Universität Siegen
Lehrstuhl Theoretische Informatik
Markus Lohrey

Übungsblatt 7

Aufgabe 1. Welche der folgenden Probleme sind in **P**? Welche sind in **NP**?

(a) Das Wortproblem für reguläre Sprachen.

(b) Das Wortproblem für kontextfreie Sprachen.

(c) Das Wortproblem für kontextsensitive Sprachen.

(d) Das Erfüllbarkeitsproblem für aussagenlogische Formeln (SAT).

(e) Das spezielle Halteproblem.

Lösung.

(a) Das Problem ist in **P** und somit auch in **NP**; siehe nächste Teilaufgabe.

(b) Das Problem ist in **P** und somit auch in **NP**. Ein polynomialzeit Algorithmus für diese Problem geht etwa auf Cocke-Younger-Kasami zurück.

(c) Das Problem ist vollständig für die Komplexitätsklasse **PSPACE**, die aus jenen Problemen besteht, die in polynomiellem Platz gelöst werden können. Zwar gilt

$$\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE},$$

aber es ist über keine dieser Inklusionen bekannt, ob sie strikt ist. Insbesondere wissen wir also nicht, ob das Problem in **NP** oder gar in **P** liegt.

(d) Das Problem ist **NP**-vollständig und somit in **NP**. Ob das Problem in **P** liegt, und somit $\mathbf{P} = \mathbf{NP}$ gilt, ist nicht bekannt.

(e) Das Problem ist unentscheidbar. Daher liegt es weder in **P** noch in **NP**.

Aufgabe 2. Beweisen Sie die folgenden Abschlusseigenschaften, wobei $A, B, L \subseteq \Sigma^*$.

(a) Sind A und B in **P** (bzw. **NP**), dann auch $A \cup B$ und $A \cap B$.

(b) Sind A und B in **P** (bzw. **NP**), dann auch $A \cdot B = \{uv \in \Sigma^* \mid u \in A, v \in B\}$.

(c) Ist L in **NP**, dann auch $L^* = \{w \in \Sigma^* \mid w = u_1 \cdots u_n \text{ mit } u_1, \ldots, u_n \in L\}$.

Aufgabe 3. Zeigen Sie: Die Variante des Problems SUBSETSUM (siehe Folie 186), bei der die zu erreichende Summe $t \in \mathbb{N}$ als Teil der Eingabe unär kodiert ist, liegt in **P**.
Hinweis: Betrachten Sie für $t, w_1, \ldots, w_n$ alle Instanzen $s, w_1, \ldots, w_k$ mit $s \leq t$ und $k \leq n$.

Lösung. Zur Lösung der unären Variante des Problems SUBSETSUM konstruieren wir eine Tabelle T, die für jedes Paar (s, k) mit $0 \leq s \leq t$ und $0 \leq k \leq n$ einen Wahrheitswert enthält. Dabei gibt der Wert $T(s, k)$ an, ob die Instanz $s, w_1, \ldots, w_k$ eine Lösung besitzt, also ob es eine Menge von Indizes $I \subseteq \{1, \ldots, k\}$ mit $s = \sum_{i \in I} w_i$ gibt.
Wir füllen nun die Tabelle T nun Zeile für Zeile aus. Die erste Zeile der Tabelle initialisieren wir durch $T(0, k) \leftarrow 1$ für alle $0 \leq k \leq n$, da hier stets die Indexmenge $I = \emptyset$ eine gültige Wahl ist. Für $1 \leq s \leq t$ setezen wir zunächst $T(s, 0) \leftarrow 0$, da die einzige zulässige Indexmenge $I = \emptyset$ die Summe $\sum_{i \in I} w_i = 0 \neq s$ liefert. Für $1 \leq k \leq n$ bestimmen wir dann $T(s, k)$ wie folgt. Ist $T(s, k-1) = 1$, so setzen wir $T(s, k) \leftarrow 1$, da eine gültige Wahl der Indexmenge I für $(s, k-1)$ auch eine gültige Wahl für (s, k) darstellt. Hierbei wird der Wert w_k für die Summe nicht benötigt. Andernfalls, wenn also der Wert w_k für die Summe benötigt wird, setzen wir dementsprechend $T(s, k) \leftarrow T(s - w_k, k-1)$ oder, falls dies nicht sinnvoll ist, da $w_k = 0$ oder $w_k > s$ gilt, dann setzen wir $T(s, k) \leftarrow 0$.
Am Ende der Berechnung geben wir den Wert $T(t, n)$ aus. Für jedem Schritt benötigen wir höchstens zwei Tabellenabfragen sowei eine Subtraktion und und zwei Vergleiche. Da die Tabelle T nur polynomiell viele Einträge besitzt und die auftretenden Zahlen polynomiell beschränkt werden können, lässt sich die gesamte Berechnung in polynomieller Zeit durchführen. (Dabei habe wir benutzt, dass die Eingabe in dieser Variante mindestens die Länge $t + n$ hat.)

Aufgabe 4. Beweisen Sie die folgenden Aussagen.

(a) Die Relation $\leq_p$ ist reflexiv und transitiv.

(b) Genau dann gilt $L \in \mathbf{P}$, wenn $L \leq_p \{\varepsilon\}$.

Lösung.

(a) Seien $A \subseteq \Sigma^*$, $B \subseteq \Gamma^*$ und $C \subseteq \Delta^*$ Sprachen mit $A \leq_p B$ und $B \leq_p C$. Nach Definition der Relation $\leq_p$ existieren also totale, in polynomieller Zeit berechenbare Funktionen $f \colon \Sigma^* \to \Gamma^*$ und $g \colon \Gamma^* \to \Delta^*$, sodass für alle $x \in \Sigma^*$ und $y \in \Gamma^*$ gilt:

$$x \in A \iff f(x) \in B \quad \text{und} \quad y \in B \iff g(y) \in C.$$

Die Komposition $h = g \circ f \colon \Sigma^* \to \Delta^*$ ist dann eine totale Funktion mit

$$x \in A \iff f(x) \in B \iff h(x) = g(f(x)) \in C.$$

Es verbleibt zu zeigen, dass die Komposition $h = g \circ f$ ebenfalls in polynomieller Zeit berechenbar ist. Seien dazu $P_f(n)$ und $P_g(n)$ Polynome, sowie M_f und M_g Turningmaschinen, welche die Funktionen f bzw. g auf Eingaben der Länge n in höchstens $P_f(n)$ bzw. $P_g(n)$ Schritten berechnen. Wir nehmen an, dass dabei der Schreib-Lese-Kopf am Ende der Berechnung jeweils wieder am Anfang des Ein-Ausgabe-Bandes der Turingmaschine steht. Wir können zudem annehmen, dass beide Polynome monoton wachsende Funktionen in $n \geq 0$ sind: Ersetzen wir die Koeffizienten eines Polynoms $P(n)$ durch ihren jeweiligen Absolutbetrag, dann erhalten wir nämlich ein solches Polynom $Q(n)$ mit $P(n) \leq Q(n)$ für alle $n \in \mathbb{N}$.

Unsere Turingmaschine M_h arbeitet nun wie folgt. Auf Eingabe $x \in \Sigma^*$ simulieren wir zuerst M_f auf der Eingabe und anschließend M_g auf der Ausgabe von M_f. Es ist leicht einzusehen, dass diese Maschine dann in der Tat h berechnet. Die dafür benotigte Zeit $t(x)$ (d.h., die Anzahl der Berechnungschritte der Maschine M_h auf Eingabe von $x \in \Sigma^*$) lässt sich dann durch

$$t(x) \leq P_f(|x|) + P_g(|f(x)|)$$

abschätzen. Da die Berechnung von $f(x) \in \Gamma^*$ in höchstens $P_f(|x|)$ Schritten erfolgt, gilt zudem $|f(x)| \leq P_f(|x|) + |x|$. Wegen der Monotonie von $P_g(n)$ liefert dies nun

$$t(x) \leq P_f(|x|) + P_g(P_f(|x|) + |x|) = P_h(|x|)$$

für das Polynom $P_h(n) \coloneqq P_f(n) + P_g(P_f(n) + n)$.

(b) Da $\{\varepsilon\} \in \mathbf{P}$ gilt, folgt aus $L \leq_p \{\varepsilon\}$ stets, dass auch $L \in \mathbf{P}$ gilt. Wir nehmen nun umegekehrt an, dass die Sprache $L \subseteq \Sigma^*$ in $\mathbf{P}$ liegt. Sei $a \in \Sigma$ beliebig, aber fest. Die Funktion $f\colon \Sigma^* \to \Sigma^*$ mit $f(x) = \varepsilon$ falls $x \in L$, sowie $f(x) = a$ falls $x \notin L$ ist total und in polynomieller Zeit berechenbar. Letzteres liegt daran, dass wir wegen $L \in \mathbf{P}$ die Fälle $x \in L$ und $x \notin L$ in polynomieller Zeit entscheiden können.

Nach Konstruktion gilt für alle $x \in \Sigma^*$ nun $x \in L$ genau dann, wenn $f(x) \in \{\varepsilon\}$. Also liefert die Funkion f die gewünschte Reduktion, d.h. es gilt $L \leq_p \{\varepsilon\}$.

Aufgabe 5. Wir betrachten das Problem k-COLORABILITY, bei dem auf Eingabe eines ungerichteten Graphen $G = (V, E)$ entschieden werdern soll, ob dieser *k-färbbar* ist:

Existiert $\chi\colon V \to \{1, \ldots, k\}$, sodass $\chi(u) \neq \chi(v)$ für alle $\{u, v\} \in E$?

(a) Zeigen Sie für alle $k \geq 1$, dass k-COLORABILITY $\leq_p$ $(k+1)$-COLORABILITY.

(b) Zeigen Sie, dass das Problem 2-COLORABILITY in $\mathbf{P}$ liegt.

(c) Was können Sie hieraus für das Problem 3-COLORABILITY folgern?

Lösung.

(a) Eine Reduktion k-COLORABILITY $\leq_p$ $(k+1)$-COLORABILITY ist etwa durch die Abbildung gegeben, die jedem Graphen $G = (V, E)$ den Graphen $G' = (V', E')$ zuordnet mit $V' = V \cup \{u\}$, wobei $u \notin V$, und $E' = E \cup \{\{u, v\} \mid v \in V\}$. Es wird also ein neuer Knoten u hinzugefügt und mit jedem anderen Knoten verbunden.

Diese Abbildung ist (bei geeigneter Codierung) in polynomieller Zeit berechenbar. Ist der Graph G dabei k-färbbar, so ist der resultierende Graph $(k+1)$-färbbar, denn wir können eine gültige Färbung $\chi\colon V \to \{1, \ldots, k\}$ von G stets zu einer gültigen Färbung $\chi'\colon V' \to \{1, \ldots, k{+}1\}$ von G' fortsetzen. Dazu setzen wir einfach $\chi'(v) = \chi(v)$ für alle $v \in V$ und $\chi'(u) = k{+}1$. Sei umgekehrt $\chi'\colon V' \to \{1, \ldots, k{+}1\}$ eine gültige Färbung von G'. Bis auf vertauschen der Farben können wir dann annehemen, dass $\chi'(u) = k + 1$ gilt. Wegen $\{u, v\} \in E'$ muss $\chi'(v) \neq \chi'(u) = k + 1$ für alle $v \in V$ gelten. Also ist $\chi\colon V \to \{1, \ldots, k\}$ mit $\chi(v) = \chi'(v)$ eine gültige Färbung von G mit k Farben.

(b) Es genügt 2-Färbbarkeit für nicht-leere zusammenhängende Graphen in polynomieller Zeit zu entscheiden, denn ein Graph ist k-färbbar genau dann, wenn jede seiner Zusammenhangskomponenten k-färbbar ist und diese Zusammenhangskomponenten lassen sich (bei geeigneter Codierung) in polynomieller Zeit bestimmen.

Ein nicht-leerer zusammenhängender Graph $G = (V, E)$ besitzt aber entweder keine gültige Färbung mit 2 Farben, oder genau zwei solche Färbungen, wobei die eine dann jeweils aus der anderen durch Vertauschung der beiden Farben entsteht. Ist nämlich die Farbe $\chi(u)$ eines Knotens $u \in V$ festgelegt, dann verbleiben für alle Nachbarknoten $v \in V$ (d.h. mit $\{u, v\} \in E$) nur die jeweils andere Farbe. Wir können also mittels Breiten- oder Tiefen-Suche (ausgehend von einem beliebigen Startknoten) versuchen, den Graphen gültig mit zwei Farben zu färben. Stoßen wir dabei auf eine bereits gefärbten Knoten, dem nun die andere Farbe zugewiesen werden soll, dann können wir schließen, dass der Graph nicht 2-färbbar ist. Andernfalls erhalten wir nach polynomiell vielen Schritten eine gültige Färbung.

(c) Aus den vorherigen Aufgabenteilen ist bekannst, dass 2-COLORABILITY $\in$ **P** gilt und, dass 2-COLORABILITY $\leq_p$ 3-COLORABILITY gilt. Über 3-COLORABILITY lässt sich daraus keine weitere Aussage ableiten. Anschaulich: Wir haben lediglich gezeigt, dass das "leichtere" von zwei Problemen in der Tat "leicht" ist.

Tatsächlich ist das Problem k-COLORABILITY für alle $k \geq 3$ **NP**-vollständig.