

**Grundlagen der Programmierung
und Algorithmik**

Vorlesung mit Übungen

Martin Lehmann

Wintersemester 2009/2010

Literatur:

Informatik:

**Gumm, Heinz-Peter und Sommer, Manfred:
Einführung in die Informatik, 8., vollst.
überarb. Aufl., München, Oldenbourg, 2009,
ISBN: 3-486-58724-2, 978-3-486-58724-1**

Uwe Schöning:

**Ideen der Informatik: grundlegende Modelle
und Konzepte der theoretischen Informatik,
3., korr. Aufl., München, Oldenbourg, 2008,
ISBN: 978-3-486-58723-4**

Manfred Broy:

**Informatik: eine grundlegende Einführung
Bd. 1: Programmierung und Rechner-
strukturen**

**Bd. 2: Systemstrukturen und theoretische
Informatik**

**3., überarb. Aufl, Berli, Springer, 2009,
ISBN: 3-540-21417-8, 3-540-21418-6**

Anmerkung: Erscheint September 2009 ??

Carsten Vogt:
Informatik: eine Einführung in Theorie und Praxis,
1. Aufl., Heidelberg, Spektrum Akad. Vl., 2004,
ISBN: 3-8274-1392-3

C++:

Bjarne Stroustrup:
The C++ programming language,
Special ed., 3rd ed., 6th print,
Boston, Addison-Wesley, 2002,
ISBN: 0-201-70073-5

Ulrich Breymann:
Der C++-Programmierer: C++ lernen,
professionell anwenden, Lösungen nutzen,
München, Hanser, 2009,
ISBN: 978-3-446-41644-4

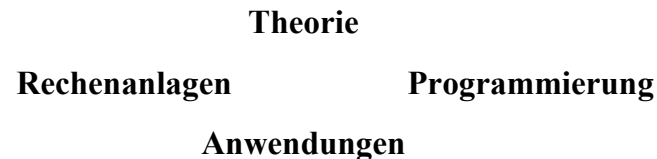
Bjarne Stroustrup:
Programming: principles and practice
using C++
Boston, Mass., Addison-Wesley, 2009,
ISBN: 0-321-54372-6, 978-0-321-54372-1

- 1 Einleitung**
- 1.1 Was ist Informatik?**
- 1.2 Leistungssteigerung von Rechnern**
- 1.3 Schätzungen von Reddy und Newell**
- 1.4 Turing Test und Eliza**
- 1.5 Euklids Algorithmus**
- 1.6 Zusammenfassung**
- Anhang: Präfixe für Maßeinheiten**

Was ist Informatik?

Informatik ist die Wissenschaft von der systematischen Verarbeitung von Informationen. Sie befaßt sich mit der Struktur, den Eigenschaften und den Beschreibungsmöglichkeiten von Information und informationsverarbeitenden Systemen.

Die vier Schwerpunkte der Informatik:



Teilgebiete der Informatik:

**Theoretische Informatik,
Praktische Informatik,
Angewandte Informatik,
Technische Informatik.**

Das Wort "Informatik":

Das Wort Informatik ist ein Kunstwort, zusammengesetzt aus den Wörtern Information und Automatik. In Deutschland wurde es wahrscheinlich erstmals von Karl Steinbuch im Jahre 1957 benutzt. In dem Aufsatz "INFORMATIK: Automatische Informationsverarbeitung, SEG-Nachrichten 1957, Heft 4, S. 171-176" beschreibt Steinbuch den Umfang eines Gebietes Informatik. 1968 umreißt Forschungsminister Gerhard Stoltenberg in einer Berliner Rede ein Förderprogramm für eine neue Wissenschaft, der er den Namen Informatik gibt. Hierbei stützt er sich auf das französische Wort "Informatique", das 1962 von Philippe Dreyfus zur Bezeichnung der Unternehmung "Société d'Informatique Appliquée" genutzt wurde. 1967 erfolgt eine Definition des Begriff Informatique durch die Académie Française.

Auswahl aus den Bindestrich-Informatiken:

**Bio-Informatik,
Chemo-Informatik,
Geo-Informatik,
Medieninformatik,
Medizin-Informatik,
Quanteninformatik,
Rechtsinformatik,
Umwelt-Informatik,
Wirtschafts-Informatik.**

In ihrem Buch "Parallel Computers 2" geben Hockney und Jesshope einen Faktor von 10 in 5 Jahren für die Steigerung der Rechengeschwindigkeit von Computern an. Diesen Wert leiten sie ab aus der Veränderung der Ausführungsgeschwindigkeit der Multiplikationsoperation für Fließkommazahlen im Zeitraum 1950 bis 1975.

Jahr	Steigerung
0	1,00
1	1,58
2	2,51
5	10
6	15,85
7	25,12
10	100
11	158,49
12	251,19
15	1.000
20	10.000
30	1.000.000
40	100.000.000
50	10.000.000.000
60	1.000.000.000.000

Für die Entwicklung der Taktzyklen geben Hockney und Jesshope folgende Zahlen:

Rechner	Jahr	Zykluszeit
EDSAC-1	1949	2000 ns
ACE	1951	1000 ns
CDC-6600	1964	100 ns
CDC-7600	1969	27,5 ns
CRAY-1	1976	12,5 ns
CRAY-2	1984	4 ns

Leistungssteigerung der Spitzenrechner seit 1993:

Jahr	Rechner	Linpackleistung in Gflop/s	Zahl der Kerne
1993	CM-5, TMC	60	1.024
1994	Intel Paragon XP/S MP	143	3.680
1995	Fujitsu NWT	170	140
1996	Hitachi SR2201/1024	220	1.014
1997	Intel ASCI Red	1.068	7.264
1998	Intel ASCI Red	1.338	9.152
1999	Intel ASCI Red	2.121	9.472
2000	Intel ASCI Red	2.379	9.632
2001	ASCI White, IBM	7.226	8.192
2002	Earth Simulator, NEC	35.860	5.120
2003	Earth Simulator, NEC	35.860	5.120
2004	Earth Simulator, NEC	35.860	5.120
2005	Blue Gene/L, IBM	136.800	65.536
2006	Blue Gene/L, IBM	280.600	131.072
2007	Blue Gene/L, IBM	280.600	131.072
2008	Roadrunner, IBM	1.026.000	122.400
2009	Roadrunner, IBM	1.105.000	129.600

Daten aus den TOP500-Listen, jeweils vom Juni, durchschnittlicher Jahresfaktor etwa 1,85.

Eine Version von "Moore's Law":

$$(\text{circuits per chip}) = C_{1975} * 2^{(\text{year} - 1975) / 1,5}$$

Allgemeine Interpretation:

Die zu einem festen Preis kaufbare Rechenleistung verdoppelt sich alle 18 Monate.

Ursprüngliche Formulierung von Gordon E. Moore im Artikel: "Cramming More Components Onto Integrated Circuits" im Magazin Electronics (38, 8, April 19, 1965):

"The complexity for minimum component costs has increased at a rate of roughly a factor of two per year. Certainly over the short term this rate can be expected to continue, if not to increase. Over the longer term, the rate of increase is a bit more uncertain, although there is no reason to believe it will remain nearly constant for at least 10 years."

Tabelle zur erwarteten Steigerung der Rechenleistung:

Jahr 1:	1,5874
Jahr 2:	2,1984
Jahr 3:	4
Jahr 4:	6,3496
Jahr 5:	10,0794
Jahr 6:	16
Jahr 7:	25,3984
Jahr 8:	40,3175
Jahr 9:	64

Illustration zu Moores Beobachtung:

Beispiel: Entwicklung der Intelprozessoren

Jahr	Rechner	Transistorenzahl
1971	4004	2.250
1972	8008	2.500
1974	8080	5.000
1978	8086	29.000
1982	286	120.000
1985	i386	275.000
1989	i486	1.180.000
1993	Pentium	3.100.000
1997	Pentium II	7.500.000
1999	Pentium III	24.000.000
2000	Pentium 4	42.000.000
2002	Itanium	220.000.000
2003	Itanium 2	410.000.000
2004	Itanium 2 (9MB Cache)	592.000.000
2005	Pentium D	291.000.000
2006	Dual-Core Itanium 2	1.720.000.000
2007	Quad-Core Intel	820.000.000

Kennzahlen eines Hochleistungsrechners:

Name: NEC Earth Simulator

Jahr: 2002

Zahl der Prozessoren: 5120
aufgeteilt auf 640 Knoten zu je 8 Prozessoren,
Zykluszeit eines Prozessors 2 ns,
Leistung eines Prozessors 8 GFlop/s.

Hauptspeicher: 10 TiB,
pro Knoten 16 GiB gemeinsamer Speicher.

Plattenspeicher: 640 TB (415 TB System, 225 TB Nutzer)

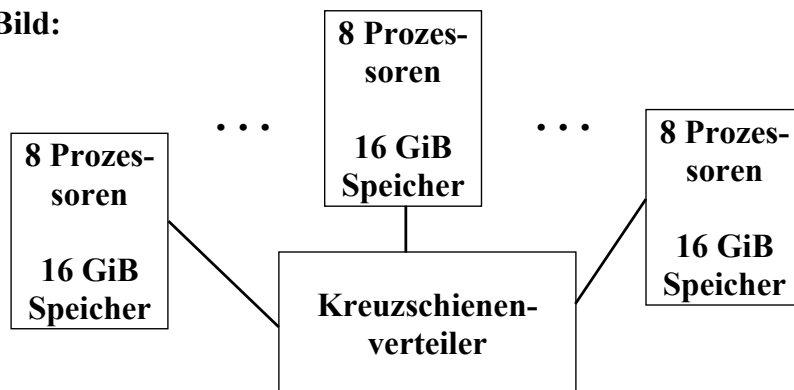
Massenspeicher: 1,5 PB

Theoretische Gesamtleistung: 40 TFlop/s

Erreichbare Gesamtleistung: 36 TFlop/s

Verbindungsnetz: Kreuzschienenverteiler

Bild:



Kennzahlen eines Hochleistungsrechners:

Name: NEC Earth Simulator 2

Jahr: 2009

Zahl der Prozessoren: 1280
aufgeteilt auf 160 Knoten zu je 8 Prozessoren,
Arbeitsfrequenz eines Prozessors 3,2 GHz,
Leistung eines Prozessors 102,4 GFlop/s.

Hauptspeicher: 20 TiB,
pro Knoten 128 GiB gemeinsamer Speicher.

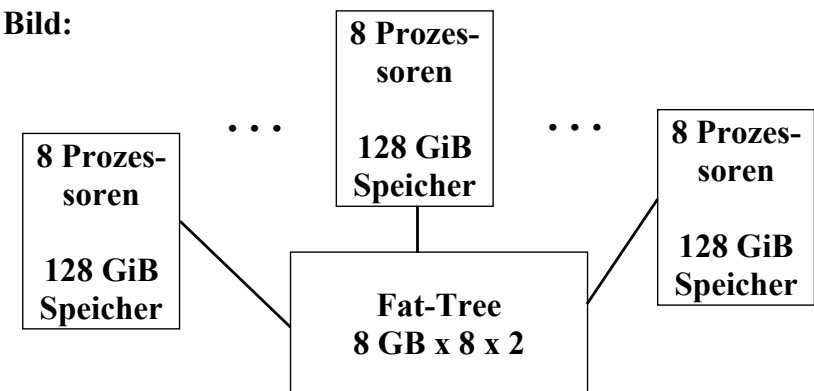
Theoretische Gesamtleistung: 131 TFlop/s

Erreichbare Gesamtleistung: 122 TFlop/s

Effizienz: 93 %

Verbindungsnetz: "Fat-Tree" - Netzwerk

Bild:



Geschwindigkeitssteigerung beim Problemlösen:

**Tabelle aus Raj Reddy und Allen Newell
Multiplicative Speedup of Systems
In Perspectives on Computer Science (A. K. Jones);
Academic Press, 1977.**

Beobachtete Steigerungsfaktoren der Ausführungsgeschwindigkeit von Programmen auf Grund von Spartenfortschritten:

Technologie	20 bis 50
Architektur	2 bis 10
Systemsoftware	2 bis 10
Programmorganisation	2 bis 20
Algorithmen	2 bis 20
Reimplementation	2 bis 10
Problemwissen	10 bis 100
Heuristiken	10 bis 100

Bemerkung: Falls die einzelnen Steigerungen unabhängig voneinander sind, sind Geschwindigkeitssteigerungen um bis zu elf Zehnerpotenzen möglich.

Turing Test:

Können Computer denken?

Im Jahre 1950 schlug Alan Turing zur Beantwortung der obigen Frage folgende Testsituation vor. Ein Mensch führt ein Gespräch mit einem unsichtbaren Gegenüber; kann er nicht aufgrund des Gespräches entscheiden, ob sein Gesprächspartner ein Mensch oder eine Maschine ist, dann ist dem Gegenüber menschliche Intelligenz zuzusprechen.

Turing vermutete, daß ein Rechner mit einer Speicherkapazität von 1 Gigabit, dessen Existenz er für das Jahr 2000 erwartete, einen Durchschnittsmenschen in 70 % der Fälle täuschen würde.

Als Turing Test läßt sich das Programm Eliza ansehen, das Joseph Weizenbaum um das Jahr 1963 schuf. Später schrieb er dazu:

"I was startled to see how very deeply people conversing with the program became emotionally involved with the computer and how unequivocally they anthropomorphized it. Once my secretary, who had watched me work on the program for many months and therefor surely knew it to be merely a computer program, started conversing with it. After only a few exchanges, she asked me to leave the room."

Eine Anekdote von Daniel Bobrow:

One Saturday morning about 9 a.m., a Vice President of Bolt, Beranek, and Newman in charge of selling our Telcomp commercial service arrived at our PDP-1 computer room hoping to be able to use the computer for a demonstration. He found a note on the PDP-1 teletype that read: "TYPE ALL MESSAGES INTO THIS TELETYPE AND END THEM WITH A PERIOD." This note was left there by one of the programming bums who had just succeeded in making run a LISP version of the DOCTOR program (based on ELIZA as developed by Weizenbaum). However, the VP thought that this note indicated that this teletype was linked with the one in my home, since I often use the computer all night. Therefore, he started typing to "me," but in fact was typing to this version of the DOCTOR program. The following conversation ensued:

VP: Do you think that I can use the computer this morning?

Eliza: Why do you ask?

VP: I might be able to make some additional Telcom sales.

Eliza: Why aren't you so sure?

VP: My perspective customers have never seen a demonstration of the system.

Eliza: Does this mean very much to you?

VP: Of course it does.

Eliza: You seem very positive about that.

VP: This is ridiculous.

Eliza: What do you mean, this is ridiculous?

VP: Is there any reason why I can't use the computer now?

Eliza: Tell me more about your perspective customers.

VP: Please dial me up on 491-1850

Note that after the last remark the VP did not obey instructions, and left out the period. Therefore, of course, the computer didn't answer him. This so infuriated the VP, who thought I was playing games with him, that he called me up, woke me from a deep sleep, and said:

VP: Why are you being so snotty to me?

Bobrow: What do you mean I am being snotty to you?

The VP angrily read me the dialog that "we" had been having, and couldn't get any response but laughter from me. It took a while to convince him it really was a computer.

Ein Beispieldialog mit Eliza:

H: *Men are all alike.*
Eliza: In what way?
H: *They're always bugging us about something or other.*
Eliza: Can you think of a specific example?
H: *Well, my boyfriend made me come here.*
Eliza: Your boyfriend made you come here?
H: *He says I'm depressed much of the time.*
Eliza: I am sorry to hear that you are depressed.
H: *It's true. I am unhappy.*
Eliza: Do you think coming here will help you not to be unhappy?
H: *I need some help, that much seems certain.*
Eliza: What would it mean to you if you got some help?
H: *Perhaps I could learn to get along with my mother.*
Eliza: Tell me more about your family.
H: *My mother takes care of me.*
Eliza: Who else in your family takes care of you?
H: *My father.*
Eliza: Your father.
H: *You are like my father in some ways.*
Eliza: What resemblance do you see?
H: *You are not very aggressive but I think you don't want me to notice that.*
Eliza: What makes you think I am not very aggressive?
H: *You don't argue with me.*
Eliza: Why do you think I don't argue with you?
H: *You are afraid of me.*
Eliza: Does it please you to believe I am afraid of you?
H: *My father is afraid of everybody.*
Eliza: What else comes to your mind when you think of your father?
H: *Bullies.*
Eliza: Does that have anything to with the fact that your boyfriend made come you here?

Quelle: Weizenbaum, J. (1976): Computer Power and Human Reason, Freeman, New York.

Zum Aufbau von Eliza:

Eliza besteht aus zwei Teilen, einem allgemeinen Verwaltungsteil und einem austauschbaren Rollenskript. Das Skript enthält zu einer Rolle passende Schlüsselwörter. Für die Rolle des Psychotherapeuten sind dies Wörter wie Mutter, Vater, Traum, Angst, u. s. w. Das Programm Eliza sucht nun in der Eingabe nach dem Auftreten dieser Wörter; es führt eine Tabelle mit vorgestanzten Antwortmustern und nutzt diese zur Bildung intelligent klingender Antworten.

Beispiel:

Muster: ?x my mother ?y

Antworten: Who else in your family ?y?
 Tell me more about your family.

Möglicher Eingabesatz: Yesterday I was miserable when
 my mother told me I was a disappointment.

Antwortsatz nach Anreideanpassung: Who else in your
family told you you were a disappointment?

Findet Eliza kein Schlüsselwort in der Eingabe, dann benutzt es gesprächserhaltende Aussagen wie:
 What makes you say that?
 Could you be more specific?
 ...

Euklids Algorithmus zur Berechnung des größten gemeinsamen Teilers GGT natürlicher Zahlen:

Seien $x, y \in \mathbb{N}$, $y \neq 0$, $x \geq y$,
dann $\exists \alpha \in \mathbb{N}$ mit $x = \alpha y + z$ und $y > z$,
damit $\text{GGT}(x, y) = \text{GGT}(y, z)$.

Beispiel:

3220 : 79 = 40 Rest 60,
79 : 60 = 1 Rest 19,
60 : 19 = 3 Rest 3,
19 : 3 = 6 Rest 1,
3 : 1 = 3 Rest 0.

Damit $\text{GGT}(3220, 79) = 1$.

Formulierung des Algorithmus:

Seien $a, b \in \mathbb{N}$,

$x := a$

$y := b$

Solange $y \neq 0$ führe aus

$z := \left\lfloor \frac{x}{y} \right\rfloor$ "ganzzahliger Anteil von $\frac{x}{y}$ "

$r := x - z * y$

$x := y$

$y := r$

$x = \text{GGT}(a, b)$

Bemerkung: (i) Algorithmus terminiert.
(ii) Beweis der Korrektheit einfach.

// Euklids Algorithmus als C++-Programm
// Eine erste verbesserungswürdige Version

```
#include <iostream>
using namespace std;
```

```
int ggt (int, int); // Größter gemeinsamer Teiler
```

```
int main () {
```

```
    int a = 1; // Erste natürliche Zahl
```

```
    int b = 1; // Zweite natürliche Zahl
```

```
    cerr << "Programm berechnet den größten "  
          "gemeinsamen Teiler zweier natürlicher "  
          "Zahlen"
```

```
    << endl << endl
```

```
    << "Abbruch durch b = 0" // fragwürdig
```

```
    << endl << endl;
```

```
while (b != 0) {
```

```
    cerr << "Erste Zahl a = ";
```

```
    cin >> a;
```

```
    cerr << "Zweite Zahl b = ";
```

```
    cin >> b;
```

```
    int c = ggt (a, b);
```

```
    cout << "GGT ("
```

```
        << a << ", " << b << ") = "
```

```
        << c
```

```
        << endl << endl;
```

```
    }
```

```
}//main
```

```

int ggt (int x, int y) {
    // Bei fehlerhafter Eingabe wird -2 zurückgegeben.
    if (x < 0)
        return -2;
    if (y <= 0)
        // y = 0 wird hier als fehlerhafte Eingabe betrachtet!
        return -2;
    while (y != 0) {
        int z = x / y;
        int r = x - z * y;
        x = y;
        y = r;
    }
    // Bemerkung: Schleife läßt sich vereinfachen
return x;
} //ggt

```

Beispiel eines Programmlaufs:

Programm berechnet den größten gemeinsamen Teiler zweier natürlicher Zahlen

Abbruch durch b = 0

**Erste Zahl a = 56
Zweite Zahl b = 16
GGT (56, 16) = 8**

**Erste Zahl a = 103
Zweite Zahl b = 19
GGT (103, 19) = 1**

**Erste Zahl a = 19
Zweite Zahl b = 103
GGT (19, 103) = 1**

**Erste Zahl a = 36
Zweite Zahl b = 300
GGT (36, 300) = 12**

**Erste Zahl a = -36
Zweite Zahl b = 300
GGT (-36, 300) = -2**

**Erste Zahl a = 0
Zweite Zahl b = 8
GGT (0, 8) = 8**

**Erste Zahl a = 8
Zweite Zahl b = 0
GGT (8, 0) = -2**

Bemerkung: Die letzten beiden Fälle sollten möglichst das gleiche Resultat liefern. Die Korrektur des C++-Programms ist einfach.

1.6 Zusammenfassung:

- **Reale Rechner sind nicht so leistungsfähig, wie man sie sich wünscht.**
- **Es ist einfach, auf einem Rechner beschränkte Verhaltensweisen eines Menschen nachzuahmen.**
- **Das Herzstück der Informatik bilden Algorithmen.**
- **Ein Algorithmus und der Beweis seiner Korrektheit sollten gemeinsam hergeleitet werden.**
- **Die Umsetzung eines Algorithmus in eine Programmiersprache sollte einfach sein.**

Anhang:

Präfixe für Einheiten im Dezimalsystem:

Faktor	Name	Symbol
10^{24}	yotta	Y
10^{21}	zetta	Z
10^{18}	exa	E
10^{15}	peta	P
10^{12}	tera	T
10^9	giga	G
10^6	mega	M
10^3	kilo	k
10^2	hecto	h
10^1	deka	da
10^{-1}	deci	d
10^{-2}	centi	c
10^{-3}	milli	m
10^{-6}	micro	μ
10^{-9}	nano	n
10^{-12}	pico	p
10^{-15}	femto	f
10^{-18}	atto	a
10^{-21}	zepto	z
10^{-24}	yocto	y

Präfixe im Dualsystem:

Standard: IEC 60027-2, Second edition, 2000-11, Letter symbols to be used in electrical technology – Part 2: Telecommunications and electronics.

Faktor	Kurzname	Symbol	Langname
2^{10}	kibi	Ki	kilobinary
2^{20}	mebi	Mi	megabinary
2^{30}	gibi	Gi	gigabinary
2^{40}	tebi	Ti	terabinary
2^{50}	pebi	Pi	petabinary
2^{60}	exbi	Ei	exabinary

Beispiele:

1 kibibit	=	1.024 bit
1 kilobit	=	1.000 bit
1 mebibit	=	1.048.576 bit
1 megabit	=	1.000.000 bit
1 gibibit	=	1.073.741.824 bit
1 gigabit	=	1.000.000.000 bit