
}
```

Es wäre schön, wenn …

Compilerwarnung: Eine Endlosschleife kann auftreten.

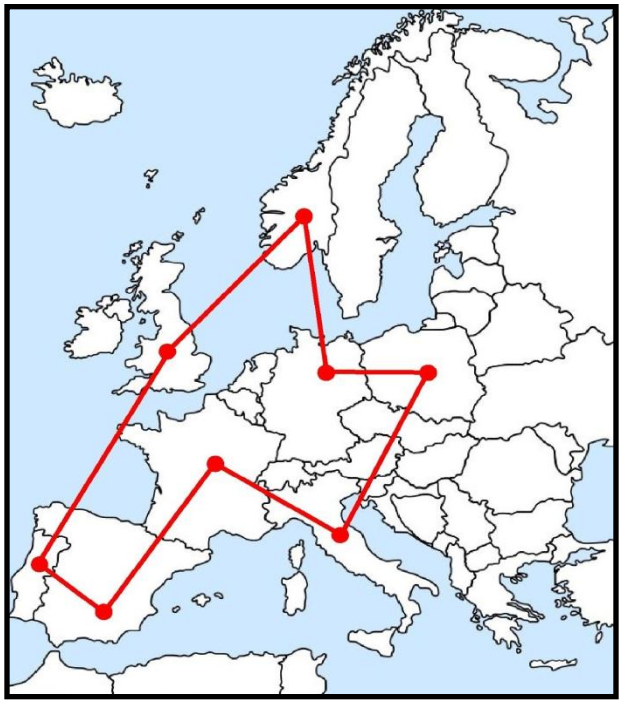
Ein Resultat der Berechenbarkeitstheorie besagt:
Es gibt keinen Compiler, der automatisch testet, ob ein gegebenes Programm in eine Endlosschleife geraten kann; egal, wie leistungsstark unsere Rechner auch sein mögen.

→ Algorithmen und Berechnungskomplexität II

universitätbonn

Komplexitätstheorie

Welche Ressourcen (z.B. Rechenzeit und Speicherplatz) sind nötig, um ein Problem zu lösen?



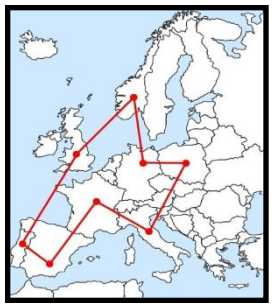
Problem des Handlungsreisenden:
Finde kürzeste Tour durch alle Städte.

Algorithmus: Teste alle Möglichkeiten und wähle die kürzeste Tour.
Keine gute Idee: Bei 20 Städten mehr als $6 \cdot 10^{16}$ Möglichkeiten.

Das Problem ist **NP-hart**, d.h. vermutlich, gibt es keinen effizienten Algorithmus.

universitätbonn

Komplexitätstheorie (2)



Es gibt zahlreiche natürliche **NP-harte Probleme**.
Entweder: Für alle gibt es effiziente Algorithmen.
Oder: Für keines gibt es effizienten Algorithmus.

Dies ist eine große ungelöste Frage (**P≠NP-Vermutung**).
Lösung ist 1 Million Dollar Wert.

Negative Ergebnisse sind hilfreich:
Existiert kein (effizienter) Algorithmus, so kann die Suche eingestellt werden.

→ Algorithmen und Berechnungskomplexität II

Automatentheorie und formale Sprachen

- Was ist ein gültiges Java/C++-Programm?
- Wie funktioniert ein Compiler?
- Wie muss eine formale Sprache (Programmiersprache) aussehen, damit ein effizienter Compiler existiert?

Grammatik:
Regelsystem zur Erzeugung syntaktisch korrekter Programme.

```
WhileStatement:
    while ( Expression ) Statement
BasicForStatement:
    for ( ForInit_opt ; Expression_opt ; ForUpdate_opt ) Statement
```

Automat:
Einfaches Rechnermodell, das bei Compilern zum Einsatz kommt, z.B. für die lexikalische Analyse (Zerlegung des Quelltextes in Tokens).

universitätbonn

Logik

- Was ist ein Beweis?
- Wie zieht man formal Schlüsse?
- Logik gehört eigentlich zur Mathematik, aber auch Informatiker müssen formal korrekte Beweise führen können.
- Aussagenlogik, Prädikatenlogik, …

SQL-Anfrage ≈ Formel der Prädikatenlogik erster Stufe

```
SELECT a.PersNr, a.Name
FROM Professor a
WHERE NOT EXISTS (SELECT * FROM Vorlesung WHERE PersNr = a.PersNr)
```

universitätbonn

Viel Spaß und Erfolg!

© Dr. Thomas Mauersberg / Uni Bonn

universitätbonn