EBKFS Cebeci, Yascha Mat# 6254677 Übungsgruppe Nr.4
Dietrich, Bernhard Mat# 6256800
Fernhomberg, Lars Mat# 6256030
Kniesburges, Sebastian Mat# 6257120

Aufgabe 41:

1. $L_1 := \{(\langle M \rangle, w, d) \mid d \in N$ und M akzeptiert die Eingabe w nach höchstens d Schritten $\}$

L_1 kann mit der folgenden DTM entschieden werden:
Man lässt eine beliebige DTM durch die Eingabe w wandern, allerdings maximal d Schritte lang. Sollte die DTM anhalten, so akzeptiert man und geht in den Endzustand. Sollte sie nach d Schritten noch nich angehalten haben so verwirft man die Eingabe.

2. $L_2 := \{(\langle M \rangle, w, d) \mid d \in N$ und M akzeptiert die Eingabe w nach mindestens d Schritten $\}$

L_2 kann nicht entschieden werden, jedoch akzeptiert:
Man lässt wieder eine beliebige DTM durch die Eingabe w wandern. Wieder zählt man die Schritte mit. Sollte die DTM vor d Schritten halten so verwirft man die Eingabe. Sollte die DTM nach d Schritten halten so akzeptiert man. Dies ist keine Entscheidung da die DTM nach den d Schritten immer noch in eine Endlosschleife landen könnte, bei der sie niemals hält.

Aufgabe 42:

$L := \{\langle M_1 \rangle \langle M_2 \rangle \mid L(M_1) \subset L(M_2)\}$

z.Z: L ist nicht entscheidbar mittels Reduktion

Akzeptanzproblem $\leq L$

⇨ Es muss eine berechenbare f Funktion geben, für die gilt:

$\langle M \rangle x \in A \rightarrow f(\langle M \rangle) \in L$
$\langle M \rangle x \notin A \rightarrow f(\langle M \rangle) \notin L$

$\Rightarrow$ $f(\langle M \rangle)$ soll zwei DTMs berechnen für die gilt $\langle M' \rangle \langle M'' \rangle \in L$

Lösung:

- Um diese DTMs *M'* und *M''* zu berechnen muss f die DTM *M* nicht grossartig verändern. Wir wissen das $\langle M \rangle x \in A$.
- $\langle M'' \rangle := \langle M \rangle$
- Ausserdem wird $L(M') := L(M) / x$. Dazu nimmt f die DTM *M* und modifiziert die Übergangsfunktion δ so, dass x nicht mehr von *M* akzeptiert wird.
- Danach wird *M'* := *M* gesetzt.

"$\Rightarrow$"
$\langle M \rangle x \in A \ \wedge L(M') = L(M) / x \wedge L(M'') = L(M) \Rightarrow L(M') \subset L(M'') \Rightarrow \langle M' \rangle \langle M'' \rangle \in L$

"$\Leftarrow$"
$\langle M \rangle x \notin A \wedge L(M') = L(M) / x \Rightarrow L(M') = L(M) \wedge L(M'') = L(M) \Rightarrow L(M') = L(M'')$
$\Rightarrow L(M') \not\subset L(M'') \Rightarrow \langle M' \rangle \langle M'' \rangle \notin L$

Aufgabe 43:

Man kann durch eine Diagonalisierung nicht zeigen, dass *S* nicht entscheidbar ist. Man könnte höchstens versuchen mit ihr zu zeigen, dass weder *S* noch $\overline{S}$ rekursiv aufzählbar sind. Daraus würde dann folgen, dass *S* nicht entscheidbar ist.
Dies kann man sich allerdings auch einfacher verdeutlichen, denn um zu testen ob die von einer DMT akzeptierten Sprache leer ist, müsste man ja alle unendlich vielen Eingaben durchtesten. Denn bei KEINER Eingabe darf die DTM in einen akzeptierenden Zustand geraten. Da es ausserdem unendlich viele verschiedene DTMs gibt, kann man nicht undendlich mal undendlich viele Eingaben durchprobieren.
$\rightarrow S$ ist überabzählbar unendlich.

Aufgabe 44:

z.Z: $NP = Co - NP \quad \Leftrightarrow \quad \exists L : L$ ist NP-vollständig und $\overline{L} \in NP$.

"$\Rightarrow$"
$NP = Co - NP$.
Da NP-vollständig $\neq$ der leeren Menge ist gilt weiter:
$\exists L : L \in NP - vollständig \Rightarrow L \in NP \wedge L \in Co - NP$, da $NP = Co - NP$
$\Rightarrow \overline{L} \in NP$

"$\Leftarrow$"
$\exists L : L$ ist NP-vollständig und $\overline{L} \in NP$.
$\Rightarrow L \in Co - NP$
"$\subseteq$" $F \in NP$ beliebig gewählt.
z.Z $F \in Co - NP$
L ist NP-vollständig und $F \in NP$
$\Rightarrow F \leq_p L$
$\Rightarrow$ es Existiert eine Funktion f mit:
$n \in F \rightarrow f(n) \in L \quad \wedge \quad n \notin F \rightarrow f(n) \notin L$
$\Rightarrow n \in \overline{F} \Leftrightarrow f(n) \in \overline{L}$
$L \in Co - NP \Leftrightarrow \overline{L} \in NP$
$\Rightarrow$ Es gibt eine NTM *M* die $\overline{L}$ in polynomieller Zeit entscheidet
Konstruktion:
- Eingabe w
- Berechne $f(w)$
- Simuliere *M* auf $f(w)$

- Akzeptiere, falls $f(w) \in \overline{L}$
- Dann gilt: $w \in \overline{F} \Rightarrow \overline{F} \in NP$

$\Rightarrow F \in Co-NP$

$\Rightarrow NP \subseteq Co-NP$

"$\supseteq$" $F \in Co-NP$ beliebig gewählt.

z.Z: $F \in NP$

L ist NP-vollständig und $F \in NP$

$\Rightarrow F \leq_p L$

$\Rightarrow$ es Existiert eine Funktion f mit:

$n \in \overline{F} \rightarrow f(n) \in L \quad \wedge \quad n \notin \overline{F} \rightarrow f(n) \notin L$

$\Rightarrow n \in F \Leftrightarrow f(n) \in \overline{L}$

$L \in Co-NP \Leftrightarrow \overline{L} \in NP$

$\Rightarrow$ Es gibt eine NTM M die $\overline{L}$ in polynomieller Zeit entscheidet

Konstruktion:

- Eingabe w
- Berechne $f(w)$
- Simuliere M auf $f(w)$
- Akzeptiere, falls $f(w) \in \overline{L}$
- Dann gilt: $w \in \overline{F} \Rightarrow \overline{F} \in NP$

$\Rightarrow F \in Co-NP$

$\Rightarrow NP \supseteq Co-NP$

$\Rightarrow NP = Co-NP$

Aufgabe 45:

z.Z: $INDEPENDENTSET \leq_p BIN\_PROG$

Wir zeigen dies durch:

$IS \leq_p CLIQUE \leq_p BIN\_PROG$

⇨ Es muss eine berechenbare Funktion Θ geben für die gilt:

$\langle G \rangle, k \in IS \rightarrow \Theta(\langle G \rangle k) \in QLIQUE$

$\langle G \rangle, k \notin IS \rightarrow \Theta(\langle G \rangle k) \notin QLIQUE$

Lösung:

- Wenn wir einen Graphen $G = (V, E)$ mit einer Zahl $k \in \{0,...,|V|\}$ so dass gilt: $\langle G \rangle k \in IS$ dann brauch die Funktion Θ nichts anders zu tun als den Graphen zu invertieren.
- D.h: Jede Kannte $(v_i, v_j) \in E$ für $i \neq j$ und $v_i, v_j \in V$ muss gelöscht werden und zwischen allen Knoten für die gilt $v_x, v_y \in V$ und $(v_x, v_y) \notin E$ muss erzeugt werden.

"$\Rightarrow$"
Der Graph besaß ein *INDEPENDENTSET* der Grösse k. D.h: es gab k Knoten bei denen keiner mit keinem Verbunden war. Jetzt wurde der Graph durch Θ wie geschildert invertiert und alle diese Kanten wurden gelöscht
$\Rightarrow$ Es gibt eine *QLIQUE* der Grösse k, Bei dem genau die k Knoten miteinander verbunden sind.
$\Rightarrow \langle G \rangle, k \in IS \rightarrow \Theta(\langle G \rangle k) \in QLIQUE$

"$\Leftarrow$"
Der Graph besaß kein *INDEPENDENTSET* der Grösse k. D.h: nach seiner Invertierung kann es ebenso wenig eine *QLIQUE* der Grösse k geben, bei dem genau k Knoten miteinander verbunden sind.
$\Rightarrow \langle G \rangle, k \notin IS \rightarrow \Theta(\langle G \rangle k) \notin QLIQUE$

Ausserdem ist die Funktion Φ offensichtlich total berechenbar.

- ⇨ Weiter muss es eine berechenbare Funktion Φ geben für die gilt:

$\langle G \rangle, k \in CLIQUE \rightarrow \Phi(\langle G \rangle k) \in BIN\_PROG$
$\langle G \rangle, k \notin CLIQUE \rightarrow \Phi(\langle G \rangle k) \notin BIN\_PROG$

Mit $\Phi(\langle G \rangle k) = \langle A, b \rangle \in BIN\_PROG$

Lösung:

- Damit *BIN_PROG* genau dann erfüllt wird wenn es eine *k-QLIQUE* in einem Graphen *G* gibt, muss das Ungleichungssystem der Matrix *A* nur dann durch einen 0-1-Vektor erfüllbar sein, wenn sie die Eigenschaften einer *k-QLIQUE* enthält.
- Wir bauchen mit der berechenbaren Funktion Φ eine Matrix *A* und einen Ergebnisvektor *b*, die abhänig vom *k* der *QLIQUE* und von den Kanten bzw Knoten des Graphen *G*.

- Wir bauen als 1. Teil der Matrix eine Inzidenzmatrix auf, die eine Zeilenlänge von $|E|$ und eine Spaltenlänge von $|V|$ hat. Der Egebnisteil für den Vektor *b* muss hier die Höchstanzahl der Kanten eines Knoten sein. Da ein Knoten innerhalb einer *k-QLIQUE* genau *k-1* Kanten besitzt muss der Ergebnisvektor , bei diesen Gleichungen schon mal *k-1* sein. Unter diesem 1. Teil der Matrix wird ein 2. Teil gesetzt. Der wiederrum eine Inzidenzmatrix wiederspiegelt. Allerdings so, dass jede Kante eines Knoten nicht durch 1 sondern durch -1 gesetzt wird und der Ergebnisvektor wird natürlich ebenfalls *–(k-1)* sein.
- Damit haben wir schon mal zwei bzw eine Eigenschaft eines Graphen mit einer *k-QLIQUE* abgedeckt. Die Knoten innerhalb der *QLIQUE* haben mindestens und höchstens (damit GENAU) *k-1* Kanten.
- Der 3. Teil der Matrix muss nun sicherstellen das höchstens und mindestens (damit wieder GENAU) $\frac{k(k-1)}{2}$ in dem Teilgraph, der die *QLIQUE*) darstellen soll, enthalten sind. Dafür schreiben wir 2 Zeilen zur Matrix *A* hinzu. In der 1. Zeile

komplett nur einsen und in der zweiten Zeile komplett nur zweien. Bei dem Ergebnisvektor ergibt sich dann natürlich für die 1. Zeile $\frac{k(k-1)}{2}$ und für die 2. Zeile $-(\frac{k(k-1)}{2})$.

- $\Rightarrow$ Die Matrix hat eine Zeilenlänge von $|E|$ und eine Spaltenlänge von $(2*|V|)+2$, der Ergebnisvektor *b* hat eine Spaltenlänge von $(2*|V|)+2$. Und der 0-1-Vektor *y* besitzt eine Spaltenlänge von $|E|$ und kann somit mir *A* multipliziert werden.
- $\Rightarrow$ *y* kann eine beliebige Kantenmenge von *G* wiederspiegeln (1 für Kante enthalten und 0 für Kante nicht enthalten). Dieser Vektor wird mit *A* multipliziert und somit überprüft ob für die Kanten und Knoten, die im Teilgraph enthalten sind, alle Eigentschaften enthalten sind.

"$\Rightarrow$"
Der Graph besitzt eine *k-QLIQUE.* $\Rightarrow$ es gibt eine Kantenmenge *y* die alle Eigenschaften der Matrix *A* erfüllt. Jeder Knoten dieser Kanten hat mindestens und höchstens *k-1* Kanten und alle Knoten zusammen besitzen genau $\frac{k(k-1)}{2}$ Kanten. Dieses ist dann genau ein Teilpgraph von *G* der eine *k-QLIQUE* entspricht.
$\Rightarrow \langle G\rangle, k \in CLIQUE \rightarrow \Phi(\langle G\rangle k) \in BIN\_PROG$

"$\Leftarrow$"
Besitzt *G* keine *k-QLIQUE.* $\Rightarrow$ es gibt keine Kantenmenge *y* die alle Eigenschaften der Matrix *A* erfüllt.
$\Rightarrow \langle G\rangle, k \notin CLIQUE \rightarrow \Phi(\langle G\rangle k) \notin BIN\_PROG$

$\Rightarrow$ Das BIN_PROG Problem ist *NP(vollständig).*

Ausserdem ist die Funktion Φ offensichtlich total berechenbar. Man hätte natürlich auch sehr einfach die direkte Reduktion $INDEPENDENTSET \leq_p BIN\_PROG$ zeigen können.

- ⇨ Es muss eine totale und berechenbare Funktion Σ geben für die gilt:

$\langle G\rangle, k \in IS \rightarrow \Sigma(\langle G\rangle k) \in BIN\_PROG$
$\langle G\rangle, k \notin IS \rightarrow \Sigma(\langle G\rangle k) \notin BIN\_PROG$

Lösung:
- Man kombiniert auf die Eingabe einfach beide Funktionienen und hintereinander.
- Wie oben geschilder kommt man dann zu dem selben Schluss.

Aufgabe 46:

Die untere Schranke für die Mindestanzahl der benötigten Behälter:

$$\left\lceil \frac{\sum_{i=1}^{n} a_i}{1} \right\rceil = \sum_{i=1}^{n} a_i = B_{optimal}$$

Sei b die anzahl der vom Algorithmus benötigten Behälter. Es kann höchstens 1 Behälter nur bis zur Hälfte gefüllt sein. Gäbe es zwei so wären die Gegenstände in einen Behälter vereint worden.

$\Rightarrow$ im Schlimmsten Fall ist ein Behälter halb voll und $b-1$ Behälter über die Hälfte gefüllt.

$$\Rightarrow \frac{b-1}{2} < \sum_{i=1}^{n} a_i$$

$$\Rightarrow b-1 < 2 * \sum_{i=1}^{n} a_i$$

$$\Rightarrow b \leq 2 * \sum_{i=1}^{n} a_i$$

$$\Rightarrow b \leq 2 * B_{optimal}$$

$\Rightarrow$ *Approximationsfaktor* von 2