Theoretische Grundlagen der Informatik II
Blatt 3

Markus Vieth, David Klopp, Christian Stricker

16. November 2015

Aufgabe 1

a)

Behauptung: A ist NP-vollständig und $B \in NP$. Wenn $A \leq_{poly}$ B, dann ist B auch NP-vollständig.
Beweis: A ist NP-vollständig => A ist NP-hart => Da man A auf B in polynomieller Zeit reduzieren kann, ist B auch NP-hart. Da B in NP liegt, folgt daraus: B ist NP-vollständig.

b)

~~Behauptung:~~ Annahme: Sei A ein NP-hartes Problem. Wenn $A \leq_{poly} B$, dann ist B in der Komplexitätsklasse P.
Widerspruchsbeweis: Angenommen wahr: A ist NP-hart => $X \leq_{poly} A \ \forall \ X \in NP => X \leq_{poly} A \leq_{poly} B \ \forall \ X \in NP => X \leq_{poly} B \ \forall \ X \in NP, da B \in P => NP = P \ Widerspruch.$

c)

Behauptung: Sei A ein Problem. Sei B ein NP-hartes Problem. Wenn A in der Komplexitätsklasse P ist, dann gibt es eine Reduktion $A \leq_{poly} B$.
Beweis: $A \in P \subset NP => A \leq_{poly} B$

d)

Behauptung: Für ein beliebiges Problem X gelte $SAT \leq_{poly} X \leq_{poly} SAT$. Dann ist X NP-vollständig.
Beweis: SAT ist ein NP-vollständiges Problem. Da SAT auf X in polynomieller Zeit reduziert werden kann, ist ~~SAT~~ X NP-hart.
$SAT \in NP$ und $X \leq_{poly} SAT$ => $X \in NP$. DA $X \in NP$ und X NP-hart => X ist NP-vollständig

e)

Definition Klasse NP: Menge aller Entscheidungsprobleme, die effizient überprüft werden können.
Es existiert ein polynomieller Algorithmus B mit zwei Eingaben s und t und ein Polynom p, so dass gilt:
Ist $s \notin X$, so ist B(s,t) = no für alle $t \in \{0,1\}^*$
Ist $s \in X$, so existiert rin $t \in \{0,1\}^*$ mit $|t| \leq p(s|)$ und B(s,t) = yes
Verifizierer?Zertifikat?

f)

Satz von Cook und Levin: SAT ist NP-vollständig und wenn ein Problem X auf SAT reduziert werden kann, kann man auch das Problem X auf SAT reduzieren.

g)

Unter der Annahme, das die Reduktion den Algorithmus von A nicht beinhaltet:
Da der Algorithmus B durch A gelöst wird, beträgt die Laufzeit die Summe von A und der Reduktion: $O(n^5 + n^4) = O(n^5)$ f

Aufgabe 2

Behauptung: Set-Cover(SC) ist NP-Vollständig

Zu zeigen: Set-Cover(SC) $\in$ NP

Beweis:

Guess

Es werden nichtdeterministisch k paarweise verschiedene Teilemengen S ausgewählt und in T gespeichert.

Check

kopiere M in M'. (Laufzeit |M|=n)
für alle Teilmengen S_i in T (Laufzeit höchstens n)
  für alle Elemente $e_j \in S_i$ (Laufzeit höchstens n)
    für alle Elemente $e_k \in M'$ (Laufzeit höchstens n)
      wenn $e_j = e_k$ (Laufzeit konstant)
        lösche e_k in M' (Laufzeit konstant)
Wenn $|M'| = 0$
  return ja
sonst
  return nein.
Viel zu kompliziert!

$$\Rightarrow T(n) \in O(n^3)$$

$\Rightarrow SC \in NP$

Beweis:

Zu zeigen: Set-Cover(SC) ist NP-hart

Für jeden Knoten im Graphen existiert genau eine Teilemenge. (Laufzeit n)

$$\forall v_i \in V \exists! S_i \text{ Leerzeichen?!}$$

Für jede Kante im Graphen existiert genau ein Element in M. (Laufzeit bis zu n^2)

$$\forall k_i \in E \exists! e_i \in M$$

Die Elemente, welche eine Kante repräsentieren, befinden sich genau dann in einer Teilemenge eines Knoten, wenn die Kante am Knoten endet.

$$e_i \in S_j \Leftrightarrow \exists k_i = \{v_i, v_x\} \text{mit beliebigen x}$$

für alle Kanten $k_i \in E$ (Laufzeit bis zu n^2)
  für alle Knoten $v_j \in k_i$ (Laufzeit 2)
    speichere e_i in S_j

$$\Rightarrow T(n) \in O(n^2)$$

$$\Rightarrow \text{SC ist NP-Hart} \wedge SC \in NP \Rightarrow \text{SC ist NP-Vollständig}$$

Richtige Idee, aber ihr schreibt alles so umständlich auf! Reduktionen müssen nicht so mmathematischßein.