Institut für Theoretische Informatik
Leibniz Universität Hannover

1. Übungsblock

Anleitung zur Bearbeitung dieser Aufgaben

Dieser Übungsblock unterteilt sich in Aufgabenabschnitte, die Sie nach dem Bearbeiten verschiedener Vorbereitungsblätter lösen können sollten.

Kategorie. Die Aufgaben sind dabei in verschiedene Kategorien eingeteilt, welche Sie an folgende Symbolen erkennen.

E steht dabei für eine Einstiegsaufgabe. Aufgaben dieser Art sollten Sie direkt nach der Bearbeitung der Vorbereitungsblätter lösen können.

K bezeichnet Aufgaben auf Klausurniveau. So eine Aufgabe muss nicht zwingend so vorkommen, aber der Schwierigkeitsgrad ist entsprechend. Bei diesen wird jeweils die Punktzahl angegeben. Diese Punktzahl bezieht sich auf eine 90-minütige Klausur mit 64 Punkten (also rechnerisch eine Minute und 24 Sekunden pro Punkt).

⋆ bezeichnet schwierige Aufgaben. Diese Aufgaben gehen noch über das Klausurniveau hinaus.

Zu den mit † gekennzeichneten Aufgaben finden sich Lösunge im Laufe des Semsters veröffentlichten Dokument 'exemplarischer Musterlösungen'.

Teaser. In den Aufgabenkästen befinden sich Links zu Teaservideos. Diese sind über den QR-Code oder in dem digital hinterlegten Link aufrufbar. Die Teaservideos sollen zeigen, wie man eine Aufgabe diesen Aufgabentyps idealerweise löst. Wichtig: Es handelt sich hierbei nicht (zwingend) um die Lösung zu exakt dieser Aufgabe.

Hinweise. Wenn Sie bei der Bearbeitung einer Aufgabe oder einer Teilaufgabe nicht weiterkommen, sollen Ihnen die Hiweise weiterhelfen. Diese können Sie über drei Wege öffnen:

- Sie scannen den QR-Code mit Ihrem Handy
- Sie geben den Hinweiscode (z. B. 02c für Aufgabe 2 c) auf folgender Seite ein: `https://kva.thi.uni-hannover.de/hinweise.php`
- Wenn Sie dieses Dokument digital betrachten, können Sie die QR-Codes auch direkt anklicken. Es ist ein Link zu dem Hinweis hinterlegt

Trennlinien. Verschiedene Aufgabenabschnitte, die zu den einzelnen Vorbereitungsblättern zählen, sind optisch durch rote Trennlinien gekennzeichnet und mit den entsprechenden Vorbereitungsblätternummern markiert.

Zusatzaufgaben. Aufgaben nach diesem Trenner werden zur zusätzliche Übung zu diesem VB angeboten. Sie können freiwillig bearbeit werden.

Klausuraufgaben. Am Ende jedes Aufgabenblocks sind für sie als Angebot weitere Aufgaben mit einem Klausurniveau aufgeführt. Hier sind sehr bewusst keine Lösungen und Hinweise aufgeführt. Sie geben Ihnen die Möglichkeit echte Klausuraufgaben (aus vergangenen Klausuren) zur Vorbereitung der Klausur zu bearbeiten.

Beginn Vorbereitungsblatt 0

Zum Lösen der nachfolgenden Aufgaben, sollten Sie zunächst das Vorbereitungsblatt 0 bearbeitet haben.

Kochrezept für Aufgabentyp O-Notation und o-Notation
Zeigen oder widerlegen Sie: $f(n) \in \mathcal{O}(g(n))$ bzw. $f(n) \in o(g(n))$

Rezept-Video: (2 Minuten) — ***Link:*** `http://go.lu-h.de/kLYBU` ***Titel:*** Rezept O-Notation

Beispiel-Video: (3 Minuten) — ***Link:*** `http://go.lu-h.de/5oMwD` ***Titel:*** Beispielaufgabe 1b

Beispielhafte Aufgabenstellung:
Beweisen oder widerlegen Sie: $n^2 \in \mathcal{O}(n)$.

Rezept O-Notation:
Grundsätzlich gibt es zwei Herangehensweisen für diesen Aufgabentyp, die sich anhand der verwendeten, aber äquivalenten Definition von O unterscheiden:

1. $f(n) \in \mathcal{O}(g(n)) \Leftrightarrow \exists n_0 \in \mathbb{N}\ \exists c > 0\ \forall n \geq n_0 : f(n) \leq cg(n)$.
   Beachten Sie, dass c eine Konstante ist. Das heißt, dass c nicht von n abhängen darf.
2. $f(n) \in \mathcal{O}(g(n)) \Leftrightarrow \limsup\limits_{n\to\infty} \frac{f(n)}{g(n)} < \infty$.

   – In den von uns betrachteten Aufgaben existiert in der Regel dieser Grenzwert. Anstatt des Limes superior können Sie also einfach folgendes bestimmen:

   $$\lim_{n\to\infty} \frac{f(n)}{g(n)}$$

   – Zur Grenzwertbestimmung kann die Regel von **l'Hospital** nützlich sein. Dies ist dann der Fall, wenn $\lim_{n\to\infty} f(x) = 0$ oder $= \infty$ und $\lim_{n\to\infty} g(x) = 0$ oder $= \infty$ gilt. Denn dann gilt:

   $$\text{Falls } \lim_{n\to\infty} \frac{f'(n)}{g'(n)} \text{ existiert, so gilt } \lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}.$$

   Hierbei bezeichnet $f'(n)$ die erste Ableitung von f nach n.

Häufig ist die Argumentation über den Grenzwert die leichtere der beiden Möglichkeiten. Es ergeben sich so die folgenden Strategien:

Fall	**Quantoren**	**Grenzwerte**
$f(n) \in \mathcal{O}(g(n))$	$\exists n_0 \in \mathbb{N}\ \exists c > 0\ \forall n \geq n_0 : f(n) \leq cg(n)$	$\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} < \infty$
$f(n) \notin \mathcal{O}(g(n))$	$\forall n_0 \in \mathbb{N}\ \forall c > 0\ \exists n \geq n_0 : f(n) > cg(n)$	$\lim\limits_{n\to\infty} \frac{f(n)}{g(n)} = \infty$

Rezept o-Notation:

Kochrezept für Aufgabentyp o-Notation
Zeigen oder widerlegen Sie: $f(n) \in o\left(g(n)\right)$

Rezept-Video: (2 Minuten) — ***Link:*** `http://go.lu-h.de/kLYBU` ***Titel:*** Rezept O-Notation

Beispiel-Video: (3 Minuten) — ***Link:*** `http://go.lu-h.de/2BjGL` ***Titel:*** Beispielaufgabe 2c

Beispielhafte Aufgabenstellung:

Beweisen oder widerlegen Sie: $1 + 2 + \ldots + n \in o\left(n^2\right)$.

Per Definition gilt (mit der obigen Annahme, dass der Grenzwert ex.):

$$f(n) \in o\left(g(n)\right) \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Um Aufgaben diesen Typs zu lösen muss also lediglich ein Grenzwert bestimmt werden. Wie bei den Aufgaben zur O-Notation ist auch hier die Regel von **l'Hospital** hilfreich:

$$\text{Falls } \lim_{n\to\infty} \frac{f'(n)}{g'(n)} \text{ existiert, so gilt } \lim_{n\to\infty} \frac{f(n)}{g(n)} = \lim_{n\to\infty} \frac{f'(n)}{g'(n)}.$$

E† **Aufgabe 1**

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

Teaser

a) $2n \in \mathcal{O}(n)$

b) $n^2 \in \mathcal{O}(n)$

c) $\log_2(n) \in \mathcal{O}(\log_k(n))$ für alle festen $k \in \mathbb{N}\backslash\{0,1\}$

d) $n \cdot \log_2(n) \in \mathcal{O}(n^2)$

e) $3^n \in 2^{\mathcal{O}(n)}$

f) $(2^n)^3 \in 2^{\mathcal{O}(n)}$

g) $2^{n^3} \in 2^{\mathcal{O}(n)}$

h) $\mathcal{O}(2^n) = \mathcal{O}(3^n)$

i) $\mathcal{O}(n^2) + \mathcal{O}(n) = \mathcal{O}(n^2)$

Tipp c) 01c

Tipp h) 01h

E† **Aufgabe 2**

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

a) $n \in o(2n)$

b) $2n \in o(n^2)$

c) $1 \in o(n)$

d) $n^k \in o(2^n)$ für alle festen $k \in \mathbb{N}$

e) $1 + 2 + \cdots + n \in o(n^2)$

Tipp d) 02d

Tipp d) 02d-1

Tipp e) 02e

Zusatzaufgaben

K **Aufgabe 3** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

a) $n^2 \in \mathcal{O}(n \cdot \log(n))$

b) $n! \in \mathcal{O}(2^n)$

c) $\mathcal{O}(2^{2n}) = \mathcal{O}(2^n)$

d) $\mathcal{O}(n) - \mathcal{O}(n) = \mathcal{O}(0)$

Teaser

Tipp 03

Tipp 03-1

Tipp b) 03b

Tipp b) 03b-1

Tipp c) 03c

Tipp d) 03d

Tipp d) 03d-1

K **Aufgabe 4** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

a) $2^n \in o\,(3^n)$

b) $\log_2(n) \in o\,(n)$

c) $n^2 \in o\,(\log_2(n))$

d) $o\,(g(n)) \subseteq \mathcal{O}(g(n))$ für alle $g : \mathbb{N} \to \mathbb{N}$

Tipp b) 04b

Tipp b) 04b-1

Tipp d) 04d

Tipp d) 04d-1

Tipp d) 04d-2

Ende Vorbereitungsblatt 0

Beginn Vorbereitungsblatt 1

Zum Lösen der nachfolgenden Aufgaben, sollten Sie zunächst das Vorbereitungsblatt 1 bearbeitet haben.

E **Aufgabe 5**

Es sei $f : \mathbb{N} \to \mathbb{N}$.

a) Zeigen Sie: Gilt $A \in \mathsf{TIME}(f(n))$, so gilt auch $\overline{A} \in \mathsf{TIME}(f(n))$.

b) Zeigen Sie: Gilt $A \in \mathsf{SPACE}(f(n))$, so gilt auch $\overline{A} \in \mathsf{SPACE}(f(n))$.

c) Sind die Erkenntnisse übertragbar auf die Komplexitätsklassen NTIME bzw. NSPACE? Begründen Sie ihre Antwort.

Tipp c) 05c

Kochrezept für Aufgabentyp 1
Zeigen Sie, dass ein Problem in $\mathsf{SPACE}(s(n))$/$\mathsf{TIME}(t(n))$/$\mathsf{NSPACE}(s(n))$/$\mathsf{NTIME}(t(n))$ liegt

Rezept-Video: (2 Minuten) — ***Link:*** `http://go.lu-h.de/wwt6R` ***Titel:*** Rezept Mitgliedschaftsresultat für NTIME, TIME, SPACE, NSPACE

Beispiel-Video: (8 Minuten) — ***Link:*** `http://go.lu-h.de/oR8m7` ***Titel:*** Beispielaufgabe 3

Beispielhafte Aufgabenstellung:
Im folgenden betrachten wir die aus der Vorlesung bekannte Sprache

$$A \coloneqq \{0^i 1^i \mid i \geq 0\}.$$

Zeigen Sie, dass $A \in \mathsf{TIME}(n \cdot \log(n))$ gilt. Beschreiben Sie hierzu die Funktionsweise der Turingmaschine vollständig und begründen Sie den Speicherbedarf der Maschine.

Rezept:
In diesem Aufgabentyp müssen Algorithmen gefunden werden, die auf ihren Platz- bzw. Zeitbedarf untersucht werden müssen.

Wichtig ist, dass bei der Betrachtung immer eindeutig auf die Eingabelänge referenziert wird, um die Gültigkeit eindeutig festzuhalten. Außerdem sollte man die Laufzeit von Teilen des Algorithmus erläutert, die nicht eindeutig zu erkennen sind.

Das Vorgehen sollte dabei grundsätzlich wie folgt sein:

1. Sich das beschriebene Problem mittels Beispielen verdeutlichen.
   - In der Regel ist es gut, einmal Positiv- und Negativinstanzen durchzusprobieren und sich mindestens ein Beispiel dazu zu notieren.
   - Dann sollte man sich überlegen, welche Randfälle es gibt.
     - Graphenbeispiele: ein Graph, der nur aus isolierten Knoten besteht oder ein vollständiger Graph, ...
     - Aussagenlogische Formeln: Formel, die nur aus einem Literal besteht,...
2. Algorithmus im Pseudo-Code aufschreiben und mit Zeilennummern versehen.
3. Laufzeiten je Zeile ergänzen/abschätzen und erläutern.
   - Referenz für die Laufzeitberechnung ist immer die Eingabe bzw. ein Teil der Eingabe (z. B. $|G|$, $|V|$, $|E|$, n oder konstant)
   - Bei bekannten Schritten kann man die Laufzeit hinschreiben und auf die Übung verweisen
   - Bei den entscheidenden Schritten sollte man im Detail erläutern, wie sich die Laufzeit zusammensetzt und notfalls eine Erläuterung dazu schreiben.
4. Gesamtlaufzeit zusammenrechnen und folgern, dass sie in der beschriebenen Vorgabe liegt, also: $z.B.$ Problem $\in \mathsf{TIME}(t(n))$.
5. 1-2 Sätze zur Korrektheit aufschreiben. „Warum funktioniert der Algorithmus?“.

Machen Sie sich aber bewusst, dass ihre Betrachtung immer jeweils nur auf den Platzbedarf oder den Zeitbedarf fokussiert sind. Für die Speicherbetrachtung ist es unerheblich, wie oft eine Bandzelle besucht wird.

E[†] Aufgabe 6

Wir betrachten die aus der Vorlesung bekannte Sprache

$$A := \{0^i 1^i \mid i \geq 0\}$$

sowie den folgenden Algorithmus, der die Sprache A entscheidet:

Eingabe: $w \in \{0,1\}^*$
if *w ist das leere Wort* **then** akzeptiere;
if *w hat nicht die Form* 0^+1^+ **then** verwerfe;
while *es ist mehr als eine Null vorhanden* **do**
    if *Summe nun vorhandener Nullen und Einsen ist ungerade* **then** verwerfe;
    Lösche jede zweite Null und jede zweite Eins;
end
if *es gibt noch jeweils genau eine Null und eine Eins* **then**
    akzeptiere
else
    verwerfe
end

a) Begründen Sie, dass der angegebene Algorithmus korrekt ist.

b) Geben Sie die Laufzeit des Algorithmus an und begründen Sie, dass $A \in \mathsf{TIME}(n \cdot \log n)$ gilt.

Teaser

Tipp a)
06a

E Aufgabe 7

Für eine Zahl $k \in \mathbb{N}$ bezeichne $\mathrm{bin}(k)$ die Binärdarstellung von k ohne führende Nullen.
Zeigen Sie, dass es eine Turingmaschine gibt, die bei Eingabe 1^k (d.h. unär kodierte Darstellung von k) den Wert $\mathrm{bin}(k)$ berechnet und Platzbedarf $\mathcal{O}(\log n)$ hat, wobei n die Eingabelänge ist.

Teaser

E **Aufgabe 8**

Es seien $a, b, c \in \mathbb{N}$. Die Sprache SUM ist definiert als

$$\mathsf{SUM} = \{\langle \text{bin}(a), \text{bin}(b), \text{bin}(c)\rangle \mid a + b = c\}.$$

Zeigen Sie, dass $\mathsf{SUM} \in \mathsf{TIME}(n)$ gilt, wobei n die Eingabelänge ist.

Teaser

Zusatzaufgaben

E **Aufgabe 9**

Geben Sie je eine Sprache aus der Klasse $\mathsf{TIME}(1)$ und $\mathsf{SPACE}(1)$ an.

K† **Aufgabe 10**

Wir betrachten erneut die aus Aufgabe 6 bekannte Sprache

$$A := \{0^i 1^i \mid i \geq 0\}.$$

Zeigen Sie, dass $A \in \mathsf{SPACE}(\log n)$ gilt.

Teaser

⋆ **Aufgabe 11**

Gegeben sei eine aufsteigend sortierte Sequenz von Binärzahlen

$$\text{bin}(i_1) \diamond \text{bin}(i_2) \diamond \cdots \diamond \text{bin}(i_m),$$

wobei $\diamond$ ein Trennsymbol ist, $i_\mu \in \mathbb{N}$ für $1 \leq \mu \leq m$ und $\text{bin}(k)$ für $k \in \mathbb{N}$ die Binärdarstellung von k ohne führende Nullen ist. Zeigen Sie, dass es eine Turingmaschine gibt, welche überprüft, dass $i_{\mu+1} = i_\mu + 1$ ist und Speicherbedarf $\mathcal{O}(\log(\log(i_m)))$ hat.

Tipp
11

Ende Vorbereitungsblatt 1

Beginn Vorbereitungsblatt 2

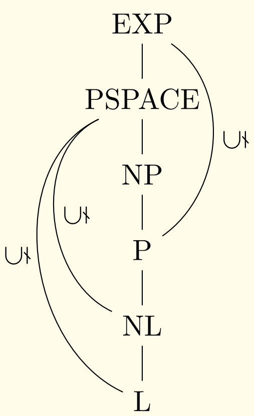
Zum Lösen der nachfolgenden Aufgaben, sollten Sie zunächst das Vorbereitungsblatt 2 bearbeitet haben.

Kochrezept für Aufgabentyp 2
Beweisen oder widerlegen Sie für zwei Komplexitätsklassen X, Y:
$X = Y$, $X \subseteq Y$ bzw. $X \subsetneq Y$.

Rezept-Video: (4 Minuten)		***Link:***	`http://go.lu-h.de/8CISq`
		Titel:	Rezept Inklusion/Gleichheit von Komplexitätsklassen
Beispiel-Video: (5 Minuten)		***Link:***	`http://go.lu-h.de/JzE3K`
		Titel:	Beispielaufgabe 6

Beispielhafte Aufgabenstellung:
Beweisen oder widerlegen Sie $\mathsf{NTIME}(n) \subsetneq \mathsf{PSPACE}$.

Spezielle Komplexitätklassen, Sätze der Vorlesung und deren Zusammenhang:



$\text{EXP} = \mathsf{TIME}(2^{n^{O(1)}}) = \bigcup_{c\in\mathbb{N}} \mathsf{TIME}(2^{n^c})$

$\text{PSPACE} = \mathsf{SPACE}(n^{O(1)}) = \bigcup_{c\in\mathbb{N}} \mathsf{SPACE}(n^c)$

$\text{NP} = \mathsf{NTIME}(n^{O(1)}) = \bigcup_{c\in\mathbb{N}} \mathsf{NTIME}(n^c)$

$\text{P} = \mathsf{TIME}(n^{O(1)}) = \bigcup_{c\in\mathbb{N}} \mathsf{TIME}(n^c)$

$\text{NL} = \mathsf{NSPACE}(\log(n))$

$\text{L} = \mathsf{SPACE}(\log(n))$

Für die Bearbeitung der Aufgaben sind die Zusammenhänge zwischen den Komplexitätsklassen wichtig. Die folgende Tabelle gibt einen Überblick über die Resultate aus der Vorlesung und die zu erfüllenden Voraussetzungen:

	Voraussetzung	**Aussage**
Korollar A	keine	$\mathsf{TIME}(t(n)) \subseteq \mathsf{NTIME}(t(n))$
Korollar B	keine	$\mathsf{SPACE}(s(n)) \subseteq \mathsf{NSPACE}(s(n))$
Satz 3	$t(n) \geq n$	$\mathsf{NTIME}(t(n)) \subseteq \mathsf{TIME}(2^{\mathcal{O}(t(n))})$
Satz 4	$t(n) \geq n$	$\mathsf{NTIME}(t(n)) \subseteq \mathsf{SPACE}(t(n))$
Korollar C	$t(n) \geq n$	$\mathsf{TIME}(t(n)) \subseteq \mathsf{SPACE}(t(n))$
Satz 5	$s(n) \geq \log(n)$	$\mathsf{SPACE}(s(n)) \subseteq \mathsf{TIME}(2^{\mathcal{O}(s(n))})$
Satz 7	$s(n) \geq \log(n)$ raumkonstruierbar	$\mathsf{NSPACE}(s(n)) \subseteq \mathsf{TIME}(2^{\mathcal{O}(s(n))})$
Satz 8 (Savitch)	$s(n) \geq \log(n)$ raumkonstruierbar	$\mathsf{NSPACE}(s(n)) \subseteq \mathsf{SPACE}((s(n))^2)$

Achtung: Mit den folgenden Sätzen können nur echte Teilmengenbeziehungen zwischen *einzelnen deterministischen* Komplexitätsklassen gezeigt werden, aber nicht zwischen deren Vereinigungen oder zwischen Mengen von Funktionen.

Platzhierarchiesatz (PHS):

Sei $s_1 \in o(s_2)$ und s_2 raumkonstruierbar. Dann ist $\mathsf{SPACE}(s_1) \subsetneq \mathsf{SPACE}(s_2)$.

Rezept:

Schauen Sie sich als ersten Schritt die jeweilige Aufgabe an und überlegen Sie was der erfolgversprechende Ansatz sein könnte (Beweisen oder Widerlegen).

Hier muss anschießend je nach Fall weitergearbeitet werden

X = Y : Bei der Gleichheit müssen beide Inklusionsrichtungen gezeigt werden, also $X \subseteq Y$ und $Y \subseteq X$.

X ⊆ Y : Mithilfe obiger Sätze lässt sich eine Inklusionskette von X nach Y konstruieren. Manchmal ist es dabei einfacher, X zunächst auf den selben Typ (SPACE, NSPACE, TIME, NTIME) wie Y zu bringen. Anschließend kann man die Teilmengenbeziehungen der so erhaltenen Klassen z. B. durch O-Abschätzung bestimmen. Am Beispiel: $\mathsf{SPACE}(f(n)) \subseteq \mathsf{SPACE}(g(n))$, falls $f(n) \in O(g(n))$. Alternativ kann ggf. auch die echte Inklusion (siehe nächster Punkt) gezeigt werden.

X ⊊ Y : Mit Blick auf obige Sätze sieht man, dass man hierfür den Platzhierarchiesatz anwenden muss. Es ist oft sinnvoll, zu zeigen, dass die Klasse X in einer anderen Klasse enthalten ist und von dort eine echte Inklusion nach Y zu etwas zwischen X und Y zu finden. Hierbei verwendet man obige Sätze.

Zusammenfassend gibt es für den Beweis oder Widerspruch die folgenden Strategien:

Fall	**Beweis**	**Widerspruch**
X = Y	$X \subseteq Y$ und $Y \subseteq X$	$X \subsetneq Y$ oder $Y \subsetneq X$ (über PHS)
$X \subseteq Y$	$X \subseteq Y$	$Y \subsetneq X$ (über PHS)
$X \subsetneq Y$	$X \subsetneq Y$ (über PHS)	$Y \subsetneq X$ (über PHS) oder X = Y

Für alle drei Fälle gilt: Zur besseren Visualisierung des Vorgehens sollte eine Inklusionskette aufgeschrieben werden, die wie folgt aussehen könnte:

$$X(f(n)) \underset{(1.)}{\subseteq} X'(g(n)) \underset{(2.)}{\subsetneq} Y'(h(n)) \underset{(3.)}{\subseteq} Y(j(n))$$

Wichtig ist, dass die einzelnen Inklusionen begründet und die jeweiligen Vorrausetzungen korrekt verwendet werden.

- Satz aus Vorlesung: Satz notieren und Voraussetzung (und damit Anwendbarkeit) prüfen.
- Platzhierarchiesatz
  - Raumkonstruierbarkeit von s_2 prüfen
  - Zeige $s_1 \in o(s_2)$ über Grenzwertbetrachtung
  - Ergebnissatz: Nach Platzhierarchiesatz gilt: $\mathsf{SPACE}(s_1(n)) \subsetneq \mathsf{SPACE}(s_2(n))$
- Begründung über Auswahl einer Teilmenge aus einer Vereinigung
  (z. B. $\mathsf{TIME}(n^2) \subseteq \bigcup_{k \in \mathbb{N}} \mathsf{TIME}(n^k) = P$)
- Zur Überprüfung der Raumkonstruierbarkeit gilt:
  - Logarithmen, Polynome und Konstanten sind raumkonstruierbare Funktionen.
  - Aus Übung bekannt: Summen, Produkte und Potenzen zweier raumkostruierbarer Funktionen sind wieder raumkonstruierbar.

– Zerlegen Sie die Funktion also in Teilfunktionen, von denen Sie wissen, dass sie raumkonstruierbar sind und begründen Sie so die Raumkonstruierbarkeit der Gesamtfunktion.

E **Aufgabe 12**

Es sollen die folgenden Aussagen gezeigt werden:

Teaser

- $\mathsf{TIME}(\log(n)) \subseteq \mathsf{SPACE}(n \cdot \log(n))$
- $\mathsf{SPACE}(\log(n)) \subseteq \mathsf{P}$
- $\mathsf{NTIME}(\log(n)) \subseteq \mathsf{SPACE}(n)$

Überprüfen Sie die folgenden Inklusionsbeziehungen mithilfe von Sätzen aus der Vorlesung und korrigieren Sie ggf. Fehler. Prüfen Sie insbesondere, ob etwaige Voraussetzungen für die Sätze erfüllt sind.

a) $\mathsf{TIME}(\log(n)) \subseteq \mathsf{TIME}(n \cdot \log(n)) \subseteq \mathsf{SPACE}(n \cdot \log(n))$

b) $\mathsf{SPACE}(\log(n)) \subseteq \mathsf{TIME}(2^{\log(n)}) \subseteq \mathsf{TIME}(n) \subseteq \mathsf{TIME}(n^{\mathcal{O}(1)}) \overset{\text{def}}{=} \mathsf{P}$

c) $\mathsf{NTIME}(\log(n)) \subseteq \mathsf{SPACE}(\log(n)) \subseteq \mathsf{SPACE}(n)$

K **Aufgabe 13** (12 Punkte)

Beweisen Sie die folgenden Aussagen.

Teaser

a) $\mathsf{TIME}(2^n) = \mathsf{TIME}(2^{n+1})$

b) $\mathsf{NTIME}(n) \subseteq \bigcup_{k\in\mathbb{N}} \mathsf{SPACE}(n^k)$

Tipp a) 13a

Tipp a) 13a-1

Tipp b) 13b

Tipp b) 13b-1

K† **Aufgabe 14** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

a) $\mathsf{TIME}(n \cdot \log(n)) \subseteq \mathsf{TIME}(n^2)$

b) $\mathsf{NTIME}(n \cdot \log(n)) \subseteq \mathsf{SPACE}(n^4)$

Tipp a) 14a

Tipp a) 14a-1

Tipp b) 14b

Zusatzaufgaben

E **Aufgabe 15**

Zeigen Sie für die Sprache

$$L = \{w\$w^R \mid w \in \{0,1\}^*\},$$

dass $L \in \mathsf{SPACE}(n)$ ist, wobei n die Eingabelänge ist und und w^R steht für w rückwärts.

Teaser

Tipp
15

K **Aufgabe 16** (12 Punkte)

Zeigen Sie für die Sprache

$$L = \{\, ww \mid w \in \{0,1\}^* \,\},$$

dass $L \in \mathsf{NTIME}(n)$ ist, wobei n die Eingabelänge ist.

Teaser

Tipp
16

K **Aufgabe 17** (12 Punkte)

Zeigen Sie für die Sprache

$$L = \{x \in \{0,1,\ldots,9\}^+ \mid x \text{ ist keine Primzahl in Dezimaldarstellung}\},$$

dass $L \in \mathsf{SPACE}(n)$ ist, wobei n die Eingabelänge ist.

Teaser

Tipp
17

Ende Vorbereitungsblatt 2

- -

Beginn Vorbereitungsblatt 3

Zum Lösen der nachfolgenden Aufgaben, sollten Sie zunächst das Vorbereitungsblatt 3 bearbeitet haben.

Kochrezept für Aufgabentyp
Raumkonstruierbarkeit

Rezept-Video: (1 Minuten) — ***Link:*** `http://go.lu-h.de/D7R3d` ***Titel:*** Rezept Raumkonstruierbarkeit

Zur Raumkonstruierbarkeit:

Eine Funktion $f : \mathbb{N} \to \mathbb{N}$ heißt **raumkonstruierbar**, falls es eine deterministische Einband-Turingmaschine gibt, die bei Eingabe eines Wortes x einen Speicherbedarf von **genau** $f(|x|)$ hat. Wie sonst auch unterscheiden wir wieder zwischen Eingabe- und Arbeitsband. Um die Raumkonstruierbarkeit einer Funktion nachzuweisen, muss man also eine Turingmaschine mit einem Arbeitsband angeben, die bei Eingabe x **genau** $f(|x|)$ Bandzellen des Arbeitsbandes **mindestens einmal besucht**.

Um Raumkonstruierbarkeit für eine Funktion $f(n)$ zu zeigen, kann man z.B. eine Turingmaschine konstruieren, die genau $f(n)$ Felder markiert.

Beispiele

Beispielsweise ist die Funktion $f(n) = c$ für ein beliebiges $c > 0$ raumkonstruierbar, da die DTM, die unabhängig von der Eingabe genau $c - 1$ mal auf dem Arbeitsband nach rechts geht und dann hält, genau c Arbeitsbandzellen besucht.

Auch die Funktion $f(n) = n$ ist raumkonstruierbar. Eine entsprechende DTM geht auf dem Eingabeband einen Schritt nach rechts und läuft anschließend für jedes weitere Zeichen der Eingabe auf dem Arbeitsband einen Schritt nach rechts.

Nach Satz 6 der Vorlesung gibt es keine nichtkonstante Funktion in $o(\log(n))$, die raumkonstruierbar ist. So ist zum Beispiel $f(n) = \log(\log(n))$ **nicht** raumkonstruierbar.

K† **Aufgabe 18** (12 Punkte)

Zeigen Sie, dass Polynome raumkonstruierbar sind, indem Sie die folgenden Aussagen beweisen. Im Folgenden seien f, g raumkonstruierbare Funktionen und $k \in \mathbb{N}$.

a) $f + g$ b) $f \cdot g$ c) f^g d) $h(n) = n^k$

sind raumkonstruierbar.

Tipp 18 — **Tipp** 18-1 — **Tipp** 18-2 — **Tipp** 18-3

E **Aufgabe 19**

Definieren Sie drei Mengen A, B, C, sodass $A \subsetneq C$ und $B \subsetneq C$, aber $A \cup B = C$ gilt.

K† **Aufgabe 20** (24 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

a) $\mathsf{SPACE}(2^n) \subsetneq \mathsf{SPACE}(2^{2n})$

b) $\mathsf{NTIME}(n) \subsetneq \mathsf{PSPACE}$

c) $\mathsf{PSPACE} \subsetneq \mathsf{SPACE}(2^n)$

d) $\mathsf{NSPACE}(2^{3n}) = \mathsf{NSPACE}(2^n)$

Tipp 20 — **Tipp b)** 20b — **Tipp c)** 20c — **Tipp c)** 20c-1 — **Tipp d)** 20d

Zusatzaufgaben

K **Aufgabe 21** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

Teaser

(a) $\mathsf{NSPACE}(n \cdot \log n) \subsetneq \mathsf{SPACE}(n^2 \cdot (\log n)^3)$

(b) $\mathsf{NP} \subsetneq \mathsf{SPACE}(2^{n^2})$

Tipp 21 — **Tipp (a)** 21a — **Tipp (b)** 21b

Ende Vorbereitungsblatt 3

Beginn Vorbereitungsblatt 4

Zum Lösen der nachfolgenden Aufgaben, sollten Sie zunächst das Vorbereitungsblatt 4 bearbeitet haben.

Kochrezept für Aufgabentyp 3

Zeigen Sie, dass ein Problem in der Klasse P liegt.

Rezept-Video: (2 Minuten)
Link: `http://go.lu-h.de/wwt6R`
Titel: Rezept Mitgliedschaftsresultat für NTIME, TIME, SPACE, NSPACE

Beispiel-Video: (10 Minuten)
Link: `http://go.lu-h.de/djJZI`
Titel: Beispielaufgabe 5

Beispielhafte Aufgabenstellung: Zeigen Sie, dass die Sprache

$$\text{CONNECTED} := \{\langle G, k\rangle \mid G \text{ ist ein ungerichteter Graph mit } \leq k \text{ ZHK}\}$$

zur Klasse P gehört.

Rezept:

Eine Umformulierung der Musteraufgabenstellung ist:

Schreibe einen Algorithmus, der das Problem entscheidet und zeige, dass dieser in polynomieller Zeit läuft.

Das Vorgehen ist hierbei analog zu Aufgabentyp 1 (siehe oben) und folgt grundsätzlich dem gleichen Schema:

1. Sich das beschriebene Problem mittels Beispielen verdeutlichen.
2. Algorithmus im Pseudo-Code aufschreiben und mit Zeilennummern versehen.
3. Laufzeiten je Zeile ergänzen/abschätzen und erläutern.
4. Gesamtlaufzeit zusammenrechnen und folgern, dass es in polynomieller Zeit liegt, also: Problem $\in \mathsf{P}$.
5. 1-2 Sätze zur Korrektheit aufschreiben. „Warum funktioniert der Algorithmus?“.

E† **Aufgabe 22**

Teaser

Ein *ungerichteter Graph* $G = (V, E)$ besteht aus einer *Knotenmenge* V sowie einer Kantenmenge $E \subseteq V \times V$. Eine *Zusammenhangskomponente* (ZHK) eines ungerichteten Graphen $G = (V, E)$ ist eine inklusionsmaximale Menge von Knoten $U \subseteq V$ mit der Eigenschaft, dass es zwischen je zwei Knoten aus U einen Pfad in G gibt.

Zeigen Sie, dass die Sprache

$$\text{CONNECTED} := \{\langle G, k\rangle \mid G \text{ ist ein ungerichteter Graph mit } \leq k \text{ ZHK}\}$$

zur Klasse P gehört.

Tipp 22 | Tipp 22-1 | Tipp 22-2

E† **Aufgabe 23**

Zwei ungerichtete Graphen $G = (V, E)$ und $H = (V', E')$ sind zueinander isomorph, $G \cong H$, falls es eine bijektive Abbildung $\pi\colon V \to V'$ gibt, so dass für alle $u, v \in V$ gilt

$$\{u, v\} \in E \Leftrightarrow \{\pi(u), \pi(v)\} \in E'.$$

Teaser

So eine Funktion π wird als Isomorphismus zwischen G und H bezeichnet. Anschaulich gesprochen kann man die Knoten von G so umbenennen, dass H entsteht.

$\mathrm{GI} := \{\langle G, H\rangle \mid G \text{ und } H \text{ sind zueinander isomorphe ungerichtete Graphen}\}$.

Wir möchten zeigen, dass $\mathrm{GI} \in \mathsf{NP}$ gilt.

a) Welche der folgenden Möglichkeiten eignen sich als Zertifikate für die polynomielle Überprüfbarkeit von GI?
   - H
   - $f(G, H) = \begin{cases} 1, & \langle G, H\rangle \in \mathrm{GI} \\ 0, & \text{sonst} \end{cases}$
   - Bijektion π von G auf H

b) Geben Sie einen polynomiellen Überprüfungsalgorithmus für GI mithilfe des in der letzten Teilaufgabe gewählten Zertifikats an.

c) Zeigen Sie die Korrektheit Ihres Algorithmus und bestimmen Sie die Laufzeit.

Tipp 23 **Tipp** 23-1 **Tipp** 23-2

E **Aufgabe 24**

Es sei

Teaser

$$\mathrm{EVAL} := \left\{ \langle \varphi, I\rangle \;\middle|\; \begin{array}{l} \varphi \text{ ist eine aussagenlogische Formel} \\ \text{über den Konnektoren } \{\wedge, \vee, \neg\} \\ \text{und } I \text{ ist eine erfüllende Belegung für } \varphi \end{array} \right\}.$$

Zeigen Sie, dass $\mathrm{EVAL} \in \mathsf{P}$ gilt.

Tipp 24

K† **Aufgabe 25** (4 Punkte)

Es sei

Teaser

$$\mathsf{DOUBLESAT} := \left\{ \langle\varphi\rangle \;\middle|\; \begin{array}{l} \varphi \text{ ist eine aussagenlogische Formel, die} \\ \text{mindestens zwei erfüllende Belegungen besitzt} \end{array} \right\}.$$

Zeigen Sie, dass $\mathsf{DOUBLESAT} \in \mathsf{NP}$ ist.

Tipp
25

Zusatzaufgaben

⋆ **Aufgabe 26**

Es sei

Teaser

$$\text{NTMACC} := \left\{ \langle M, x\rangle \circ 1^t \;\middle|\; \begin{array}{l} t \in \mathbb{N} \text{ und } M \text{ ist eine NTM, die die Eingabe } x \\ \text{in höchstens } t \text{ Schritten akzeptiert} \end{array} \right\}$$

Zeigen Sie, dass NTMACC $\in \mathsf{NP}$ gilt.

Tipp
26

K **Aufgabe 27** (4 Punkte)

Teaser

Ein ungerichteter Graph $G = (V, E)$ heißt k-*färbbar* für $k \in \mathbb{N}$, falls seine Knoten mit k zur Verfügung stehenden Farben markiert werden können, sodass keine benachbarten Knoten dieselbe Farbe tragen. Formal definiert bedeutet das: Für $k \in \mathbb{N}$ heißt G genau dann k-färbbar, wenn es eine Abbildung $f\colon V \to \{1, 2, \dots, k\}$ gibt mit $f(u) \neq f(v)$ für alle $(u, v) \in E$.

Es sei

$$\text{COLORABILITY} := \{\langle G, k\rangle \mid k \in \mathbb{N} \text{ und } G \text{ ist ein } k\text{-färbbarer ungerichteter Graph}\}.$$

Zeigen Sie, dass COLORABILITY $\in \mathsf{NP}$ gilt.

Tipp
27

Ende Vorbereitungsblatt 4

Zusätzliche Klausuraufgaben

Im Folgenden sind weitere Aufgaben aus vergangenen Klausuren passend zu diesem Aufgabenblock aufgeführt. Sie sind in die drei Aufgabentypen unterteilt, die dem Teil 0 zugeordnet sind. Diese Aufgaben sollen noch einmal einen Einblick in das zu erwartende Niveau geben. Je Aufgabe ist jeweils die Punktzahl angegeben. Diese Punktzahl bezieht sich auf eine 90-minütige Klausur mit 64 Punkten (also rechnerisch eine Minuten und 24 Sekunden pro Punkt). Für die Aufgaben sind bewusst keine Hinweise und Teaser-Videos gegeben, um eine realistische Klausursituation darzustellen. Es wird empfohlen diese Aufgaben erst zu bearbeiten, wenn Sie vorher Erfahrung mit ähnlichen Aufgaben aus dem vorangegangenen Aufgabenblock gesammelt haben.

Aufgabentyp 1: Zeigen Sie, dass ein Problem in SPACE/TIME/NSPACE/NTIME($t(n)$) liegt.

K **Aufgabe 28** (12 Punkte)

Zeigen Sie, dass CLIQUE $\in$ SPACE(n) gilt.

K **Aufgabe 29** (12 Punkte)

Zeigen Sie, dass 3SAT $\in$ SPACE(n) gilt.

3SAT $:= \{\langle\varphi\rangle | \varphi$ ist eine erfüllbare aussagenlogische Formel in 3KNF$\}$

Aufgabentyp 2: Beweisen oder widerlegen Sie für zwei Komplexitätsklassen X, Y: $X = Y$ bzw. $X \subseteq Y$ bzw. $X \subsetneq Y$.

K **Aufgabe 30** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

1. TIME(n^{42}) $=$ EXP.
2. $\bigcup_{c\in\mathbb{N}}$ SPACE(2^{cn}) $= \bigcup_{c\in\mathbb{N}}$ NSPACE(2^{cn}).

K **Aufgabe 31** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

a) NSPACE($\log n$) $\subsetneq$ SPACE($\sqrt{n^3}$).

b) SPACE($\log n$) $\subsetneq$ TIME(2^{n^2}).

K **Aufgabe 32** (12 Punkte)

Beweisen Sie die folgenden Aussagen.

1. SPACE($\log n$) $\subsetneq$ TIME(2^{n^2}).
2. TIME($n \cdot \log n$) $\subsetneq$ NSPACE(n^4).

K **Aufgabe 33** (12 Punkte)

Beweisen oder widerlegen Sie die folgenden Aussagen.

1. Sei $f(n) = \begin{cases} 1 & \text{, falls } n = 1 \\ n^2 \cdot f(n-1) & \text{, falls } n > 1 \end{cases}$

   Es gilt: $f \in n^{\mathcal{O}(n)}$.

2. Sei $f \geq \log n$ eine Funktion, die raumkonstruierbar ist.
   Es gilt: $\mathsf{NSPACE}(f(n)) = \mathsf{SPACE}((f(n))^4)$.

Aufgabentyp 3: Zeigen Sie, dass ein Problem in der Klasse P liegt.

K **Aufgabe 34** (12 Punkte)

Sei $G = (V, E)$ ein ungerichteter Graph. Die Knoten $u, v, w \in V$ bilden ein *Dreieck in* G, wenn $\{u, v\}, \{v, w\}, \{w, u\} \in E$ gilt. Gibt es kein Dreieck in G, so nennen wir G *dreiecksfrei.*
Zeigen Sie:

$$\text{TRIANGLE-FREE} = \{\langle G\rangle \mid G \text{ ist ein ungerichteter und dreiecksfreier Graph }\} \in \mathsf{P}.$$

K **Aufgabe 35** (12 Punkte)

Ein ungerichteter Graph $G = (V, E)$ ist *bipartit* gdw. es gibt eine Partition der Knotenmenge $V = S \uplus T$ und sowohl für alle $u, v \in S$ gilt $\{u, v\} \notin E$ als auch für alle $u, v \in T$ gilt $\{u, v\} \notin E$. Beweisen Sie: Das Problem

$$\mathsf{BIPARTIT} := \left\{\langle G\rangle \,\middle|\, \begin{array}{l} G = (V, E) \text{ ist ein ungerichteter, bipartiter} \\ \text{Graph} \end{array}\right\}$$

liegt in der Komplexitätsklasse P.

K **Aufgabe 36** (12 Punkte)

Ist $G = (V, E)$ ein gerichteter Graph, dann ist eine Knotenfolge $\kappa = v_0, v_1, \ldots, v_m$ ein *Zyklus von* G, wenn $v_0 = v_m$, $v_i \in V$ für $1 \leq i \leq m$ und $(v_i, v_{i+1}) \in E$ für $1 \leq i < m$. Ein Knoten $v \in V$ ist *Teil eines Zyklus von* G, wenn es einen Zyklus $\kappa = v_0, v_1, \ldots, v_m$ von G gibt mit es gibt ein $0 \leq i \leq m$ mit $v_i = v$.
Beweisen Sie: Das Problem

$$\text{CYCLE} := \left\{\langle G, v\rangle \,\middle|\, \begin{array}{l} G = (V, E) \text{ ist ein gerichteter Graph und} \\ v \in V \text{ ist Teil eines Zyklus von } G. \end{array}\right\}$$

liegt in der Komplexitätsklasse P.

K **Aufgabe 37** (12 Punkte)

Eine Teilmenge von Knoten $V' \subseteq V$ aus einem ungerichteten Graphen $G = (V, E)$ heißt

- *dominierend*, falls für alle Knoten $u \in V \setminus V'$ gilt, dass es ein $v \in V'$ gibt mit $\{u, v\} \in E$.
- *zusammenhängend*, falls es für alle $u, v \in V'$ mit $u \neq v$ einen Pfad zwischen u und v in G gibt.

Zeigen Sie, dass das Problem

$$\mathrm{CDST} := \left\{ \langle T, k \rangle \;\middle|\; \begin{array}{l} T = (V, E) \text{ ist ein ungerichteter Baum, } k \in \mathbb{N} \text{ und es gibt} \\ \text{eine zusammenhängende und dominierende Knotenmenge} \\ D \subseteq V \text{ mit } |D| \leq k \end{array} \right\}.$$

in P ist.