



RWTHAACHEN
UNIVERSITY

Berechenbarkeit und Komplexität

Vorlesung im Wintersemester 22/23

Martin Grohe
Lehrstuhl Informatik 7

Folien bis auf kleine Änderungen von [Pascal Schweitzer](#), basierend auf früheren Folien von [Martin Grohe](#) und [Bertold Vöcking](#).

Vorlesung 1

Einführung

In BuK beschäftigen wir uns mit den
Grenzen der Berechenbarkeit.

Organisatorisches

Komponenten und Materialien

Die Vorlesung und alle dazugehörigen Veranstaltungen finden in Präsenz statt.

BuK hat folgende Komponenten

- ▶ Vorlesungen,
- ▶ Globalübungen,
- ▶ Kleingruppenübungen (Tutorien),
- ▶ Übungsblätter,
- ▶ E-Tests.

Alle Materialien und ein FAQ sind im Moodle-Raum zu finden.

Bitte sehen Sie regelmäßig in Moodle nach aktuellen Informationen.

Vorlesungen

- ▶ Die Vorlesung wird von **Martin Grohe** gehalten. Die Vorlesungen finden
 - ▶ Dienstags (ab 11.10.) um 16:30 Uhr im Großen Hörsaal (AM)
 - ▶ Mittwochs (ab 12.10.) um 08:30 Uhr im Großen Hörsaal (AM)
- statt.
- ▶ Die Vorlesungen werden von der **Video-AG der Fachschaft Informatik** aufgezeichnet.



Martin Grohe

Zulassungskriterien

Zulassung zur Klausur

- ▶ Es gibt 11 Übungsblätter zu je 15 Punkten.
 - ▶ Bonusblatt über Weihnachten
 - ▶ Zur Zulassung benötigt man je mindestens 50 % der Punkte aus den Blättern 1–6 (Berechenbarkeit) und aus den Blättern 7–11 (Komplexität).
- ▶ Es gibt 12 E-Tests zu je 10 Punkten.
 - ▶ Zur Zulassung benötigt man mindestens 50 % der Punkte aus den E-Tests.
 - ▶ Weiterhin müssen alle bis auf einen E-Test bearbeitet (mit mehr als 0 Punkten abgeschlossen) werden; bei bis zu 3 nicht bearbeiteten Tests muss ein Bonus-E-Test (nach den regulären E-Tests) bestanden werden.

Klausur

- ▶ Erster Termin: Dienstag, der 21.02.2023 um 08:30
- ▶ Zweiter Termin: Montag, der 27.03.2023 um 17:00
- ▶ Die Anmeldung ist ab dem 15.11. bis zum 15.01. möglich

Übungen

Übungsblätter

- ▶ Ausgabe wöchentlich montags bis 18:00 Uhr im Moodle, beginnend nächste Woche (17.10.)
- ▶ Abgabe in der folgenden Woche mittwochs um 14:30
- ▶ Bearbeitung in Gruppen zu 3 Studierenden
- ▶ Abgabe via Moodle als 1 PDF (max 15 MB) **mit Übungsgruppe, Namen und Matrikelnummern im Dokument**
 - ▶ Rückgabe der korrigierten Blätter über Moodle
 - ▶ Musterlösungen für die Hausaufgaben werden nach der Globalübung hochgeladen.
- ▶ Tutoriumsaufgaben werden in Tutorien **gemeinsam** erarbeitet. Es gibt *keine* Musterlösung für Tutoriumsaufgaben.

E-Tests

- ▶ Veröffentlichung und Abgabe montags um 18:00 Uhr, Bearbeitungszeit eine Woche
- ▶ Individuell zu bearbeiten

Globalübung

- ▶ Mittwochs (ab 26.10.) um 14:30 Uhr im Hörsaal H03 (CARL)
- ▶ Ausführliche Diskussion der abgegebenen Übung
 - ▶ Wie kommt man auf eine Lösung?
 - ▶ Wie sehen alternative Lösungswege aus?
 - ▶ Was sind häufige Fehler?

Diese Woche (12.10.): Besprechung Blatt 0, Unterstützung bei der Anmeldung, L^AT_EX-Einführung

Tutorien

- ▶ 22 Tutorien, verteilt auf Dienstag, Donnerstag und Freitag (siehe RWTHonline).
- ▶ Englischsprachiges Tutorium: Gruppe 19, Freitag um 14:30 im 5056
- ▶ Beginnen in der nächsten Woche (17.10.-21.10.).
- ▶ Anmeldung zu Tutorien in RWTHonline
 - ▶ Unabhängig von der Anmeldung zur Vorlesung
 - ▶ Prioritäten und Teams (siehe Bild rechts) in RWTHonline angeben
 - ▶ Abgabegruppen zu 3 Studierenden **aus dem selben Tutorium**

The screenshot shows the RWTHonline registration interface for tutor sessions. It displays three separate registration forms for different sessions:

- 12.53108 Berechenbarkeit und Komplexität**: VO | 4 SWS. Vorlesende*in: Grohe, Martin. A note says "Zur LV-Anmeldung".
- 12.53110 Berechenbarkeit und Komplexität - Übung**: TU | 2 SWS. Vorlesende*in: Thak, Eva. A note says "Zur LV-Anmeldung".
- 12.53110 Berechenbarkeit und Komplexität (Globalübung)**: UE | 2 SWS. Vorlesende*in: Thak, Eva. A note says "Zur LV-Anmeldung".

Below these, there is a larger panel for **12.53194 Berechenbarkeit und Komplexität - Übung**. It includes:

- A note: "Bitte wählen Sie mindestens 22 Gruppen aus 1 unterschiedlichen Lehrveranstaltungen."
- A note: "Bitte wählen Sie insgesamt 22 (derzeit 0) Gruppen aus, davon 1 (derzeit 0) aus unterschiedlichen Lehrveranstaltungen."
- A red box highlights the input field "Teamname (max. Teamgröße 3)" with the value "0/10".
- A note: "Alle zusammen"
- A note: "Freie Anmeldung"
- A note: "Präferenz bearbeiten"
- A note: "Hoch", "Mittel", "Niedrig"
- A note: "... alle anzeigen"
- A note: "Nachsenden Formular: DL 18.10.2022, 08:30 - 10:00
9522/2356/9523"

Anmeldefrist: Freitag, der 14. Oktober um 23:59.

Tutorien (forts.)

Abgabegruppen können in Moodle gebildet werden, sobald die Tutorien übertragen wurden.

1. Nutzt ggf. die Abgabegruppen-Börse eures Tutoriums, um eine Gruppe zu finden
 2. Tragt euch unter “Abgabegruppen” dann gemeinsam in eine freie Gruppe ein (siehe Bild)
-
- ▶ Ohne Gruppe ist keine Abgabe möglich!
 - ▶ Frist für Moodle-Gruppen: Montag, der 24.10. um 15:00 Uhr, erst danach Abgabe der Übungen möglich

Gruppenwahl	Gruppe	Beschreibungen anzeigen
<input type="radio"/>	Tutorium 01 Abgabegruppe 1	
<input type="radio"/>	Tutorium 01 Abgabegruppe 10	
<input type="radio"/>	Tutorium 01 Abgabegruppe 11	
<input type="radio"/>	Tutorium 01 Abgabegruppe 12	
<input type="radio"/>	Tutorium 01 Abgabegruppe 13	
<input type="radio"/>	Tutorium 01 Abgabegruppe 14	
<input type="radio"/>	Tutorium 01 Abgabegruppe 15	
<input type="radio"/>	Tutorium 01 Abgabegruppe 16	
<input type="radio"/>	Tutorium 01 Abgabegruppe 17	

Anmeldungen

Anmeldungen

Für die Vorlesung brauchen Sie drei unabhängige Anmeldungen

1. Anmeldeverfahren der Vorlesung: Zugang zum Moodle-Raum
2. Anmeldeverfahren Übungen: Zuordnung zu Tutorien
3. Anmeldung zur Klausur

Scheine und Auflagen

- ▶ Für einen Schein nimmt man wie alle anderen an der Klausur teil und muss auch die Zulassung erlangen.
- ▶ Studierende, die die Vorlesung als Master-Auflage absolvieren müssen, nehmen ebenfalls ganz normal an Übungen und Klausur teil. Das ZPA schaltet in diesem Fall das Anmeldeverfahren in RWTHonline gesondert frei.

Beteiligte

Dozent



Martin Grohe

Assistent*innen



Eva Fluck



Athena Riazsadri



Julia Feith

Tutor*innen

Tabea Hegewald, André Kuslido, Simon Wilmes, Lennard Schober, János Arpasi, Naomi Barth, David Philipp, Kevin Lu, Matthäus Micun, Yaman Faour, Imogen Hergeth, Jonas Groven, Georg Albrecht, Florian Cramer, Jannik Hiller, Larissa Meffert, Magnus Giesbert, Tomáš Novotný, Geonho Yun, Klara Pakhomenko, Arian Kulmer

Was tun bei Fragen?

1. Ist die Frage auch interessant für andere Studierende? Dann poste die Frage bitte ins Diskussionsforum im Moodle.
2. Für Fragen bezüglich der Korrektur wendet euch bitte an die Tutor*in eurer Kleingruppe.
3. Eure Tutor*innen sind auch ansonsten euer erster Anlaufpunkt.
4. Dann erst, oder in dringenden(!) Fällen E-Mail an:

buk@lists.rwth-aachen.de

Bitte keine E-Mail direkt an die Assistent*innen oder mich. Diese werden ungelesen gelöscht!

Wie können Sie mich erreichen?

- ▶ Immer nach der Vorlesung
- ▶ Sprechstunde Dienstags, 14–15 Uhr

Bitte keine E-Mail an mich und keine Telefonanrufe.

Buchempfehlungen



Uwe Schöning. [Theoretische Informatik - kurzgefasst](#).
Spektrum Akademischer Verlag, 2001.

Michael Sipser. [Introduction to the Theory of Computation](#), 3rd Edition. Cengage Learning, 2012.

Diese und noch weitere Bücher zum Thema sind zu finden in der Informatikbibliothek, im Handapparat unter BuK.

Zwei weiterführende Bücher sind

- ▶ Nigel J. Cutland. [Computability: An Introduction to Recursive Function Theory](#). Cambridge University Press, 1980.
- ▶ Sanjeev Arora, Boaz Barak. [Computational Complexity](#). Cambridge University Press, 2009.

Motivation und Übersicht

Rechenmaschinen



1672/1700

User:Kolossos/Wikimedia Commons/CC-BY-SA-3.0



1923

Greg Goebel/Wikimedia Commons/Public Domain



?



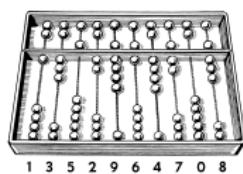
~2014



1980



2013



~3000 AC

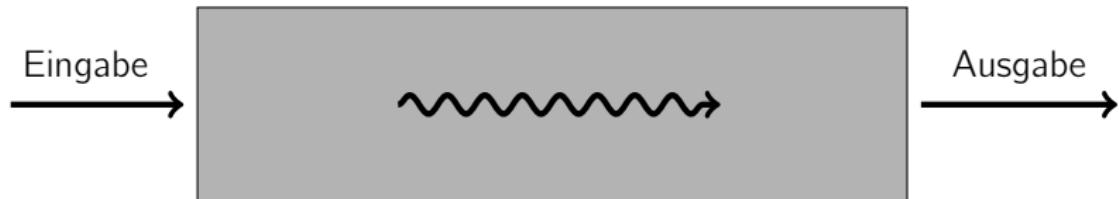
Pearson Scott Foresman/Wikimedia Commons/Public Domain



1983

Berechenbarkeit – die absoluten Grenzen des Computers

In der Vorlesung interessieren wir uns allgemein für Berechnungsprobleme.



Gibt es überhaupt Probleme, die wir nicht mit dem Computer lösen können?

Bessere Frage

Gibt es "algorithmische Probleme" (oder "Berechnungsprobleme"), die ein Computer nicht lösen kann?

Die absoluten Grenzen des Computers

Wir werden sehen:

- ▶ Es gibt keinen Algorithmus, der entscheidet, ob ein gegebenes Programm in einen bestimmten Zustand läuft.
(Error: 0E : 016F : BFF9B3D4.)
- ▶ Allgemein lässt sich die Funktionsweise von Programmen nur schwer algorithmisch überprüfen.
- ▶ Zum Beispiel gibt es keinen Algorithmus, der entscheidet, ob ein gegebenes Programm immer die Summe zweier eingegebenen Zahlen berechnet.

Windows

An error has occurred. To continue:

Press Enter to return to Windows, or

Press CTRL+ALT+DEL to restart your computer. If you do this,
you will lose any unsaved information in all open applications.

Error: 0E : 016F : BFF9B3D4

Press any key to continue _

Das Halteproblem

Allgemeines Halteproblem

- ▶ **Eingabe:** Ein Programm in einer wohldefinierten, universellen Programmiersprache (z.B. Java, C++, Python, Scala).
- ▶ **Frage:** Terminiert dieses Programm?

Wir werden beweisen, dass es keinen Algorithmus gibt, der dieses Problem entscheiden kann.

Komplexitätstheorie — die Grenzen der effizienten Berechenbarkeit

Frage:

Lässt sich ein gegebenes algorithmisches Problem **effizient** lösen, das heißt, in vernünftiger Laufzeit, mit vernünftigem Speicherbedarf und vernünftiger Verwendung anderer Ressourcen?

Beispiel 1: Zauberdodekaeder



Aufgabe

Löse den Dodekaeder.

Beispiel 2: Rush Hour



User:Welt-der-Form/Wikimedia Commons/CC-BY-SA-3.0

Aufgabe

Befreie das rote Auto aus dem Verkehrsstau.

Dieses algorithmische Problem (und ähnliche) lassen sich offensichtlich mit dem Computer lösen. Man kann zum Beispiel alle Möglichkeiten durchprobieren.

Leider lassen sie sich (bisher) oft nur durch extrem ineffiziente Algorithmen lösen.

Beispiel 3: Traveling Salesperson

Aufgabe

Finde eine kurze Rundreise durch die größten deutschen Städte und zurück zum Ausgangsort.

Die angegebene Route ist die kürzeste von 43.589.145.600 möglichen.

Wieder kann man das Problem lösen, indem man alle Möglichkeiten durchprobiert, wieder ist dies extrem ineffizient.

Die Suche nach einem effizienten Algorithmus für das Problem begann schon vor fünfzig Jahren . . .



Traveling Salesperson (Forts.)

Traveling Salesperson Problem (TSP)

- ▶ Eingabe: vollständiger Graph G mit Kantenlängen
- ▶ Ausgabe: eine Rundreise, die alle Knoten in G besucht und dabei so kurz wie möglich ist

Wir werden zeigen, dass es unter einer gewissen Hypothese ($P \neq NP$) **keinen effizienten** Algorithmus für dieses Problem gibt.

Teil 1: Grundlagen

- ▶ Modellierung von Problemen
- ▶ Einführung der Turingmaschine (TM)
- ▶ Einführung der Registermaschine (RAM)
- ▶ Vergleich TM – RAM
- ▶ Church-Turing-These

Teil 2: Berechenbarkeit

- ▶ Existenz unentscheidbarer Probleme
- ▶ Unentscheidbarkeit des Halteproblems
- ▶ Diagonalisierung / Unterprogrammtechnik / Reduktion
- ▶ Hilberts zehntes Problem
- ▶ Das Postsche Korrespondenzproblem
- ▶ WHILE- und LOOP-Programme

Teil 3: Komplexität

- ▶ Die Komplexitätsklassen P und NP
- ▶ NP-Vollständigkeit und der Satz von Cook und Levin
- ▶ Kochrezept für NP-Vollständigkeitsbeweise
(Polynomielle Reduktion)
- ▶ NP-Vollständigkeit zahlreicher Probleme
- ▶ Auswege aus der NP-Vollständigkeit