Aufgaben zum Tut am 13.02.2007

Thomas Pajor

13. Februar 2007

Aufgabe 1) [Wiederholung]

Sei $L := \{a^j b^k c^l \mid j, k, l \geq 0 \text{ und } k = l \text{ falls } j = 1\}$ mit $\Sigma := \{a, b, c\}$.

(a) Zeigen Sie, dass L alle Bedingungen des Pumping Lemma erfüllt.

(b) Zeigen Sie, dass L trotzdem nicht regulär ist.

Lösung.

(a) Sei n die Pumping Lemma Zahl. Betrachte alle Wörter $w \in L$ mit $|w| > n$. Es gibt nun folgende drei Fälle zu unterscheiden:

- w enthält genau ein a. Das Wort hat also die Form ab^kc^k. Wähle als Zerlegung $w = uvx$ mit $u = \varepsilon$ und $v = a$. Es ist also $|uv| \leq n$ und $|v| > 0$ erfüllt. Damit ist aber auch für jedes $i \in \mathbb{N}_0$ das Wort $uv^ix = a^ib^kc^k \in L$.
- w enthält genau zwei as. Wähle als Zerlegung $w = uvx$ mit $u = \varepsilon$ und $v = aa$. Damit ist auch $uv^ix \in L$ für alle $i \in \mathbb{N}_0$.
- w enthält kein a oder mehr als 2 as. Wähle als Zerlegung $w = uvx$ wieder $u = \varepsilon$ und $v = w_1$ wobei w_1 der erste Buchstabe von w ist. Ist $w_1 = a$, so gilt $|w|_a > 2$ und es gilt für alle $i \in \mathbb{N}_0$ dass $|uv^ix|_a > 1$ und somit ist $uv^ix \in L$ für alle $i \in \mathbb{N}_0$. Ist $w_1 \neq a$, so ist $w_1 = b$ oder $w_1 = c$ und damit auf triviale Weise $uv^ix \in L$ für alle $i \in \mathbb{N}_0$.

L erfüllt also die Bedingungen des Pumping Lemmas.

(b) Wir zeigen $\operatorname{index}(R_L) = \infty$, wobei R_L die Nerode Relation von L bezüglich Σ^* ist. Dann folgt mit dem Satz von Nerode, dass L nicht regulär ist.

Betrachte für beliebiges $k \geq 0$ folgende Äquivalenzklassen $A_k := [ab^{n+k}c^n]_\sim$, also

$$
\begin{aligned}
&[ab^nc^n]_\sim := \{a, abc, abbcc, abbbccc, \ldots\}\\
&[ab^{n+1}c^n]_\sim := \{ab, abbc, abbbcc, \ldots\}\\
&[ab^{n+2}c^n]_\sim := \{abb, abbbc, abbbbcc, abbbbbccc, \ldots\}\\
&\vdots
\end{aligned}
$$

Jede dieser Klassen enthält nur äquivalente Wörter, denn es gilt für zwei Wörter $w_1, w_2 \in A_k$

$$w_1 z \in L \quad \Leftrightarrow \quad z = c^k \quad \Leftrightarrow \quad w_2 z \in L$$

Außerdem fallen keine zwei paarweise verschiedenen Klassen A_k und A_l zusammen, da für zwei beliebige Wörter $w_1 \in A_k$ und $w_2 \in A_l$ gilt

$$w_1 z \in L \quad \Leftrightarrow \quad z = c^k \quad \overset{l \neq k}{\Leftrightarrow} \quad z \neq c^l \quad \Leftrightarrow \quad w_2 z \notin L$$

$\Rightarrow \Sigma^*$ zerfällt über R_L in unendlich viele Äquivalenzklassen und damit folgt mit dem Satz von Nerode dass L nicht regulär ist.

Aufgabe 2. [Wiederholung]

Zeigen Sie: Das Halteproblem ist $\mathcal{NP}$–hart. Warum ist es nicht $\mathcal{NP}$–vollständig?

Lösung.

Wir reduzieren 3SAT auf das Halteproblem.

Sei $\mathcal{M}$ eine Turingmaschine, die als Eingabe eine 3SAT Instanz $I := (C, P := \{P_1, \ldots, P_n\})$ erhält wobei C eine Menge von Klauseln über P ist. $\mathcal{M}$ arbeite wie folgt:

(1) Schreibe einen Bitvektor X der Länge $|P|$ auf das Band und initialisiere ihn mit lauter Nullen. $\mathcal{M}$ behandle den Bitvektor X wie folgt: An der i–ten Stelle von X steht genau dann eine 1, wenn P_i als wahr interpretiert wird.

(2) Prüfe ob X eine erfüllende Variablenbelgung für I ist. Falls ja: stoppe. Sonst überschreibe X durch seinen lexikographischen Nachfolger und prüfe erneut.

(3) Falls $X = (1, \ldots, 1)$ gehe in eine Endlossschleife (Beispielsweise indem der Kopf unendlich lange nach Rechts gefahren wird).

Betrachte nun eine 3SAT Instanz $I := (C, P)$. Wir konstruieren dazu eine Instanz $I' := (\langle \mathcal{T} \rangle, w)$ des Halteproblems wie folgt:

- $\langle \mathcal{T} \rangle := \langle \mathcal{M} \rangle$
- w ist die Menge der Klauseln sowie die aussagenlogischen Variablen in einer geeigneten Kodierung.

Die Transformation ist offensichtlich polynomiell.

Nach Konstruktion von $\mathcal{M}$ hält die Turingmaschine, wenn die als Eingabe übergebene 3SAT Instanz eine erfüllende Belegung hat, denn es werden in Schritt (2) alle möglichen Variablenbelegungen für P überprüft. Hat I keine erfüllende Belegung so geht $\mathcal{M}$ in Schritt (3) in eine Endlosschleife, und terminiert nicht. Wir konnten also zeigen dass das Halteproblem $\mathcal{NP}$–hart ist. $\square$

Das Halteproblem ist jedoch nicht $\mathcal{NP}$–vollständig, da dafür zusätzlich verlangt wird, dass es in $\mathcal{NP}$ liegt. Dies ist aber nicht gegeben, da das Halteproblem nicht entscheidbar ist. Dies ist also ein Beispiel für ein $\mathcal{NP}$–hartes Problem, das aber *nicht* $\mathcal{NP}$–vollständig ist.

Aufgabe 3. [Wiederholung]

Betrachten Sie das Problem SUBGRAPHENISOMORPHIE:

Gegeben: Zwei Graphen $G_1 := (V_1, E_1)$ und $G_2 := (V_2, E_2)$ mit $|V_2| < |V_1|$.

Frage: Gibt es eine Menge $U \subset V_1$ mit $|U| = |V_2|$ und einen Isomorphismus $\Phi : U \to V_2$, also eine bijektive Abbildung so, dass gilt

$$\{x, y\} \in E_1 \quad \Leftrightarrow \quad \{\Phi(x), \Phi(y)\} \in E_2 \qquad\qquad \forall x, y \in U$$

Also ist G_2 isomorph zu einem Teilgraph von G_1? Ein Beispiel ist in Abbildung (1) gegeben.

Zeigen Sie: SUBGRAPHENISOMORPHIE ist $\mathcal{NP}$-vollständig.

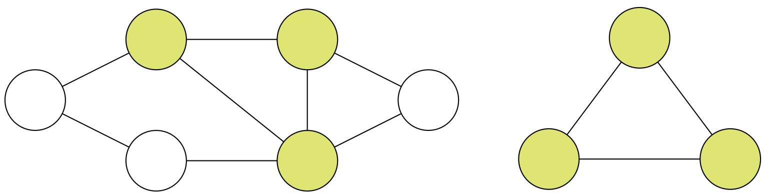

Abbildung 1: Beispiel für eine erfüllbare Instanz von SUBGRAPHENISOMORPHIE. Der Graph G_2 (rechts) ist isomorph zu der grünen Menge an Knoten in G_1.

Lösung.

Um zu zeigen dass SUBGRAPHENISOMORPHIE $\mathcal{NP}$-vollständig ist, zeigen wir zunächst, dass eine geratene Lösung in polynomieller Zeit daraufhin überprüft werden kann, ob sie eine gültige Lösung ist. Und im zweiten Schritt reduzieren wir ein bekanntes $\mathcal{NP}$-vollständiges Problem auf SUBGRAPHENISOMORPHIE. Wir werden dafür CLIQUE benutzen.

1. SUBGRAPHENISOMORPHIE $\in \mathcal{NP}$

Sei also $(U \subset V_1, \Phi)$ eine geratene Lösung. Folgender Algorithmus in Pseudocode überprüft ob U eine gültige Lösung ist:

Algorithmus 1 : VERIFIZIERE(U, Φ)

Eingabe : Eine geratene Lösung (U, Φ)
Ausgabe : Ob Φ ein gültiger Isomorphismus $\Phi : U \to V_2$ ist

für alle $x \in U$ **tue**
    für alle $y \in U$ **tue**
        wenn $\{x, y\} \in E_1$ *und* $\{\Phi(x), \Phi(y)\} \notin E_2$ **dann**
            return „nein!“
        wenn $\{x, y\} \notin E_1$ *und* $\{\Phi(x), \Phi(y)\} \in E_1$ **dann**
            return „nein!“

return „ja!“

Da wir nur zwei Schleifen über U durchlaufen und bei jedem Durchlauf überprüfen müssen ob eine Kante in E_1 bzw. E_2 enthalten ist, ist der Aufwand in $\mathcal{O}(|V_1|^2|E_1||E_2|)$, und somit polynomiell zur Eingabelänge.

Damit kann eine geratene Lösung von SUBGRAPHENISOMORPHIE in polynomieller Zeit verifiziert werden und es folgt SUBGRAPHENISOMORPHIE $\in \mathcal{NP}$.

2. SUBGRAPHENISOMORPHIE ist $\mathcal{NP}$-schwer

Wir zeigen CLIQUE $\leq_p$ SUBGRAPHENISOMORPHIE. Wir müssen also eine Abbildung f angeben, die jede Instanz $I = (V, E, K)$ von CLIQUE in eine Instanz $I' = (G_1 := (V_1, E_1), G_2 := (V_2, V_2))$ von SUBGRAPHENISOMORPHIE transformiert. Für f muss gezeigt werden, dass f in polynomieller Zeit berechnet werden kann, und außerdem muss gezeigt werden dass gilt

$$I \text{ enthält eine Lösung} \iff f(I) = I' \text{ enthält eine Lösung}$$

Sei also $I := (G := (V, E), K)$ eine beliebige Instanz von CLIQUE. Konstruiere dazu eine Instanz $I' := (G_1 := (V_1, E_1), G_2 := (V_2, V_2))$ von Subgraphenisomorphie wie folgt:

- $V_1 := V$
- $E_1 := E$
- $V_2 := \{1, \ldots, K\}$ (G_2 soll also genau K Knoten haben)
- $E_2 := \{\{a, b\} \mid a, b \in V_2, a \neq b\}$ (G_2 soll vollständig verbunden sein)

Die Konstruktion ist mit polynomiellem Zeitaufwand möglich, da mit $|V|_2 =: n$ gilt $|E_2| = \binom{n}{2} = \frac{n^2-n}{2} \in \mathcal{O}(n^2)$[1].

Nun bleibt noch zu zeigen, dass I genau dann eine Lösung enthält wenn auch $f(I) = I'$ eine Lösung enthält.

$\Rightarrow$: Sei I eine Instanz von CLIQUE, die eine Lösung enthält. Das heißt also in $G = (V, E)$ gibt es eine Clique der Größe K. Sei $U := \{u_1, \ldots, u_K\}$ die Menge der Knoten, die die Clique bilden. Dann ist $|U| = K$ und der durch U induzierte Teilgraph von G ein vollständiger Graph mit K Knoten. Da nach Konstruktion von I' der Graph G_2 ebenfalls ein vollständiger Graph mit K Knoten ist, ist G_2 isomorph zu dem durch U induzierten Teilgraph in G.

$\Rightarrow$ Die Instanz I' enthält ebenfalls eine Lösung.

$\Leftarrow$: Sei I' eine aus I (nach obigem Verfahren) hervorgegangene Instanz von SUBGRAPHENISOMORPHIE. Außerdem habe I' eine Lösung. Das heißt der Graph G_2 kommt als Teilgraph in G_1 vor. Sei U die Lösungsmenge der ausgewählten Knoten von G_1. Dann hatte aber auch I eine Lösung, da nach Konstruktion $G = G_1$ gilt, und U damit auch in G enthalten ist. Wegen $|U| = K$, enthält G eine Clique der Größe K.

$\Rightarrow$ SUBGRAPHENISOMORPHIE ist $\mathcal{NP}$-schwer.

Mit SUBGRAPHENISOMORPHIE $\in \mathcal{NP}$ folgt die $\mathcal{NP}$-Vollständigkeit. $\square$

[1] In einem vollständigen einfachen Graph $G = (V, E)$ gilt immer $|E| = \binom{|V|}{2}$, denn der Binomialkoeffizient $\binom{n}{k}$ beschreibt ja gerade die Anzahl k–elementiger Teilmengen aus einer n–elementigen Menge. In unserem Fall für $k = 2$ sind das gerade die Kanten zwischen den Knoten, da wir eine Kante durch eine zweielementige Teilmenge von V beschreiben.