Datenstrukturen (DS)

Sommersemester 2015

Prof. Dr. Georg Schnitger
Dipl-Inf. Bert Besser
Hannes Seiwert, M.Sc.

GOETHE UNIVERSITÄT FRANKFURT AM MAIN

Institut für Informatik
AG Theoretische Informatik

Übung 0

Ausgabe: 14.04.2015
Abgabe: -

Wenn Ihnen noch kein Tutorium zugeteilt wurde, gehen Sie bitte zu einem Tutorium Ihrer Wahl.

Dieses Blatt wird in den ersten drei Wochen besprochen.
Themen dieses Präsenzblatts sind Laufzeitanalye, Logarithmus, Pseudocode, Induktion und Rekursion. Bitte schauen Sie sich das Kapitel 2 "Mathematische Grundlagen" im Skript an!

Viel Spaß!

Aufgabe 0.1. *Laufzeit unter Verdoppelung* (-)

Für Eingabelänge n seien die folgenden Laufzeitfunktionen $b_1, b_2, b_3, b_4 : \mathbb{N} \to \mathbb{N}$ gegeben:

$$b_1(n) = 100n, \quad b_2(n) = n^2, \quad b_3(n) = n^3, \quad b_4(n) = 2^n$$

Stellen Sie $b_i(2n)$ als Funktion von $b_i(n)$ dar. Wir untersuchen also, wie sich der Berechnungsaufwand verändert, wenn die Eingabelänge verdoppelt wird.

Können Sie $b_4(n+1)$ als Funktion von $b_4(n)$ schreiben?

Aufgabe 0.2. *Champions League vs. Bundesliga (Abschnitt 2.2 und 2.4.4 im Skript)* (-)

a) Eine Ausscheidungsrunde mit einer geraden Anzahl von n Fußball-Teams verläuft wie folgt. Die Teams treten gleichzeitig in zufälligen Paarungen in $n/2$ Spielen gegeneinander an, wobei jeweils das gewinnende Team für die nächste Runde qualifiziert ist.

   i. Wie viele einzelne Spiele finden statt, wenn in der ersten Runde $N = 2^k$ Teams für ein $k \in \mathbb{N}$ antreten?
   ii. Wie viele Runden werden gespielt bis der Gewinner feststeht?
   iii. Modellieren Sie das Turnier durch einen binären Baum mit N Blättern.
   iv. Die *Höhe* eines Baumes ist die maximale Anzahl von Kanten auf einem Weg von einem Blatt zur Wurzel. Welche Höhe hat ein binärer Baum mit l Blättern mindestens?

b) Wenn nun eine beliebige Anzahl N von Teams antritt, wieviele Runden werden gespielt, bis der Gewinner feststeht? Wenn zu Beginn einer Runde eine ungerade Anzahl von Teams überlebt hat, dann gelangt ein Team, ausgewählt durch ein Losverfahren, kampflos in die nächste Runde. Alle anderen Teams müssen sich weiterhin durch eine K.o.-Runde qualifizieren.

c) Angenommen, das Turnier besteht stattdessen aus Hin- und Rückrunde und in jeder Runde spielt jedes Team gegen jedes andere Team. Wie viele Spiele finden insgesamt statt? Wenn stets so viele Spiele wie möglich gleichzeitig stattfinden, wie viele "Spieltage" finden dann mindestens statt?

Aufgabe 0.3. *Fibonacci-Bäume (Abschnitt 2.3.4 im Skript)* (-)

Die Folge der Fibonacci-Zahlen ist definiert durch

$$\text{fib}(n) = \begin{cases} 1 & \text{falls } n = 1, 2, \\ \text{fib}(n-2) + \text{fib}(n-1) & \text{falls } n \geq 3. \end{cases}$$

`Algo1`, beschrieben auf Seite 27 im Skript, berechnet die Fibonacci-Folge. Sei B_n der Rekursionsbaum von `Algo1`(n). (Rekursionsbäume werden auf Seite 26 beschrieben.)

a) Sei $n \geq 2$. Zeigen Sie per vollständiger Induktion, dass die Höhe von B_n gleich $n-2$ ist.
b) Sei $n \geq 4$. Wie groß ist die maximale Länge eines einfachen Weges in B_n, der von einem Blatt zur Wurzel und wieder zu einem Blatt verläuft?

Aufgabe 0.4. *Fehlerhafte Induktion* (-)

Finden Sie die Fehler in den folgenden Beweisen:

a) (Abschnitt 2.3.4 im Skript) Wir zeigen, dass je zwei natürliche Zahlen a und b gleich sind. Dazu setzen wir $k = \max\{a, b\}$ und führen eine Induktion nach k. Im Basisschritt haben wir $k = 0$ und deshalb ist $a = 0 = b$, und das war zu zeigen. Im Induktionsschritt ist $\max\{a, b\} = k + 1$, und wir können die Induktionsbehauptung auf $a - 1$ und $b - 1$ anwenden, denn $\max\{a - 1, b - 1\} = k$. Also ist $a - 1 = b - 1$ und die Behauptung $a = b$ folgt.

b) (Abschnitt 2.3.4 im Skript) Wir zeigen, dass alle Pferde die gleiche Farbe besitzen und führen einen Beweis durch Induktion über die Zahl k aller Pferde. Im Basisschritt ist $k = 1$ und die Behauptung ist offensichtlich richtig. Für den Induktionsschritt nehmen wir an, dass es $k + 1$ Pferde $p_1, \ldots, p_{k+1}$ gibt. Dann haben aber nach Induktionsvoraussetzung $p_1, \ldots, p_k$ die Farbe von p_2 und $p_2, \ldots, p_{k+1}$ ebenfalls die Farbe von p_2 und wir haben die Behauptung gezeigt.

c) (Abschnitt 3.2 im Skript) Wir beweisen $\sum_{i=1}^{n}(2i + 1) = O(n)$ durch vollständige Induktion nach n.
Verankerung für $n = 1$: In diesem Fall ist $\sum_{i=1}^{1}(2i + 1) = 3 = O(1)$.
Induktionsschritt von n nach $n + 1$: Nach Induktionsannahme gilt $\sum_{i=1}^{n}(2i + 1) = O(n)$. Wir addieren auf beiden Seiten $2(n + 1) + 1$ und erhalten

$$\sum_{i=1}^{n}(2i + 1) + 2(n + 1) + 1 = O(n) + 2(n + 1) + 1\,.$$

Nach einer Vereinfachung folgt

$$\sum_{i=1}^{n+1}(2i + 1) = O(n) + 2n + 3\,.$$

Aber $(2n + 3) = O(n)$ und somit folgt

$$\sum_{i=1}^{n+1}(2i + 1) = O(n) + O(n) = O(n) = O(n + 1)\,.$$

Aufgabe 0.5. *Pseudocode (Abschnitt 3.3 im Skript)* (-)

Pseudocode ist eine Mischung von Umgangssprache, Programmiersprache und mathematischer Notation. Typischerweise ist Pseudocode kompakter aber gleichzeitig auch leichter zu verstehen als der entsprechende Code einer Programmiersprache. Pseudocode folgt keinen klaren Regeln, aber natürlich muss die nachfolgende Implementierung trivial sein!

a) Wie oft wird in den folgenden vier Algorithmen das Wort *Hallo* gedruckt?

1) Wiederhole n mal {
   Drucke *Hallo*;
   }
2) Solange $n > 1$ {
   Drucke *Hallo*;
   $n = n/2$;
   Gehe zum Anfang der Solange-Schleife;
   }
3) Wiederhole n mal {
   Wiederhole m mal {
   Drucke *Hallo*;
   }
   }
4) Wiederhole n mal {
   $m = n$;
   Solange $m > 1$ {
   Drucke *Hallo*;
   $m = m/2$;
   Gehe zum Anfang der Solange-Schleife;
   }
   }

b) Das Array L von n ganzen Zahlen sei aufsteigend sortiert: Also ist $L[0]$ die kleinste und $L[n-1]$ die größte Zahl. Gegeben sei der folgende Algorithmus.

$i = 0$;
Solange $i \leq n - 1$ {
Wenn $L[i] \geq 0$ gilt, dann verlasse die Solange-Schleife;
Setze $i = i + 1$;
Gehe zum Anfang der Solange-Schleife;
}
Gib i aus;

Die folgenden Fragen sind zu beantworten.

1) Welche Aufgabe erfüllt der Algorithmus? Eine umgangssprachliche Beschreibung genügt.
2) Wie viele Vergleiche und Additionen werden schlimmstenfalls durchgeführt. bis der Algorithmus terminiert?
3) Mit welcher Idee aus der Vorlesung kann die Anzahl der Vergleiche drastisch verringert werden?

c) Beschreiben Sie in Pseudocode, wie Sie das Minimum eines unsortierten Arrays von n Zahlen ermitteln.

Bitte wenden!

Aufgabe 0.6. *Das Sieb des Eratosthenes* (-)

Betrachten Sie die folgende C++-Funktion.

```
int eratosthenes(int n) {
  // am Ende gilt isprime[i]==true genau dann, wenn i eine Primzahl ist
  bool isprime[n+1];
  // Array initialisieren
  for (int i=0 ; i<n+1 ; i++) {
    isprime[i]=true;
  }
  for (int i=2 ; i<sqrt(n) ; i++) {
    // nicht-prime Zahlen markieren
    if (isprime[i]==true) {
      for (int j=2*i ; j<n+1 ; j+=i) {
        // wenn isprime[i]==true, dann ist i eine Primzahl
        // Vielfache von i sind natürlich keine Primzahlen
        isprime[j]=false; // (*)
      }
    }
  }
  // Primzahlen zählen
  int num=0;
  for (int i=2 ; i<n+1 ; i++) {
    if (isprime[i]==true) {
      num++;
    }
  }
  return num;
}
```

a) Übersetzen Sie die C++-Funktion in Pseudocode. Beachten Sie, dass zur Lösung dieser Aufgabe auch äußerst rudimentäre C++-Kenntnisse ausreichen.

b) Zeige, dass Anweisung `(*)` höchstens $\mathcal{O}(n \log_2 n)$ mal ausgeführt wird.

Hinweis: Es ist $\sum_{i=1}^{n} \frac{1}{i} \leq 1 + \log_2 n$. (Das Summensymbol $\sum_{i=1}^{n} a_i$ ist Abkürzung für die Summe der Zahlen $a_1, \ldots, a_n$. Es gilt also $\sum_{i=1}^{n} a_i := a_1 + a_2 + \cdots + a_n$.)

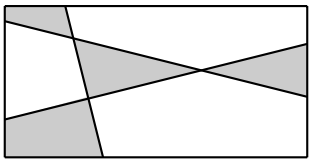
Also lassen sich alle Primzahlen kleiner 10^9 mehr oder minder mühelos berechnen!

Bitte wenden!

Aufgabe 0.7. *Mein Schreibtisch hat zwei Farben (Abschnitt 2.3.4 im Skript)* (-)

Ich nutze meinen Schreibtisch gerne als Notizzettel oder Schmierblatt. Leider hat die Reinigungskraft meine Bildchen, Gleichungen, etc. weggewischt. Zu allem Übel hat jemand auch noch lauter Geraden über meinen Schreibtisch gezogen. Dabei erstreckt sich jede Gerade von "Kante zu Kante" und endet nicht mitten auf meinem Schreibtisch.

Eine Zelle ist ein größtmögliches Stück Schreibtisch, welches nicht durch eine Gerade getrennt wird. Ich verspüre das dringende Bedürfnis, einige der Zellen mit meinem Bleistift auszumalen, sodass je zwei Zellen, die den selben Zellenrand berühren, nicht beide weiß oder beide schwarz sind, sondern unterschiedliche Farben haben. Zellen, die nur den selben Geradenschnittpunkt berühren, dürfen natürlich die gleiche Farbe haben. Ein Beispiel:



Werde ich Erfolg haben? Beweisen Sie per Induktion, dass für *jede* Menge von Geraden das resultierende Zellenmuster wie beschrieben mit zwei Farben gefärbt werden kann.