Gruppe **17**
Berechenbarkeit & Komplexität

Philipp Hochmann, 356148
Felix Kiunke, 357322

Blatt 9

Aufgabe 9.1

(a)

Sei G eine Kodierung der Adjazanzmatrix des Graphen, wobei in der Matrix entweder N steht (keine Kante), oder das binär kodierte Gewicht der Kante.
y bestimme nun eine Teilmenge des durch G dargestellten Graphen. y ist eine kodierte Matrix, deren Elemente entweder 0 (im Spannbaum nicht enthalten) oder 1 (im Spannbaum enthalten) sind. $|y| < |(G, c)|$, da jedes Element von G mindestens einstellig ist, also $|y| \leq |G|$. Als Polynom kann daher $p(x) = x$ gewählt werden.

Der Algorithmus V muss nun die Kreisfreiheit und Summe der durch y angegebenen Lösung bestimmen.
Es wird eine Tiefensuche auf den von y kodierten Graphen ausgeführt. Dabei werden die Kantengewichte aus G addiert, um am Ende schauen zu können, ob c erreicht wird. Wenn die Tiefensuche einen Zykel erkennt, wird verworfen. Wenn die Tiefensuche nicht alle Knoten aus y erreicht, wird verworfen, weil der Graph nicht zusammenhängend ist und folglich kein valider Spannbaum ist. Ist das alles nicht der Fall und die Summe der Kantengewichte tatsächlich $\geq c$, wird akzeptiert.

Komplexität: Tiefensuche benötigt $|V| + |E| = \mathcal{O}(|V|^2)$ Schritte. Überprüfen, ob alle Knoten erreicht wurden, ist in linearer Zeit möglich. Ergo ist der Algorithmus polynomiell.

(b)

y sei die binäre Kodierung einer Zahl ungleich 1 oder w, durch die sich w restfrei teilen lässt. Diese Zahl muss kleiner sein, die Kodierung lässt sich daher wieder durch $p(x) = x$ abschätzen. Der Algorithmus dividiert w durch y. Wenn kein Rest bleibt akzeptiert der Algorithmus, sonst verwirft er.
Schriftliche Division hat eine lineare Komplexität bezüglich der Länge der Zahlen.

(c)

Seien G_1 und G_2 kodierte Adjazenzmatrizen der jeweiligen Graphen. Das Zertifikat y sei nun eine Permutation der Knoten von G_1.
$|y| \leq G_1 \# G_2$, da die Permutation aus $|V|$ Vertauschungen besteht, welche jeweils mit $log(|V|)$ Zeichen kodiert werden können.

Nun muss der Algorithmus verifizieren, dass durch Vertauschungen der Spalten und Zeilen der Matrix von G_1 die Matrix von G_2 entsteht. Dies benötigt $3 \cdot |V|^2$ Schritte (Für jede Vertauschung eine Spalte und eine Zeile mit jeweils $|V|$ Einträgen und abschließendes Durchlaufen der Matrix mit Gesamtgröße $|V|^2$)

Aufgabe 9.2

(a)

Zuerst: Binäre Suche mit Hilfe der von BPP-E im Bereich 1 bis N (Anzahl der Objekte). Dadurch wird das minimale k gefunden.
Es werden so lange Elemente aus der Menge entfernt, bis BPP-E meldet, dass ein Bin weniger benötigt wird. Die entfernten Elemente liegen im Resultat in einem gemeinsamen Bin. Dies wird so lange wiederholt, bis die Menge leer ist.

Laufzeit: $p(N) \cdot \lceil log(N) \rceil + N \cdot (\sum_{i=1}^{N} p(N) - i) \leq p(N)^4 + p(N) \cdot \lceil log(N) \rceil \leq p(N)^4 + p(N) \cdot N$

(b)

Aufgabe 9.3

1. DoubleSat $\in NP$, da es einen Polynomialzeitverifizierer gibt, wobei das Zertifikat zwei Lösungen der Formel ist.
2. Als Sprache für die Reduktion wird Sat gewählt.
3. Reduktionsabbildung:
Es soll eine Instanz von Sat auf DoubleSat abgebildet werden.
Die Instanz von Sat wird für jede Klausel um ein Literal erweitert, die nicht-negiert nur in ihrer Klausel steht, und nirgends sonst.
4. Dies kann sehr einfach in polynomieller Zeit berechnet werden, da die Änderung linear in der Anzahl der Klauseln ist.
5. Korrektheit:
$x \in$ Sat $\implies$ es gibt erfüllende Belegung $\implies$ $f(x)$ hat zwei erfüllende Belegungen (die eigentliche und die, bei der alle neuen Literale positiv belegt sind) $\implies$ $f(x) \in$ DoubleSat.
$x \notin$ Sat $\implies$ es gibt keine erfüllende Belegung $\implies$ $f(x)$ hat eine erfüllende Belegung (die, bei der alle Literale positiv belegt sind) $\implies$ $f(x) \notin$ DoubleSat.