NP – Vollständigkeit

Patryk Mazur

04.05.2010

0.Gliderung

1. Einführung
    1. Definitionen P , NP, coNP, EXP, NEXP
    2. Bekannte Zusammenhänge zwischen dem Klassen
    3. Hypothesen zu deren zusammenhängen und deren Konsequenzen
2. Polynominal Reduktion und NP-Vollständigkeit
    1. Definition von NP-Vollständigkeit und (PReduktion) Polynominal Reduktion
    2. NP Probleme
    3. Bsp. Für eine Reduktion
3. Geschichte
    1. Stephen A. Cook , Richard Karp
    2. SAT als „ERSTES“ Problem
    3. Beweis von SAT-Grundidee
    4. Reale Probleme und Lösungsansetze
    5. Zusammenfassung

1. Definitionen

Eine **Komplexitätsklasse** ist eine Kategorie von Problemen/Algorithmen, zusammengefasst nach einem gemeinsamen Maß der Komplexität(gebrauchte Zeit oder Speicherplatz zum Bsp.)

Komplexitätsklasse P(auch: **PTIME**) ist diejenige Komplexitätsklasse, welche die Entscheidungsprobleme enthält, die in Polynomialzeit für deterministische Turingmaschinen lösbar sind. Diese Problemklasse wird allgemein als die Klasse der "praktisch lösbaren" Probleme betrachtet.

(Komplexitätsklasse NP). ***NP*** *ist die Klasse aller Entscheidungsprobleme L, die in polynomieller Zeit verifizierbar sind. D.h. es gibt einen Algorithmus A polynomieller Laufzeit und es gibt ein k ∈N, so dass*
L = {w| ∃ *Wort x mit |x| ≤ |w|^k, so dass A(x,w) = 1}*
x *heißt auch Zeuge für w.*

|x| ≤ |w|^k-Sicherung dass die Eingabe Polynomiell in der Länge ist

(Komplexitätsklasse co- NP). **Co-NP** ist eine Komplexitätsklasse, in der genau die Sprachen enthalten sind , deren Komplemente zu NP gehören. Intuitiv gesprochen besteht Co-NP aus der Klasse der Sprachen, für die ein Beweis, dass ein Wort nicht zur Sprache gehört, nichtdeterministisch in polynomieller Zeit überprüft werden kann.

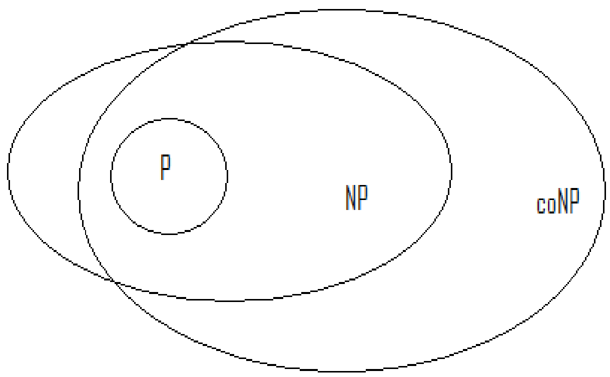
Es gibt ein Algorithmus A mit polynomieller Laufzeit und ein k ∈ N, so dass L = {w|∀x mit |x| ≤ |w|k gilt A(x,w) = 0}.

Komplexitätsklasse **EXPTIME (EXP)** ist die Komplexitätsklasse der Entscheidungsprobleme, die von einer deterministischen TM in durch $\mathcal{O}\left(2^{p(n)}\right)$ beschränkter Zeit entschieden werden können . $p(n)$ ist dabei ein beliebiges Polynom von der Eingabelänge *n*.

Komplexitätsklasse **NEXPTIME** (**NEXP**) ist die Komplexitätsklasse der Entscheidungsprobleme, die von einer nichtdeterministischen TM in durch $O(2^{p(n)})$ beschränkter Zeit akzeptiert werden können.

Wir wissen:

1.2 P ist Teilmenge von NP und non co-NP deswegen sind sie keine disjunkten Mengen



Die 1 Mio $ Frage P=NP ?

Vermutung nein , kein Beweis dafür.

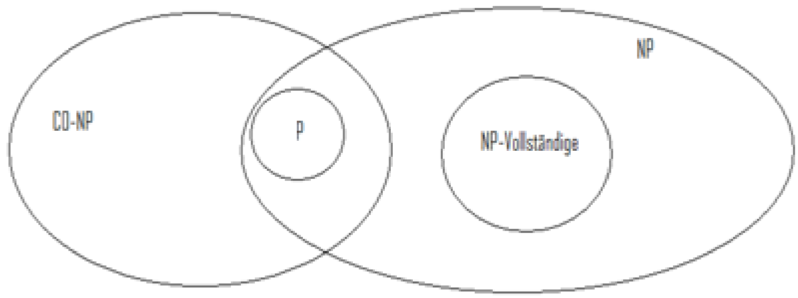
1.3 Hypothesen und deren Konsequenzen

P=NP => NP = co-NP

NP!=co-NP => P!=NP

EXP!= NEXT => P!=NP

Ist ein Problem im NP und co-NP wird vermutet dass er nicht NP-vollständig ist



2 .Polynomzeit Reduktion

L1, L2 ∈ E ^* *seien Sprachen.*

L1 heißt in polynomieller Zeit auf L2 reduzierbar (L1 ≤p L2)

genau dann wenn eine in polynomieller Zeit berechenbare Funktion

F : E^* → E^* existiert, mit w ∈ L1 genau dann wenn f(w) ∈ L2.

Polynomzeit Reduktion ist Transitiv

Sein g zwei Funktionen die im n^d und n^c wachsen dann wächst g(f(w)) in n^(c*d)

L2 ∈ NP und L1 ≤P L2, dann ist auch L1 ∈ NP.

Bsp. PReduktion

PARTITION<pSUBSET-SUM

Existiert ein Algorithmus in polynomieller Laufzeit für SUBSETSUM, so existiert auch einer für PARTITION.

Denn: Sei(a_1, ..., a_n) eine Eingabe für PARTITION.
(a_1, ..., a_n) hat eine positive Antwort ⇔(a_1, ..., a_n,b) hat eine positive Antwort bei SUBSET-SUM,
wobei b=1/2 Summ(a1,……….,an)

Jede Eingabe w für PARTITION ist leicht zu übersetzen in eine Eingabe w_0
für SUBSET-SUM, so dass w eine positive Antwort liefert gdw. w_0 eine positive liefert
.

Bsp. PReduktion

Partition <p Bin Packing

Nehme 2 Behälter mit der Größe ein halb von der Summe der Größe aller Objekte , einfach wenn man weiß wie.

Eingabe Partition

(a1,…………..,an)

Eingabe Bin Packing

Objekte (a1,……………………….,an)

K=2

B=1/2 Summ(a1,……….,an)

NP-Vollständigkeit

Def:

1. Falls L1 NP-schwer ist und L1< p L2, dann ist auch L2 NP-schwer.

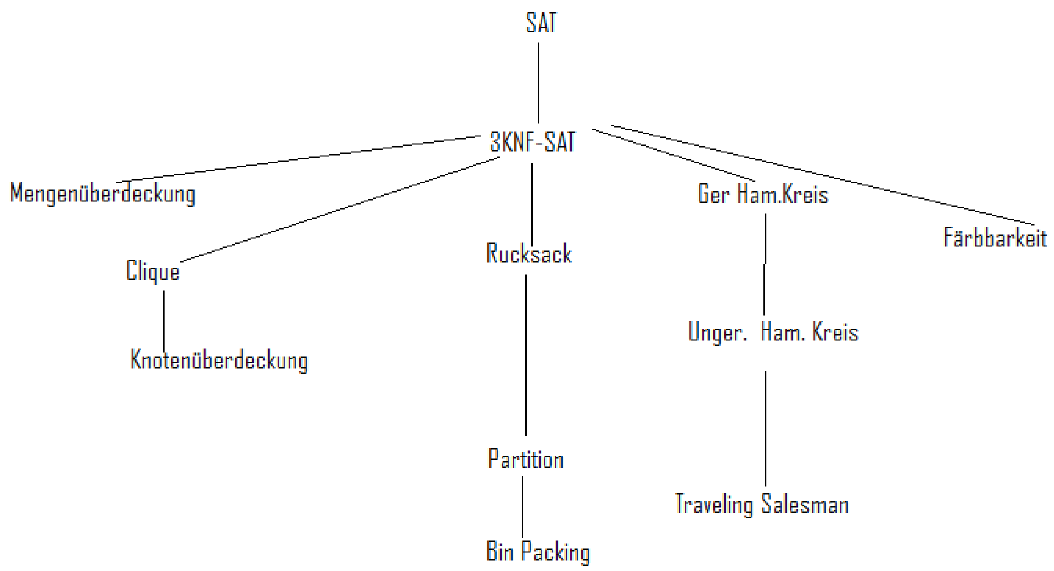
*2.*Ein Problem L heißt NP-vollständig, genau dann wenn L NP-schwer ist und L ∈ NP.

3.Problem (Sprache) L heißt NP-schwer (NP-hard) genau dann, wenn L'<p L für alle L' die zu NP gehören.

Um zu zeigen dass, ein Problem NP-Vollständig ist brauchen wir Definitionen 1 und 2 oder 2 und 3.Wenn wir aber 1 und 2 benutzen dann brauchen wir ein schon bekanntes NP schweres Problem

1971 *Cook-Lewin Theorem SAT ist NP-Vollständig(TM)*

Danach hat Karp 21 weitere Probleme vorgestellt. Alle Reduktionen lassen sich bis zum SAT zurückführen.



AB 1. Reduktionsnetz

2.2 NP Vollständige Probleme

SAT
SAT dreht sich um die Erfüllbarkeit von aussagelogischer Formel. Wir haben also Variablen und Verknüpfungen $\wedge$, $\vee$, $\neg$ und wir nehmen an, dass die Formel in konjunktiver Normalform vorliegt. Gegeben sei nun eine Formel α in konjunktiver Normalform und SAT fragt nun, ob α erfüllbar ist, d.h. gibt es eine Belegung der Variablen mit 0 oder 1, so dass sich α zu 1 auswertet?

3-SAT
Gegeben sei eine boolesche Formel α mit je drei Literalen pro Klausel. Die Frage ist nun, ob α erfüllbar ist?

CLIQUE
Gegeben ist ein ungerichteter Graph $G = (V,E)$ und eine Zahl $k \in N$. Die Frage ist nun, ob eine k-Clique in G existiert, d.h. eine Teilmenge $V' \subset V$ der Größe k mit $(u, v) \in E$ für alle $u, v \in V'$?

Unabhängige Knotenmenge (UK)
Gegeben sie ein Graph $G = (V,E)$ und ein $k \in N$. Die Frage ist nun, ob G eine unabhängige Knotenmenge V' der Größe k enthält, d.h. $\forall u, v \in V' \, (u, v) \notin E$?

Knotenüberdeckung
Gegeben sei wieder ein Graph $G = (V,E)$ und ein $k \in N$.
Frage: besitzt G eine „überdeckende Knotenmenge“ ?

SUBSET-SUM (Rucksak)
Gegeben seien natürliche Zahlen $a_1, \ldots, a_n$ und ein $b \in N$. Die Frage ist nun, $\exists I \subset \{1, \ldots, n\}$ mit $P_{i \in I}\, a_i = b$.
Partition
Geg: Natürliche Zahlen a1,…………………,ak   k gehört zu N
Frage :kann ich die Zahlen in zwei mengen Teilen so daß, deren summe
Gleich ist ?

Gerichteter Hammiltonscher Kreis (GHK)

Gegeben ist ein gerichteter Graph G = (V,E) und die Frage ist, ob ein gerichteter Hamiltonscher Kreis existiert, d.h. ein Kreis, der jeden Knoten genau einmal enthält.?

Ungerichteter Hammiltonscher Kreis (UGHK)

Gegeben ist ein ungerichteter Graph G = (V,E) und die Frage ist, ob ein Hamiltonscher Kreis existiert, d.h. ein Kreis, der jeden Knoten genau einmal enthält?

TSP:
Gibt es zu einem gegebenen ungerichteten, gewichteten, vollständigen Graphen
G = (V;E) und einer Zahl k eine Rundereise in dem Graphen, die jeden Knoten genau einmal besucht undhöchsten Kosten k hat?

Färbbarkeit
Gegeben G=(V;E) und k gehört zu N
Frage kann man die Knoten mit k Farben so färben, dass zwei Knoten mit gleicher Farbe nicht benachbart sind?

Bin Packing

Gegeben: k Behälter der Größe b, n Objekte mit der Größe/Gewicht <= b
Frage kann man diese Objekte in die k Behälter einsetzten so dass sie nicht überlaufen?

Partition ist NP-vollständig

1. Partition gehört zu NP(einfache Teil)
Eingabe für Partition (a1,a2,a3,a4,…………..,an)
Teile in 2 Mengen
(a1,a2,……….,a(n/2)) und (a(n/2)+1,…,an)
Man kann in Polynomiallzeit (sogar in linearer) nachprüfen ob der Zeuge das erfühlt.

2.Eingabe SubsetSum => Eingabe Partition
S=(a1,………………….,an,b)=>(a1,………………….,an,x,y)
Wobei
S‘=Summ(Teilmenge von S)=b
A=Summ von allen ai

X=2*A-b und y=A+b
Geht in Polynomieller Zeit ???? Ja
X und Y können nicht in derselben Partition Liegen denn
X+Y=2*A-b+A+b=3A und die Summe für Ganze Partition ist 4A
Also hat jede Partition den Wert 2A => R1 = Vereinigung X mit S‘
R2= Vereinigung Y mit(S/S‘)

„=>“
Eingabe für SubsetSum Liefert Lösung => Eingabe für Partition Liefert Lösung
S‘ ist Die Lösung für SubsetSum = b
=>
X+S‘=2A-b+b=2A
Y+Summ(S/S‘)=A+b+(A-b)=2A
„<=“
Eingabe für Partition Liefert Lösung => Eingabe für SubsetSum Liefert Lösung
R1 = Vereinigung X mit S‘, X = 2A-b
R2= Vereinigung Y mit(S/S‘), Y=A+b
=> S‘ ist die Lösung von Subset Sum

SAT ist NP-vollständig durch *Cook-Lewin Theorem*

Grundidee
1. Es muss gezeigt werden wie man aus einer beliebigen Sprache L in NP in Polynomialzeit eine Reduktion auf SAT machen kann.

Aus einer Zeichenkette x eine aussagenlogische Formel in Polynomialzeit. Die Formel ist erfühlbar wenn x zu L gehört.

2.Weil *L* in NP liegt, gibt es eine nichtdeterministische Turingmaschine *M*, welche in Polynomialzeit entscheidet, ob *x* zur Sprache *L* gehört.

3.Grundidee der Reduktion ist nun, die Aussage „Die Berechnung der Maschine M bei Eingabe *x* ergibt, dass *x* zur Sprache *L* gehört“ in einer aussagenlogischen Formel auszudrücken

4. In dieser Formel müssen sich also eine Beschreibung der Maschine M, eine Beschreibung der Eingabe *x* sowie die „Rechenregeln“, nach denen eine nichtdeterministische Turingmaschine arbeitet, sein.

4. Das alles muss in Polynomialzeit staatfinden.

5 Reale Probleme und Lösungsansetze

1.Man schreibt tatsächlich einen optimalen Algorithmus , aber man weiß dass der Computer ist stark genug , um die Datenmenge in einer Zeitspanne die für uns ok ist zu schaffen

2.Benutze einen heuristischen Algorithmus und sucht nicht nach einer optimalen Lösung sondern nur nach einer Lösung die für die konkrete Aufgabe *gut genug* ist.

3.Man wählt einen approximativen Algorithmus der nicht die perfekte Lösung aber eine nahezu perfekte Lösung bietet.

3.1 Hilft nicht immer

Für kein α > 1 existiert ein α-approximativer Algorithmus polynomieller Laufzeit für TSP, es sei denn P = NP.

5.Zusammenfassung

Was Möchte ich was, Hoffe ich, habt Ihr aus diesen Referat Mitgenommen

1. Die Klasse der NP-Vollständigen Problemen ist Jung und voller Geheimnisse
2. Sollte jemand es schaffen die 1 Mio $ Aufgabe zu lösen würde sich unsere Welt verändern weil , es keine Abstrakten Probleme sind
3. Das wichtigste Werkzeug ist hier die PReduktion, deswegen ist der Satz von Cook und Lewin so wichtig dass SAT NP-vollständig ist (Oder hat jemand Lust zu zeigen dass jede Sprache die zu NP gehört sich mit Hilfe von TM auf dieses Problem reduzieren lässt)
4. Die TM wahr sehr oft in Hintergrund, das ist auch so mit Np-Vollständigen Problemen, manche sehen ob sie nichts mit TM zu tun hätten, aber sie ist immer in Hintergrund.

Gute quellen

http://algo.rwth-aachen.de/Lehre/WS0910/VBuK/Folien/komplexitaet3.pdf
www.fbi.h-da.de/fileadmin/personal/s.lange/.../kt_teil04_4.pdf
www.inf.fu-berlin.de/lehre/WS08/HA/download/skript.pdf
http://ls11-www.cs.uni-dortmund.de/people/gutweng/DAP2-08/OptApproxWeb.pdf