Komplexitätstheorie

Einführung und Überblick

Helmut Veith
Technische Universität München

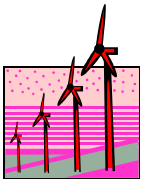
Softwarequalität

Zentrales Thema der Informatik

40 Jahre Software Engineering
40 Jahre Software Krise

Qualitätskriterien

Was ?	**Korrektheit**
Wie ?	**Skalierbarkeit**
Wer ?	**Sicherheit**



Welche Probleme können algorithmisch gelöst werden ?

Unentscheidbarkeit

Es gibt keine Software, die fehlerfrei
… toten C-Code beim Compilieren entfernt.
… Polynomialgleichungen in Z lösen kann.
… die Gleichheit zweier SQL-Abfragen feststellt.

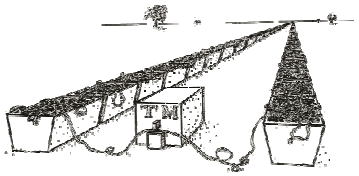
Und zwar aus prinzipiellen logischen Gründen.

Alan Turing

Cambridge, 30er Jahre

Turings Maschinenmodell

Radikal einfaches Maschinenmodell.



Erlaubt generelle logische Schlüsse über Software.

➲ Nächster Vorlesungstermin.

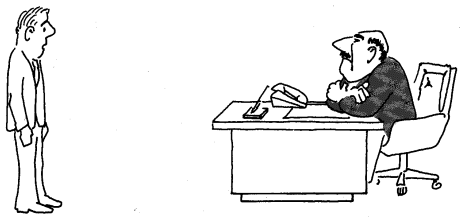
Unentscheidbar

Satz von Rice
Die nicht-trivialen Eigenschaften von Software sind nicht entscheidbar.

Entscheidbar

Nicht Effizient

Effizient



"I can't find an efficient algorithm, I guess I'm just too dumb."

Welche Probleme können algorithmisch gelöst werden ?

Welche Probleme können **effizient** algorithmisch gelöst werden ?

Unentscheidbar

Satz von Rice
Die nicht-trivialen Eigenschaften von Software sind nicht entscheidbar.

Entscheidbar

Nicht Effizient

Effizient

Garey & Johnson:
Computers and Intractability

Time complexity function	Size n: 10	20	30	40	50	60
n	.00001 second	.00002 second	.00003 second	.00004 second	.00005 second	.00006 second
n^2	.0001 second	.0004 second	.0009 second	.0016 second	.0025 second	.0036 second
n^3	.001 second	.008 second	.027 second	.064 second	.125 second	.216 second
n^5	.1 second	3.2 seconds	24.3 seconds	1.7 minutes	5.2 minutes	13.0 minutes
2^n	.001 second	1.0 second	17.9 minutes	12.7 days	35.7 years	366 centuries
3^n	.059 second	58 minutes	6.5 years	3855 centuries	2.3×10^5 centuries	1.3×10^{13} centuries

Polynomieller Rechenaufwand

Exponentieller Rechenaufwand

Figure 1.2 Comparison of several polynomial and exponential time complexity functions.

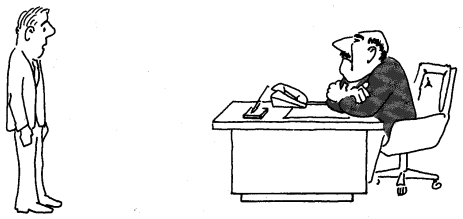
Garey & Johnson:
Computers and Intractability

Size of Largest Problem Instance Solvable in 1 Hour

Time complexity function	With present computer	With computer 100 times faster	With computer 1000 times faster
n	N_1	100 N_1	1000 N_1
n^2	N_2	10 N_2	31.6 N_2
n^3	N_3	4.64 N_3	10 N_3
n^5	N_4	2.5 N_4	3.98 N_4
2^n	N_5	$N_5+6.64$	$N_5+9.97$
3^n	N_6	$N_6+4.19$	$N_6+6.29$

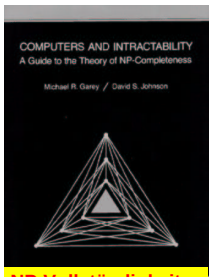
Figure 1.3 Effect of improved technology on several polynomial and exponential time algorithms.

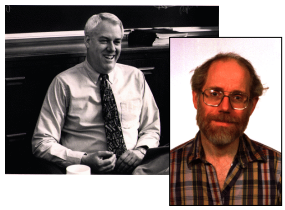
Garey & Johnson:
Computers and Intractability



"I can't find an efficient algorithm, I guess I'm just too dumb."

Garey & Johnson:
Computers and Intractability





NP-Vollständigkeit

researchindex.com

Most cited articles in Computer Science - May 2003 (CiteSeer)

Generated from documents in the CiteSeer database. This list does not include citations where one or more authors of the citing and cited articles match. This list is automatically generated and may contain errors. The list is generated in batch mode and citation counts may differ from those currently in the CiteSeer database which is continuously updated.

All Years 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003

Next 200

1. Doc Context *3327* [GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

2. Book Context *2810* [12] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. The MIT Press, 1991.

3. Doc Context *2269* [25] C.A.R. Hoare, *Communicating Sequential Processes*, Prentice-Hall International, 1985.

4. Book Context *1736* [Gol89] David E. Goldberg. *Genetic Algorithms in Search*, Optimization, and Machine Learning. Addison-Wesley, Reading, Massachusetts, 1989.

"To become a good algorithm designer, you must understand the rudiments of NP-completeness.

Perebor: Brute Force Search

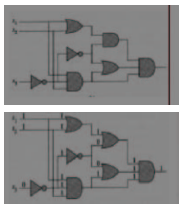
Yablonski
MGU Moskau

"On the impossibility of eliminating Perebor in solving some problems of circuit theory." (1959)

The $ 1.000.000 Problem

Existiert ein polynomieller Algorithmus für folgende Frage:

Kann ein Gatter eines kombinatorischen Schaltkreises je 1 werden ?



CIRCUIT SATISFIABILITY

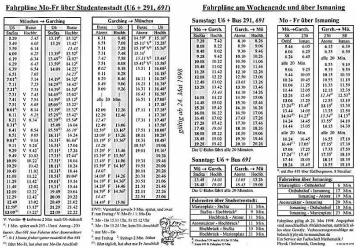
UNO Problem

Kann eine Gruppe n Delegierter so um einen runden Tisch plaziert werden, dass keine Feinde nebeneinander zu sitzen kommen ?

Graphentheoretisch: Hamiltonian Cycle ?

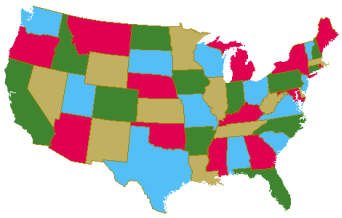
Scheduling Problem

Kann eine Menge von n abhängigen Aufgaben innerhalb von t Zeiteinheiten eingeplant werden ?



Färbbarkeitsproblem

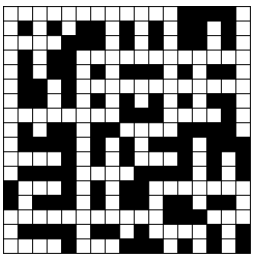
Kann eine Landkarte mit 3 Farben koloriert werden, sodass Nachbarstaaten unterscheidbar sind ?



4-Farbensatz

Kreuzwortproblem

Kann ein Kreuzworträtsel mit Wörtern einer Liste vollständig gefüllt werden ?

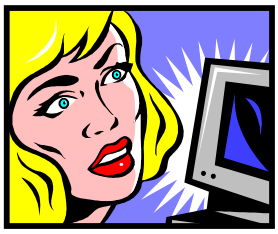


Rucksackproblem

Gegeben eine Menge von Wertobjekten unterschiedlichen Gewichts und Wertes. Kann ein Dieb mit einem Rucksack mit Maximalgewicht M den Wert W erzielen ?

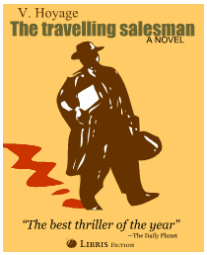
Programmkompression

Kann ein C-Programm durch Verwendung von #define auf Länge N reduziert werden ?



Travelling Salesperson

Kann ein Vertreter n Städte bereisen, und dabei nicht mehr als K Kilometer zurücklegen ?

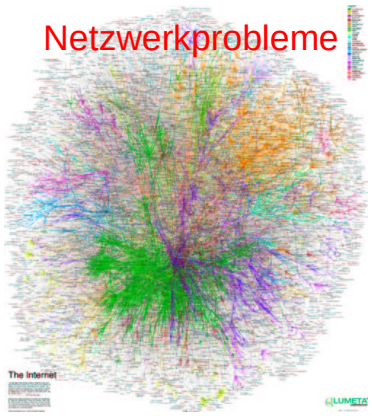


Netzwerkprobleme

Gibt es …

- k paarweise verbundene Knoten ?
- eine Aufteilung in zwei Teile, sodass die Leitungskapazitaet dazwischen >= k ist ?
- eine Erweiterung um k Verbindungen, sodass das Netzwerk redundant wird ?

Netzwerkprobleme

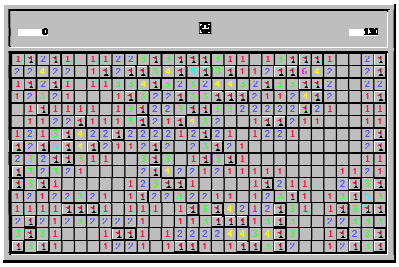


Puzzle Probleme

Kann mit einer Menge von Puzzlesteinen ein Feld der Groesse N x N gefuellt werden ?

Minesweeper

Ist eine Spielsituation im Spiel Minesweeper konsistent ?



Cluster Workload Assignment

Kann eine Menge von Prozessen bekannter Workload auf N Prozessoren eingeplant werden?

Integer Programming

Gegeben ein Optimierungsproblem, dass durch Ungleichungen ausgedrueckt wird. Gibt es eine ganzzahlige Loesung ?

Suchalgorithmen ?

Für die genannten Probleme sind nur exponentielle Algorithmen bekannt.

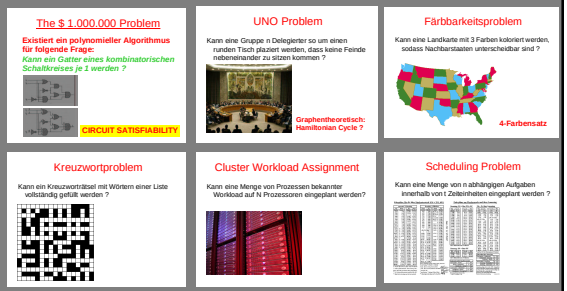
Aber: Die Eignung einer vorgeschlagenen Lösung kann effizient überprüft werden.

NP = Entscheidungsprobleme, die

(1) Lösungen polynomieller Grösse haben

(2) die effiziente Überprüfung von Lösungskandidaten erlauben.

NP Probleme



… und tausende weitere Probleme.

NP Probleme exponentiell lösbar

Angenommen, ein NP-Problem mit Eingabegrösse N hat Lösungen der Grösse N^k in Binärdarstellung.

Dann kann ein Suchalgorithmus in Zeit

$$2^{(N^k)}$$

alle Lösungen überprüfen.

Theorem NP ⊆ EXPTIME

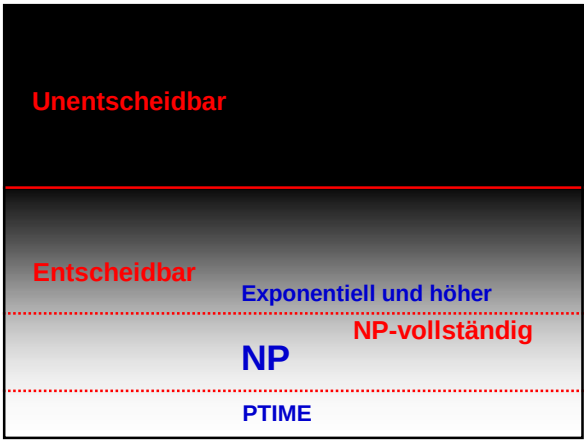
Cook-Levin 1971

Theorem

Wenn es einen polynomiellen Algorithmus für Circuit SAT gibt, dann auch für alle anderen Probleme in NP.

➲ CIRCUIT SAT ist ein "*maximal schwieriges*" Problem in NP.

➲ Probleme mit dieser Eigenschaft heissen **NP-vollständig.**



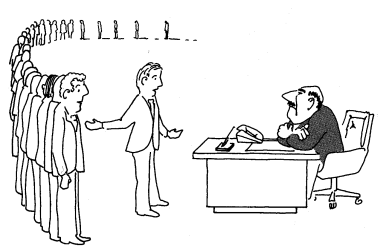
NP-Vollständigkeit

NP-Probleme, für die wir keinen polynomiellen Algorithmus kennen, sind *meist* NP-vollständig.

Viele Tausende NP-vollständige Probleme bekannt – auch alle heute genannten.

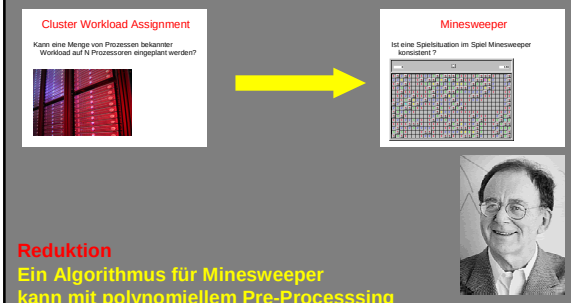
Ein polynomieller Algorithmus für nur *eines* davon würde polynomielle Algorithmen für *alle* mit sich ziehen.

NP-Vollständigkeit



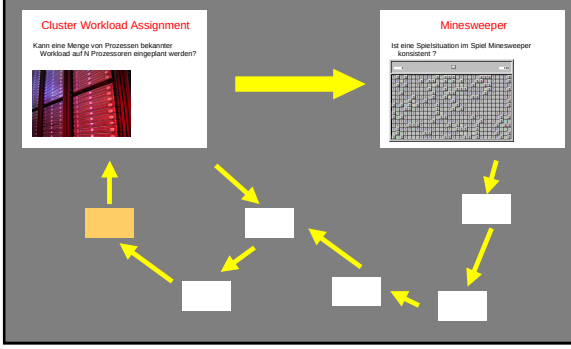
"I can't find an efficient algorithm, but neither can all these famous people."

Reduktionen



Reduktion
Ein Algorithmus für Minesweeper kann mit polynomiellem Pre-Processsing zum Lösen von Workload Assignment verwendet werden.

Reduktionen



The $ 1.000.000 Problem

Gibt es polynomielle Algorithmen für die Probleme in NP ?

NP = PTIME ?

NP = PTIME

+ Effiziente Algorithmen für schwierige Probleme.
- Annahmen der Computer-Kryptographie würden zusammenbrechen.

und damit das Bankensystem.

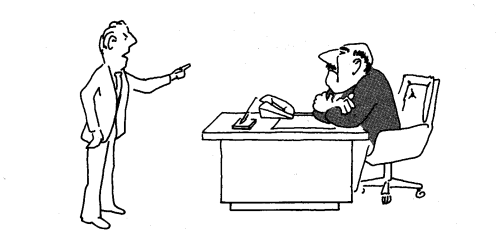
NP = PTIME

K. Gödel

Beweisfindung in der Mathematik weitgehend automatisierbar.

J. van Neumann

NP ≠ PTIME



"I can't find an efficient algorithm, because no such algorithm is possible!"

NP-Vollständigkeit

Natürliche Problemklasse.
"Intellektueller Export" der Informatik.

Biologie: Protein-Faltung
Chemie: Chemische Synthese
Stadtplanung: Verkehrsfluss
Elektronik: VLSI Layout
Medizin: Rekonstruktion von 3D Bildern.

Physik, Wirtschaft, Statistik, Finanzwissenschaft, Politikwissenschaft …

researchindex.com

Most cited articles in Computer Science - May 2003 (CiteSeer)

Generated from documents in the CiteSeer database. This list does not include citations where one or more authors of the citing and cited articles match. This list is automatically generated and may contain errors. The list is generated in batch mode and citation counts may differ from those currently in the CiteSeer database, because the database is continuously updated.

All Years 1990 1991 1992 1993 1994 1995 1996 1997 1998 1999 2000 2001 2002 2003

Next 200

1. Doc Context *3327* [GJ79] M.R. Garey and D.S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. Freeman, New York, 1979.

2. Book Context *2810* [12] Thomas H. Cormen, Charles E. Leiserson, and Ronald L. Rivest. *Introduction to algorithms*. The MIT Press, 1991.

3. Doc Context *2269* [25] C.A.R. Hoare, *Communicating Sequential Processes*, Prentice-Hall International, 1985.

4. Book Context *1736* [Gol89] David E. Goldberg. *Genetic Algorithms in Search*, Optimization, and Machine Learning. Addison-Wesley, Reading, Massachusetts, 1989.

Turing Award

"Nobelpreis der Informatik"

Alan M. Turing
The First Society in Computing

1982 Stephen Cook
1985 Richard M. Karp
1993 Juris Hartmanis & Richard E. Stearns
1995 Manuel Blum
2000 Andrew Yao

Themen der Vorlesung

- NP-Vollständigkeit
- Komplexitätsklassen und deren Beziehungen
- Komplexität als Mass der Ausdruckskraft

- Probleme aus Datenbanken, KI, VLSI, Kryptographie, Compiler, Scheduling etc.
- **Umgang mit NP-vollständigen Problemen** Approximation, Randomisierung, Einschränkungen und Heuristiken

Danke für Ihre Aufmerksamkeit

Zum Komplexitätsbegriff

Strukturelle Komplexität

Average Case Analyse
Effiziente Algorithmen

Logische Komplexität
Beschreibungssprachen: SQL

Kolmogorov Komplexität
Komplexitaet = Programmgroesse

Axiomatische Komplexität
Verallgemeinerte Komplexitaetsmasse

Literatur

UC Berkeley