Computerorientierte Mathematik I

Computerorientierte Mathematik I

Willkommen

Computerorientierte Mathematik I

Willkommen

Vorlesung: Prof. Dr. Ralf Kornhuber

Übungen: Assisstentin: Maren-Wanda Wolf

Tutorien: Anna Dittus, Tim Dittmann, Simon-Philipp Merz

Scheinkriterien

- regelmäßige Teilnahme (Anwesenheit + Vorrechnen)
- aktive Teilnahme (60% theoretische und 60% praktische Aufgaben)
- Klausur oder Nachklausur (bestimmt die Note)

Übungsaufgaben

- Jeweils am Freitag als .pdf im auf der Vorlesungs-Homepage
- Besprechung im Tutorium und Bearbeitung bis Donnerstag in 13 Tagen
  - Bearbeitung in Gruppen (max. drei Mitglieder)
  - theoretische und praktische Übungsaufgaben
  - Ziel: aktives Verständnis der Lehrinhalte

Klausur

- Klausurtermin: 5. Februar 2016, 12-14 Uhr
- Nachklausurtermin: 15. April 2016.
  - die bessere beider Noten zählt (für Studierende nach StO/PO 2013)
  - theoretische und praktische Aufgaben
  - Orientierung an Übungsaufgaben

Klausur

- Klausurtermin: 5. Februar 2016, 12-14 Uhr

- Nachklausurtermin: 15. April 2016.
  - die bessere Note zählt (für Studierende nach Sto/PO 2013)
  - theoretische und praktische Aufgaben
  - Orientierung an Übungsaufgaben

Hilfsmittel bei der Klausur:

Ihre schriftlichen Unterlagen (Skript, Bücher, Übungsaufgaben, ...)

Termine und alles andere

Homepage: http://numerik.mi.fu-berlin.de/wiki/WS_2015/CoMaI.php

z.B. suchen nach: 'Wolf Kornhuber Computerorientierte Mathematik'
$\longrightarrow$ FU Berlin, Vorlesungsverzeichnis, Bachelor Mathematik

Nur in der 2. Woche: Praktische Übungen am Rechner

Rechnertutorien: 20.10. - 22.10. im Raum 017 der A6

Dienstag	20.10.2015	8-10 Uhr
Dienstag	20.10.2015	10-12 Uhr
Dienstag	20.10.2015	12-14 Uhr
Dienstag	20.10.2015	14-16 Uhr
Mittwoch	21.10.2015	8-10 Uhr
Mittwoch	21.10.2015	10-12 Uhr
Donnerstag	22.10.2015	8-10 Uhr
Donnerstag	22.10.2015	10-12 Uhr

Nur in der 2. Woche: Praktische Übungen am Rechner

Rechnertutorien: 20.10. - 22.10. im Raum 017 der A6

Dienstag	20.10.2015	8-10 Uhr
Dienstag	20.10.2015	10-12 Uhr
Dienstag	20.10.2015	12-14 Uhr
Dienstag	20.10.2015	14-16 Uhr
Mittwoch	21.10.2015	8-10 Uhr
Mittwoch	21.10.2015	10-12 Uhr
Donnerstag	22.10.2015	8-10 Uhr
Donnerstag	22.10.2015	10-12 Uhr

Achtung: MATLAB-Steilkurs im Erstsemester-Mentoring Mathematik

Was und wozu?

Was:

... passieren kann,
wenn man mit Zahlen rechnet (nicht mit Buchstaben!)
und warum.

Wozu:

Verbindung von Analysis, Linearer Algebra und Numerik

Was und wozu?

Was:

... passieren kann,
wenn man mit Zahlen rechnet (nicht mit Buchstaben!)
und warum.

Wozu:

Verbindung von Analysis, Linearer Algebra und Numerik

Möglichkeiten und Grenzen der numerischen Simulation

Der Orkan Lothar

Aus dem Logbuch des Meteorologen Gaudenz Truog:

(verantwortlicher Schichtleiter MeteoZürich am 26. Dezember 1999)

26. Dezember 1999, 07.20h:

“*Die Bodenbeobachtungen von 06 GMT laufen ein,*
Hauptaugenmerk Frankreich.
In Rouen 25.8 hPa Druckabfall in drei Stunden!
So etwas habe ich noch nie gesehen über dem Kontinent.
Damit wird endgültig klar, dass sich Ausserordentliches anbahnt”

Der Orkan Lothar

Was ist passiert?

Am 26.12. 1999 zog Lothar über Europa:

- Windgeschwindigkeiten bis 272 km/h

- etwa 110 Todesopfer
- Versicherungsschaden ca. 6 Mrd. USD

Der Orkan Lothar

Was ist passiert?

Am 26.12. 1999 zog Lothar über Europa:

- Windgeschwindigkeiten bis 272 km/h
- etwa 110 Todesopfer
- Versicherungsschaden ca. 6 Mrd. USD

Offizielle Warnungen fast zeitgleich mit dem Eintreffen des Orkans!

Wie konnten die Wettersimulationen Lothar übersehen?

“schlechte” Datenlage:

Wetterballon geplatzt, repariert, zu spät!

Kaum Messstationen über dem Atlantik.

TRAJECTOIRE DE LA TEMPÊTE "LOTHAR" 1999-12-26

Wie konnten die Wettersimulationen Lothar übersehen?

“schlechte” Datenlage:

Wetterballon geplatzt, repariert, zu spät!

Kaum Messstationen über dem Atlantik.

TRAJECTOIRE DE LA TEMPÊTE "LOTHAR" 1999-12-26

ungewöhnlicher Typ von Sturm:

Extremer Druckabfall erst sehr spät messbar.

Wie konnten die Wettersimulationen Lothar übersehen?

"schlechte" Datenlage:

Wetterballon geplatzt, repariert, zu spät!

Kaum Messstationen über dem Atlantik.

TRAJECTOIRE DE LA TEMPÊTE "LOTHAR" 1999-12-26

ungewöhnlicher Typ von Sturm:

Extremer Druckabfall erst sehr spät messbar.

Kleine Datenfehler haben große Wirkungen!

Molekulare Simulationen

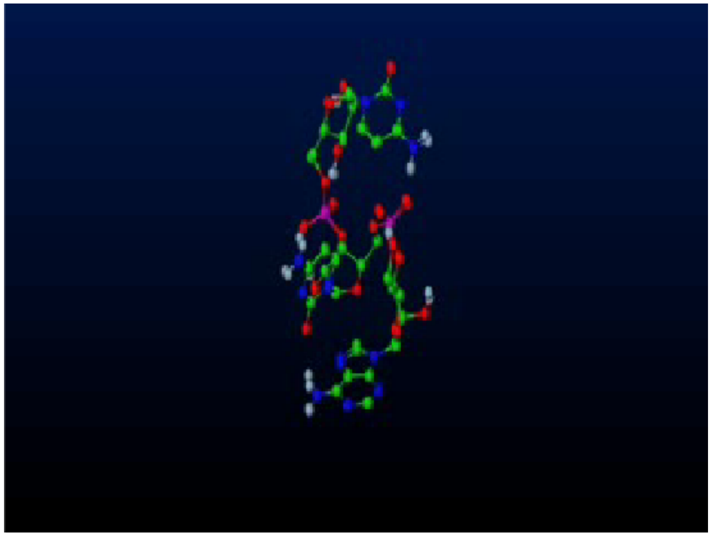
Nobelpreis in Chemie 2013 an M. Karplus, M. Levitt, A. Warshel

"... for the development of multiscale models for complex chemical systems".

Molekulare Simulationen

Nobelpreis in Chemie 2013 an M. Karplus, M. Levitt, A. Warshel

"... for the development of multiscale models for complex chemical systems".



Kleine Änderungen der Anfangsdaten haben große Wirkungen!

Kleine Ursache, große Wirkung!

Kondition

eines Problems

Können Computer rechnen?

Definition der Multiplikation: (Grundschule)

$$\underbrace{0.1 + 0.1 + 0.1 + \cdots + 0.1 + 0.1}_{m \text{ mal}}$$

Können Computer rechnen?

Definition der Multiplikation: (Grundschule)

$$\underbrace{0.1 + 0.1 + 0.1 + \cdots + 0.1 + 0.1}_{m \text{ mal}} = m * 0.1$$

Können Computer rechnen?

Definition der Multiplikation: (Grundschule)

$$\underbrace{0.1 + 0.1 + 0.1 + \cdots + 0.1 + 0.1}_{m \text{ mal}} = m * 0.1$$

MATLAB-Programm: `sumu(m)`

```
S = 0; for i=1:1:m S = S + 0.1; end; S

M = m*0.1

if S == M disp('Richtig!') else disp('Falsch!') end
```

Nein, Computer rechnen nicht richtig!

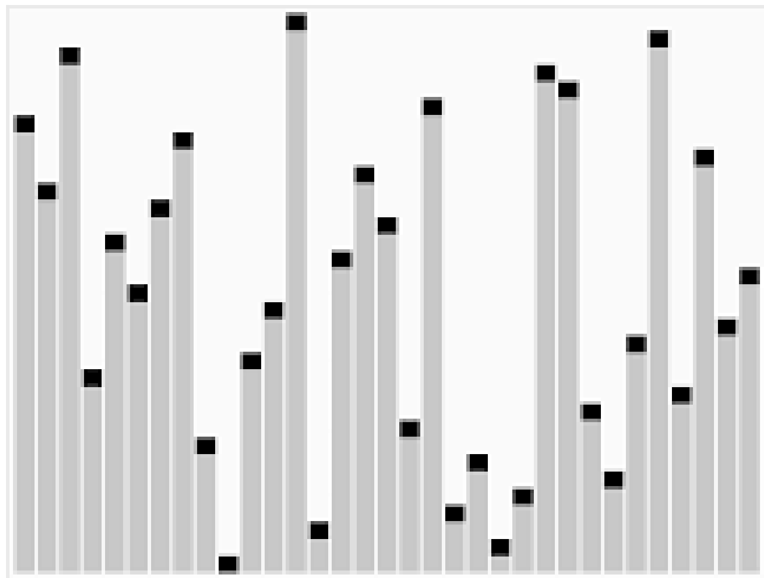
Stabilität

eines Algorithmus'

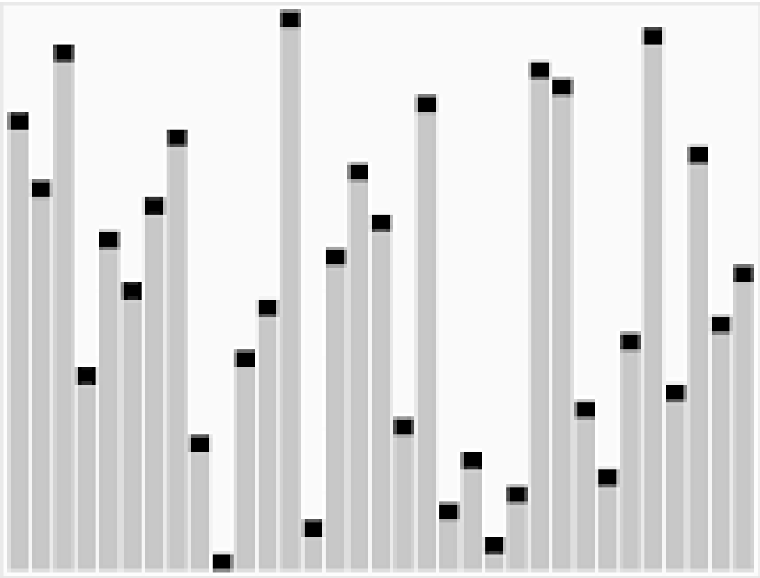
Komplexität

eines Problems

Beispiel: Sortieren von n Zahlen



Beispiel: Sortieren von n Zahlen



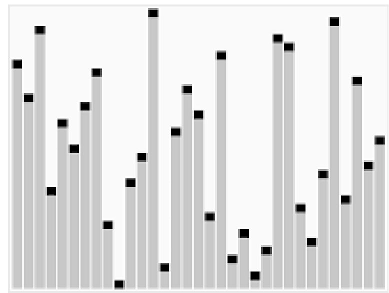
Theorem: Zu jedem Algorithmus gibt es eine Reihenfolge der n Zahlen, für die er mindestens $c\, n \log(n)$ Vergleiche benötig!

Rechenaufwand

Effizienz

eines Algorithmus'

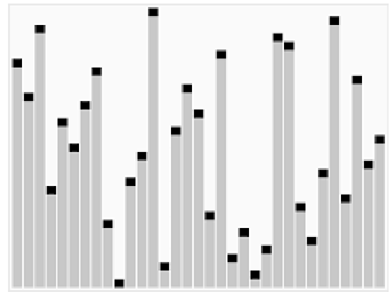
Beispiel: Sortieren von n Zahlen



Algorithmus: `bubbleSort`

Es gibt eine Reihenfolge der n Zahlen,
für die `bubbleSort` mindestens Cn^2 Vergleiche benötig!

Beispiel: Sortieren von n Zahlen



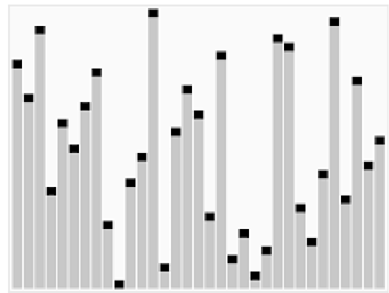
Algorithmus: `bubbleSort`

Es gibt eine Reihenfolge der n Zahlen,
für die `bubbleSort` mindestens Cn^2 Vergleiche benötig.

Algorithmus: `mergeSort`:

Für alle möglichen Reihenfolgen der n Zahlen
benötigt `mergeSort` höchstens $Cn\log(n)$ Vergleiche.

Beispiel: Sortieren von n Zahlen



Algorithmus: `bubbleSort`
Es gibt eine Reihenfolge der n Zahlen,
für die `bubbleSort` mindestens Cn^2 Vergleiche benötigt.

Algorithmus: `mergeSort`:
Für alle möglichen Reihenfolgen der n Zahlen
benötigt `mergeSort` höchstens $Cn\log(n)$ Vergleiche.

Illustration: https://www.youtube.com/watch?v=ZZuD6iUe3Pc