

Tutorium Woche 8

Dominik Bruhn, Tutorium Nr. 9 + 11

16.12.2009

Agenda

- 1 Organisation
- 2 Komplexitätstheorie
 - Wiederholung
 - Aufgaben
- 3 Ende

Weihnachten

- **Abgabe** des nächsten Übungsblattes ist Donnerstag 7. Januar.

Weihnachten

- **Abgabe** des nächsten Übungsblattes ist Donnerstag 7. Januar.
- **Nächste Woche** gibt es Montag ein "Weihnachtsübungsblatt"

Weihnachten

- **Abgabe** des nächsten Übungsblattes ist Donnerstag 7. Januar.
- **Nächste Woche** gibt es Montag ein "Weihnachtsübungsblatt"
- **Nächste Woche** findet am (23. Dezember) **kein** Tutorium statt.

Weihnachten

- **Abgabe** des nächsten Übungsblattes ist Donnerstag 7. Januar.
- **Nächste Woche** gibt es Montag ein "Weihnachtsübungsblatt"
- **Nächste Woche** findet am (23. Dezember) **kein** Tutorium statt.
- Erstes Tutorium im neuen Jahr: 12. Januar

Weihnachten

- **Abgabe** des nächsten Übungsblattes ist Donnerstag 7. Januar.
- **Nächste Woche** gibt es Montag ein "Weihnachtsübungsblatt"
- **Nächste Woche** findet am (23. Dezember) **kein** Tutorium statt.
- Erstes Tutorium im neuen Jahr: 12. Januar
- **48 Punkte** reichen ziemlich sicher für den Schein (es gibt 12 Pflichtübungsblätter)

Übungsblatt

Aufgabe 1.1

Hat niemand richtig gelöst: Korrekt wäre
 $K(x) \leq \log(n) + \log(k) + \log\binom{n}{k} + c$ mit

- n Länge des Wortes
- k Anzahl der 1en
- $\binom{n}{k}$ um die Position zu kodieren

\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:

\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:

$$f(n) \leq c * g(n)$$

\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:
$$f(n) \leq c * g(n)$$

Beispiel

$$x^2 + 2x + 1 = \mathcal{O}(x^2)$$

\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:
 $f(n) \leq c * g(n)$

Beispiel

$$x^2 + 2x + 1 = \mathcal{O}(x^2)$$

Denn, ab $x=10$ gilt:

\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:
 $f(n) \leq c * g(n)$

Beispiel

$$x^2 + 2x + 1 = \mathcal{O}(x^2)$$

Denn, ab $x=10$ gilt:

$$x^2 + 2x + 1 \leq 2 * x^2$$



\mathcal{O} -Kalkül

Definition (\mathcal{O} -Kalkül)

Für zwei Funktionen $f, g: \mathbb{N} \rightarrow \mathbb{R}^+$ schreiben wir $f(n) = \mathcal{O}(g(n))$, falls c, n_0 existieren, so dass $\forall n > n_0$ gilt:

$$f(n) \leq c * g(n)$$

Beispiel

$$x^2 + 2x + 1 = \mathcal{O}(x^2)$$

Denn, ab $x=10$ gilt:

$$x^2 + 2x + 1 \leq 2 * x^2$$

Definition (Polynominelle Zeit)

Problem P lässt sich in polynomineller Zeit lösen \leftrightarrow Es existiert ein Algorithmus f , der das Problem P löst, für den gilt:

$$f(n) = \mathcal{O}(\text{beliebiges Polynom})$$



Komplexitätsklassen

Definition (P)

Problem $P \in \mathcal{P}$, genau wenn gilt:

P kann in **polynomineller** Zeit von einer **deterministischen** Turingmaschine **entscheiden** werden.



Komplexitätsklassen

Definition (P)

Problem $P \in \mathcal{P}$, genau wenn gilt:

P kann in **polynominaler** Zeit von einer **deterministischen** Turingmaschine **entscheiden** werden.

Definition (NP)

Problem $P \in \mathcal{NP}$, genau wenn gilt:

P kann in **polynominaler** Zeit von einer **nicht deterministischen** Turingmaschine **entschieden** werden.

Aufgabe 1

Gegeben folgender Algorithmus, der die Funktion $f(n)$ berechnet.

Algorithmus

1. Initialisiere Zähler z mit 2 und Ausgabe a mit 1
2. Berechne $n \% z$
3. Prüfe, ob $n \% z = 0$ gilt:
 - Falls ja: Setze a auf 0 und gehe zu Schritt 4.
 - Falls nein: Gehe zu Schritt 4.
4. Prüfe, ob $z < n$ gilt:
 - Falls ja: Erhöhe z um 1 und gehe zu Schritt 2.
 - Falls nein: Lösche das Band, schreibe a auf das Band und stoppe

Aufgabe 1.1

Was tut der Algorithmus?

Aufgabe 1

Gegeben folgender Algorithmus, der die Funktion $f(n)$ berechnet.

Algorithmus

1. Initialisiere Zähler z mit 2 und Ausgabe a mit 1
2. Berechne $n \% z$
3. Prüfe, ob $n \% z = 0$ gilt:
 - Falls ja: Setze a auf 0 und gehe zu Schritt 4.
 - Falls nein: Gehe zu Schritt 4.
4. Prüfe, ob $z < n$ gilt:
 - Falls ja: Erhöhe z um 1 und gehe zu Schritt 2.
 - Falls nein: Lösche das Band, schreibe a auf das Band und stoppe

Aufgabe 1.2

Wie lautet die Definition der Funktion f ?

Aufgabe 1

Gegeben folgender Algorithmus, der die Funktion $f(n)$ berechnet.

Algorithmus

1. Initialisiere Zähler z mit 2 und Ausgabe a mit 1
2. Berechne $n \% z$
3. Prüfe, ob $n \% z = 0$ gilt:
 - Falls ja: Setze a auf 0 und gehe zu Schritt 4.
 - Falls nein: Gehe zu Schritt 4.
4. Prüfe, ob $z < n$ gilt:
 - Falls ja: Erhöhe z um 1 und gehe zu Schritt 2.
 - Falls nein: Lösche das Band, schreibe a auf das Band und stoppe

Aufgabe 1.3

Gib die Komplexität des Algorithmus bezüglich der Eingabe n an.



Aufgabe 1

Gegeben folgender Algorithmus, der die Funktion $f(n)$ berechnet.

Algorithmus

1. Initialisiere Zähler z mit 2 und Ausgabe a mit 1
2. Berechne $n \% z$
3. Prüfe, ob $n \% z = 0$ gilt:
 - Falls ja: Setze a auf 0 und gehe zu Schritt 4.
 - Falls nein: Gehe zu Schritt 4.
4. Prüfe, ob $z < n$ gilt:
 - Falls ja: Erhöhe z um 1 und gehe zu Schritt 2.
 - Falls nein: Lösche das Band, schreibe a auf das Band und stoppe

Aufgabe 1.4

Gib die Komplexität des Algorithmus bezüglich der Länge der Binärdarstellung von n an.

Aufgabe 2

Definition ($\text{co-}\mathcal{P}$)

Die Komplexitätsklasse $\text{co-}\mathcal{P}$ sei definiert als die Menge der Sprachen \mathcal{L} , deren Komplementsprache \mathcal{L}^c in der Komplexitätsklasse \mathcal{P} liegt.

Aufgabe 2

Definition ($\text{co-}\mathcal{P}$)

Die Komplexitätsklasse $\text{co-}\mathcal{P}$ sei definiert als die Menge der Sprachen \mathcal{L} , deren Komplementsprache \mathcal{L}^C in der Komplexitätsklasse \mathcal{P} liegt.

Erinnerung

Zu einer Sprache \mathcal{L} über einem Alphabet Σ ist die Komplementsprache $\mathcal{L}^C = \Sigma^* \setminus \mathcal{L}$.

Aufgabe 2

Definition (co- \mathcal{P})

Die Komplexitätsklasse $\text{co-}\mathcal{P}$ sei definiert als die Menge der Sprachen \mathcal{L} , deren Komplementsprache \mathcal{L}^C in der Komplexitätsklasse \mathcal{P} liegt.

Erinnerung

Zu einer Sprache \mathcal{L} über einem Alphabet Σ ist die Komplementsprache $\mathcal{L}^C = \Sigma^* \setminus \mathcal{L}$.

Aufgabe

Beweise: $\text{co-}\mathcal{P} = \mathcal{P}$

Aufgabe 3

Beschreibung

Gegeben seien ein ungerichteter Graph $G = (V, E)$ mit Knoten $v \in V$ und Kanten $e = (v_1, v_2) \in E$ mit $v_1, v_2 \in V$ und zwei Farben A und B .

Aufgabe 3

Beschreibung

Gegeben seien ein ungerichteter Graph $G = (V, E)$ mit Knoten $v \in V$ und Kanten $e = (v_1, v_2) \in E$ mit $v_1, v_2 \in V$ und zwei Farben A und B .

Beweise

Die Sprache 2-COLOR =

$\{G \mid G = (V, E) \text{ ungerichteter Graph mit } \exists \text{ totale Funktion } g : V \rightarrow \{A, B\} : \forall (v_1, v_2) \in E : g(v_1) \neq g(v_2)\}$

liegt in der Komplexitätsklasse \mathcal{P} .

Aufgabe 4

Definition (2-SAT)

Gegeben eine in ihrer Größe polynomiell beschränkte aussagenlogische Formel F in konjunktiver Normalform, wobei jede Klausel genau 2 Literale enthält. F hat also die Form

$$F = \bigwedge_{i=1}^n (L_i \vee N_i),$$

wobei L_i und N_i Literale sind, also von der Form X oder $\neg X$ für eine Variable X sind.

Problem: Gibt es eine erfüllende Belegung für F ?

Aufgabe 4

Definition (2-SAT)

Gegeben eine in ihrer Größe polynomiell beschränkte aussagenlogische Formel F in konjunktiver Normalform, wobei jede Klausel genau 2 Literale enthält. F hat also die Form

$$F = \bigwedge_{i=1}^n (L_i \vee N_i),$$

wobei L_i und N_i Literale sind, also von der Form X oder $\neg X$ für eine Variable X sind.

Problem: Gibt es eine erfüllende Belegung für F ?

Aufgabe

Gib eine polynomielle Reduktion von 2-COLOR auf 2-SAT an!

Ende

Fragen?

There are three types of mathematicians: Those who can add and those who can't.