Forschungsseminar: P = NP?

Übersicht

Das P-NP-Problem (auch: P versus NP) ist eines der zur Zeit wohl wichtigsten ungelösten Probleme der Mathematik und der theoretischen Informatik. Es stellt die Frage, ob überhaupt Probleme existieren, für die eine gegebene Lösung zwar effizient überprüft werden kann, das Finden einer solchen Lösung jedoch nicht effizient möglich ist. Die meisten Informatiker werden sofort sagen: „Na klar, z.B. das Handlungsreisenden- bzw. Hamiltonkreisproblem, oder Minesweeper, oder SAT ...!" In der Tat sind zu diesen und vielen anderen Problemen keine effizienten Lösungsalgorithmen bekannt, auch nach jahrelangem Forschen nicht, daher vermuten die meisten Wissenschaftler, dass des auch keine gibt. Aber niemand hat bisher mathematisch *beweisen* können, dass diese Probleme wirklich schwierig, d.h. nicht effizient lösbar sind (und wir bisher eben einfach noch nicht den richtigen Weg gefunden haben...).

Exakt formulieren kann man das P-NP-Problem im Rahmen der Komplexitätsteorie. In diesem Teilgebiet der Theoretischen Informatik werden algorithmisch behandelbare („berechenbare") Entscheidungsprobleme je nach Komplexität ihrer effizientest möglichen Lösungsalgorithmen in Klassen eingeteilt. Zum Beispiel ist eines der Entscheidungsprobleme SEARCH-SORT(x), welches überprüfen soll, ob das Element x in einer sortierten Datenstruktur mit n Elementen existiert; der effizienteste Algorithmus, der das Problem löst, ist die Binäre Suche mit einer Zeitkomplexität $T(n) = \Theta(\log n) \subset O(\log n)$ und einer Speicherplatzkomplexität $S(n) = \Theta(n) \subset O(n)$ (was im Übrigen auch die theoretisch benötigten Mindestkomplexitäten sind, siehe z.B. mein Skript *Grundlagen der Programmierung* aus dem 1. Semester).

Zwei auch für die „Alltagswelt" wichtige Komplexitätsklassen sind nun P und NP: P für „polynomial" ist, etwas salopp gesprochen, die Klasse der effizient lösbaren Probleme, also derjenigen Probleme, für die ein Lösungsalgorithmus der Komplexität

$$T(n) = O(n^k) \quad \text{für ein } k > 0$$

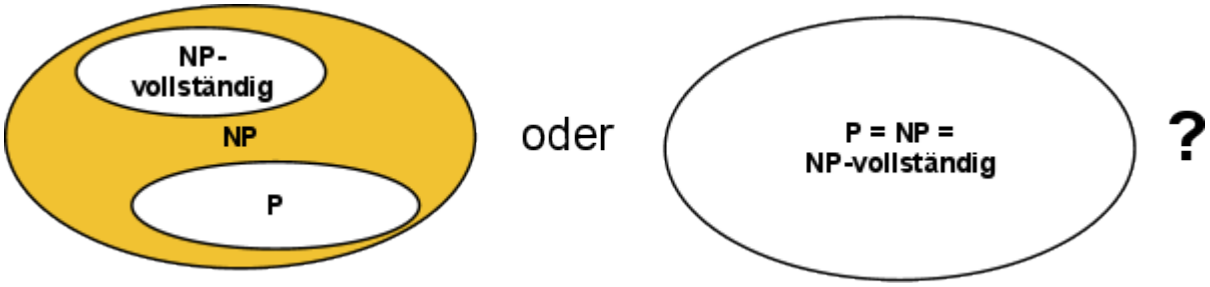
existiert, und NP für „nichtdeterministisch polynomial" bezeichnet die Klasse derjenigen Probleme, für die ein Lösungsalgorithmus der Zeitkomplexität

$$T(n) = O(2^{n\hat{}k}) \quad \text{für ein } k > 0$$

existiert, für die das Überprüfen einer möglichen Lösung („Probe") aber effizient, also in der Laufzeit $T(n) = O(n^k)$, möglich ist.

Obwohl P und NP bereits in den 1970er Jahren unabhängig von Stephen Cook und Leonid Levin definiert wurden, ist bis heute offen, ob eigentlich überhaupt $P \neq NP$ ist. Was man bisher nur weiß, ist das, was sofort klar ist: $P \subseteq NP$.

Das P-NP-Problem wurde vom Clay Mathematics Institute in die Liste der Millenniumsprobleme aufgenommen, für deren Lösung es ein Preisgeld von US$ 1 Mio gibt.



Themenkomplexe:

1. Turing-Maschinen (deterministisch und nichtdeterministisch) und Algorithmen
2. Entscheidungsprobleme und Entscheidbarkeit
3. Einige Komplexitätsklassen und Beispiele ihrer jeweils typischen Entscheidungsprobleme, insbesondere P, NP, und NP-schwer, ggf. auch L, PSPACE, EXPTIME
4. Reduzierbarkeit von Problemen
5. P = NP? Ein Milleniumsproblem und mögliche Lösungsansätze
6. Was wäre eigentlich, wenn tatsächlich P = NP gälte?
7. Ausblick: Quantenalgorithmen und Quantenrechner

Ziele des Seminars

Erarbeitung eines aktuellen Forschungsthemas, Verständnis der Komplexitätstheorie als Teil der Algorithmik, und natürlich: Das Lösen des alten Problems :)

Teilnahmevoraussetzungen

Kenntnisse der Veranstaltung Algorithmen und Optimierung, insbesondere der Begriffe O-Notation und Laufzeitkomplexität

Formale Anerkennbarkeit

Das Seminar kann als Veranstaltung "Informationstechnik: Konzepte und Verfahren" im Masterstudiengang oder als Seminar Wirtschaftsinformatik anerkannt werden.

Literatur

- Sanjeev Arora, Boaz Barak: *Computational Complexity: A Modern Approach.* Cambridge University Press, Cambridge 2009
- Dirk W. Hoffmann: Theoretische Informatik. Hanser, München 2009
- C. M. Papadimitriou: *Computational Complexity.* Addison-Wesley, Reading (Massachusetts) 1994
- Michael Sipser: *Introduction to the Theory of Computing. 2. Auflage.* Cengage Learning Services, Boston 2006