Aufgaben zur Vorlesung **WiSe 22/23**
Berechenbarkeit und Komplexität **TU Berlin**
Weller/Froese/Kellerhals/Zschoche **05.01.2023**

10. Aufgabenblatt

(Besprechung in den Tutorien 09.01.2023–13.01.2023)

Aufgabe 1. O-Notation

Welche der folgenden Beziehungen sind korrekt?

(a) $n^{10} \in O((1,001)^n)$

(b) $2^n + n \in O(2^n \cdot n)$

(c) $3^n \in O(2^n \cdot n^3)$

(d) $(\ln n)^2 \in O(\sqrt{n})$

——————Lösungsskizze——————

Wir verwenden bei der Lösung dieser Aufgabe die folgende Aussage:
Seien $f, g : \mathbb{N} \to \mathbb{N}$. Dann gilt:

- Falls $\lim_{n\to\infty} \frac{f(n)}{g(n)} = c \in \mathbb{R}$, ist $f \in O(g)$.
- Falls $\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty$, ist $f \notin O(g)$.

(a) Ja, denn

$$\lim_{n\to\infty} \frac{n^{10}}{(1,001)^n} \overset{\text{l'Hospital (10 Mal)}}{=} \frac{10!}{(\ln(1,001))^{10}} \lim_{n\to\infty} \frac{1}{(1,001)^n} = 0 < \infty$$

(b) Ja, denn für $c := 1$ und $n_0 := 2$ gilt für alle $n \geq n_0$

$$2^n + n \leq 2^n + 2^n = 2^n \cdot 2 \leq c \cdot 2^n \cdot n.$$

Alternativ:

$$\lim_{n\to\infty} \frac{2^n + n}{2^n \cdot n} = \lim_{n\to\infty} \left(\frac{2^n}{2^n \cdot n} + \frac{n}{2^n \cdot n}\right) = \lim_{n\to\infty} \left(\frac{1}{n} + \frac{1}{2^n}\right) = 0 < \infty.$$

(c) Nein, denn mit $3^n = (\frac{2\cdot 3}{2})^n = 2^n \cdot (\frac{3}{2})^n$ erhalten wir

$$\lim_{n\to\infty} \frac{3^n}{2^n \cdot n^3} = \lim_{n\to\infty} \frac{2^n \cdot (\frac{3}{2})^n}{2^n \cdot n^3} = \lim_{n\to\infty} \frac{(\frac{3}{2})^n}{n^3} \overset{\text{l'Hospital (3 Mal)}}{=} \lim_{n\to\infty} \frac{(\ln \frac{3}{2})^3 \cdot (\frac{3}{2})^n}{3!} = \infty.$$

(d) Ja, denn

$$\lim_{n\to\infty} \frac{(\ln n)^2}{\sqrt{n}} \overset{\text{l'Hospital (2 Mal)}}{=} 8 \lim_{n\to\infty} \frac{1}{\sqrt{n}} = 0$$

Aufgabe 2. Nichtdeterministische Turing-Maschinen

(a) Geben Sie eine nichtdeterministische Turing-Maschine M an, sodass:

$$T(M) = \{ww \mid w \in \{0,1\}^*\}.$$

(b) Zeigen Sie, dass die folgende Sprache in NP liegt:

$$B := \{w \in \{0,1\}^* \mid w \text{ ist keine Binärdarstellung einer Primzahl}\}.$$

Geben Sie dafür die prinzipielle Arbeitsweise einer nichtdeterministischen Turing-Maschine M mit $T(M) = B$ und $\text{time}_M(n) \in O(n^c)$ an, wobei $c \in \mathbb{N}$.

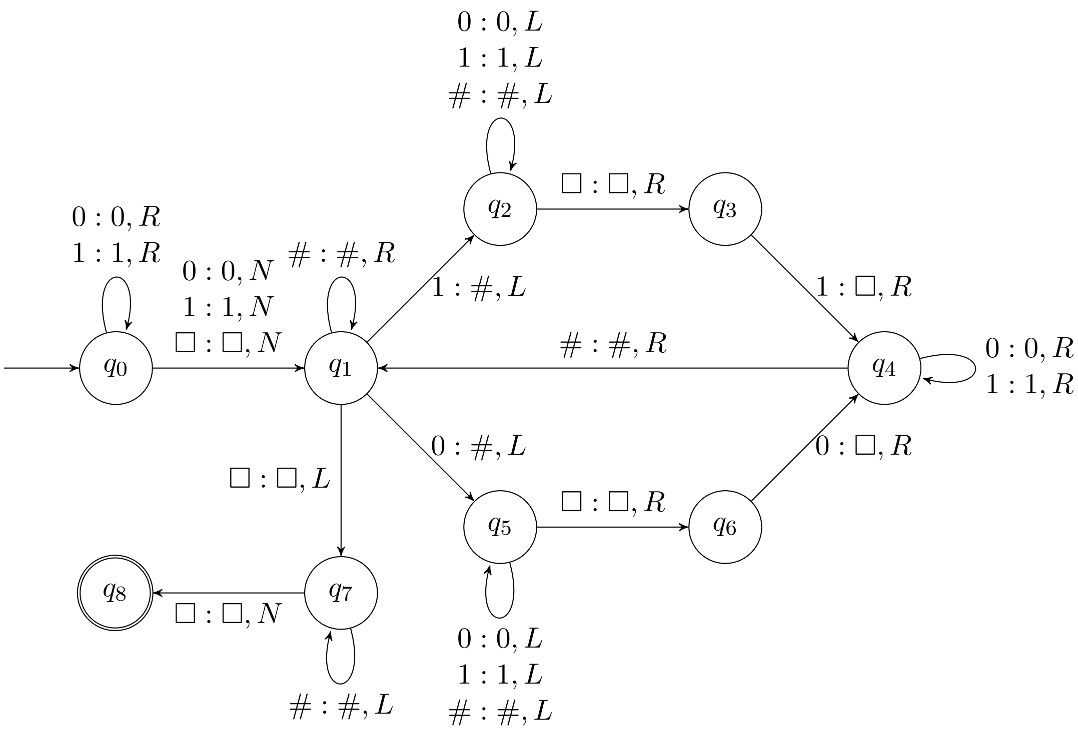
Hinweis: Sie dürfen verwenden, dass die Sprache

$$L = \{w\#a\#b \in \{0,1,\#\} \mid w,a,b \in \{0,1\}^* \text{ und } w = a \circ b\}$$

in P ist, wobei $\circ$ zwei Binärzahlen multipliziert.

——————Lösungsskizze——————

(a) Für die Turing-Maschine M mit Eingabealphabet $\{0,1\}$, Bandalphabet $\{0,1,\#\}$ und der folgenden graphischen Darstellung gilt $T(M) = \{ww \mid w \in \{0,1\}^*\}$.



Zugrundeliegende Idee: Der Übergang von q_0 nach q_1 zerteilt das Eingabewort in $a_1 \dots a_n b_1 \dots b_m$, wobei der der LS-Kopf auf b_1 steht. Nun wird iterativ für alle $i \in \{1,2,\dots,\min\{m,n\}\}$ b_i durch $\#$ ersetzt und anschließend a_i durch $\Box$ ersetzt, falls $a_i = b_i$. Dies wird durchgeführt bis alle b_j durch $\#$ ersetzt wurden. Daraus folgt $n \geq m$ und $a_i = b_i$ für alle $i \leq m$, wenn wir von q_1 zu q_7 gehen. Anschließend wird mit q_7 und q_8 überprüft, dass alle a_i durch $\Box$ ersetzt wurden. Also gilt $m = n$ und $a_i = b_i$ für alle $i \leq m = n$ genau dann, wenn es eine akzeptierende Konfigurationsfolge gibt.

(b) Die Turing-Maschine M schreibt zunächst ein $\# \in \Gamma \setminus \Sigma$ hinter das Eingabewort w. Anschließend schreibt M (nichtdeterministisch) ein Wort $a \in \{0,1\}^+$ der Länge höchstens $|w|$ aufs Band, wobei a nicht die Binärdarstellung von Eins ist. Dann wird erneut ein $\#$ geschrieben. Nun wird wieder (nichtdeterministisch) ein Wort $b \in \{0,1\}^+$ der Länge höchstens $|w|$ aufs Band geschrieben, wobei b nicht die Binärdarstellung von Eins ist. Abschließend akzeptiert M genau dann, wenn das Wort auf dem Band in L ist.

Beobachte, dass M höchstens polynomiell viele Schritte ausführen muss bevor Sie hält, da L in P ist.

Falls $w \in B$ ist, dann gibt es $a, b \in \{0,1\}^*$ der Länge höchstens $|w|$, sodass $w\#a\#b \in L$ ist. Somit gibt es in diesem Fall eine akzeptierende Konfigurationsfolge vom M.

Falls nun $w \notin B$ ist, dann gilt für alle $a, b \in \{0,1\}^*$ die M aufs Band schreiben könnte, dass $w \neq a \circ b$, da w die Binärdarstellung einer Primzahl ist. Somit gibt es in diesem Fall keine akzeptierende Konfigurationsfolge.

Aufgabe 3. 2-Coloring

Betrachten Sie das 2-Coloring-Problem.

> **2-Coloring**
> **Eingabe:** Ein ungerichteter Graph $G = (V, E)$.
> **Frage:** Gibt es eine totale Funktion $f : V \to \{1, 2\}$, sodass für jede Kante $\{u, v\} \in E$ gilt, dass $f(u) \neq f(v)$?

1. Zeigen Sie, dass 2-Coloring in NP liegt.
2. Zeigen Sie, dass 2-Coloring in P liegt.

——————Lösungsskizze——————

2-Coloring $\in$ NP: Eine Färbung f durch $|V|$ nichtdeterministische Schritte "raten" und anschließend in $\text{poly}(|V| + |E|)$ Schritten überprüfen, ob die Endpunkte jeder Kante unterschiedlich gefärbt sind.

2-Coloring $\in$ P:

Ein deterministischer Algorithmus für das 2-Coloring-Problem arbeitet wie folgt (o. B. d. A hat der Eingabegraph nur eine Zusammenhangskomponente, sonst färben wir jede Zusammenhangskomponente einzeln, wie beschrieben): Beginne bei einem beliebigen Knoten v und färbe diesen mit Farbe 1. Führe den nachfolgenden Schritt so lange wie möglich aus: Solange es noch einen ungefärbten Knoten w gibt, dessen bereits gefärbte Nachbarn alle die gleiche Farbe haben, gib w die dazu komplementäre Farbe.

Wenn alle Knoten gefärbt sind, gib Ja aus, ansonsten Nein.

Korrektheit: Im Algorithmus werden niemals zwei benachbarte Knoten gleich gefärbt. Also handelt es sich tatsächlich um eine Ja-Instanz, wenn der Algorithmus das ausgibt.

Falls der Algorithmus Nein ausgibt, lässt sich der Graph tatsächlich nicht mit zwei Farben färben: Es ist korrekt, im ersten Schritt v die Farbe 1 zu geben, denn wenn eine Zweifärbung existiert, die v die Farbe 2 zuweist, existiert durch Vertauschen der Farben auch eine, die v die Farbe 1 gibt. In jedem weiteren Färbeschritt wird einem Knoten seine einzige zulässige Farbe zugewiesen: Die Komplementäre zu einem seiner Nachbarn. Wenn der Algorithmus also Nein ausgibt, dann gibt es mindestens einen Knoten v mit ungefärbtem Nachbarn w (sonst wären alle Knoten gefärbt), der wiederum einen komplementär zu v gefärbten Nachbarn v' hat (sonst könnte w ja noch gefärbt werden). D. h. aber, dass w tatsächlich nicht zulässig gefärbt werden kann. Damit handelt es sich auch wirklich um eine Nein-Instanz.

Laufzeit: Das Schreiben und Lesen einer Knotenfarbe kann in $O(|V|^c)$ Schritten (für ein konstantes c) realisiert werden.

Nach dem Färben des ersten Knotens wird die nachfolgende Schleife maximal $|V|$ mal ausgeführt, da jedes mal ein Knoten gefärbt wird. Pro Durchlauf kann in $O(|V|^2 \cdot |V|^c)$ Schritten ein "passender" Knoten w gefunden und gefärbt werden. Die gesamte Zeitkomplexität liegt also in $O(|V|^c + |V| \cdot |V|^2 \cdot |V|^c) = O(|V|^{3+c})$.

Aufgabe 4. Probleme in NP

Zeigen Sie, dass die folgenden Probleme in NP liegen.

1. **Vertex Cover**
   Eingabe: Ein ungerichteter Graph $G = (V, E)$ und $k \in \mathbb{N}$.
   Frage: Gibt es eine Knotenmenge $X \subseteq V$ der Größe höchstens k, sodass für jede Kante $e \in E$ gilt, dass $e \cap X \neq \emptyset$?

2. **Cycle Coloring**
   Eingabe: Ein ungerichteter Graph $G = (V, E)$.
   Frage: Gibt es eine totale Funktion $f : V \to \{1, 2, 3\}$, sodass
   - für jede Kante $\{u, v\} \in E$ gilt: $f(u) \neq f(v)$ und
   - jeder Kreis in G enthält drei Knoten v_1, v_2 und v_3 mit $f(v_1) = 1$, $f(v_2) = 2$ und $f(v_3) = 3$?

——————Lösungsskizze——————

1. Eine NTM für Vertex Cover arbeitet wie folgt:

   Falls $k \geq |V|$, akzeptiere. Sonst "rate" nichtdeterministisch eine Menge $X \subseteq V$ von k Knoten und überprüfe ob für jede Kante mindestens ein Endpunkt in X ist. Falls ja, akzeptiert die NTM.

   Die Laufzeit ist klar polynomiell: Es werden höchstens $|V|$ Knoten nichtdeterministisch geraten ($O(\text{poly}(|V|))$ Schritte) und für jede Kante kann in $O(\text{poly}(|V|))$ überprüft werden, ob ein Endpunkt ausgewählt wurde.

2. Eine NTM für Cycle Coloring arbeitet wie folgt:

   (a) Erzeuge durch $O(|V|)$ nichtdeterministische Schritte eine Färbung $f \colon V \to \{1, 2, 3\}$.

   (b) Überprüfe für jede Kante, ob die Endpunkte unterschiedlich gefärbt sind. Halte und lehne ab, falls dies nicht der Fall ist. ($O(|E| \cdot \text{poly}(|V|))$ Schritte)

   (c) Überprüfe für jedes $i \in \{1, 2, 3\}$, ob der Graph $G_i := (\{v \in V \mid f(v) \neq i\}, \{\{v, u\} \in E \mid f(v) \neq i \neq f(u)\})$ kreisfrei ist. Halte und lehne ab, falls dies für ein $i \in \{1, 2, 3\}$ nicht der Fall ist.

   (3 Graphen generieren und mittels Breitensuche Kreise suchen $\rightsquigarrow O(\text{poly}(|V|))$ Schritte)

   (d) Akzeptiere.

   Korrektheit: Es ist noch zu zeigen, dass die oben beschriebene NTM genau dann akzeptiert, wenn der gegebene Graph $G = (V, E)$ eine Ja-Instanz von Cycle Coloring ist.

   Angenommen, die NTM akzeptiert. Dann entspricht die in (a) geratene Färbung f einer in Cycle Coloring beschriebenen Funktion, denn es gilt: Es gibt keine Kante in E, bei der beide Endpunkte die gleiche Farbe haben, andernfalls halten wir in (b). Es gibt keinen Kreis in G, dessen Knoten nur zwei der drei Farben benutzen, andernfalls gäbe es einen Kreis in G_1, G_2 oder G_3 und wir halten in (c).

   Angenommen es gibt eine gültige Färbung f für G. Dann kann die NTM diese Färbung in (a) wählen und somit nicht in (b) oder (c) ablehnen. Also akzeptiert die NTM, falls sie auf eine Ja-Instanz angesetzt wird.