Aufgabenblatt 2 Sommersemester 2002 Claus/Lewandowski

Übungen zur Vorlesung Einführung in die Informatik II

Ausgabe: 30. April 2002 Abgabe: 6. Mai 2002, 12.00 Uhr

Die Abgabe erfolgt ausschließlich über das System eClaus! Siehe
http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ss02/info_II_02.html

Aufgabe 1: (Asymptotische Komplexitätsbeziehungen) zum Vot.

a. (1,5 Punkte) Zeigen oder widerlegen Sie folgende Aussagen:

- $3 \log_{10} n \in O(\log_2 n)$
- $(n + a)^b \in O(n^b)$, $a, b \in \mathbb{R}$ und $b > 0$
- $3^n \in O(2^n)$

b. (0,5 Punkte) Zeigen Sie anhand der Definitionen von O und Ω, dass gilt:

$$f(n) \in O(g(n)) \Leftrightarrow g(n) \in \Omega(f(n))$$

c. (2 Punkte) Welche Beziehung ($=$, $\subseteq$ oder $\supseteq$) gilt zwischen den folgenden Komplexitäten:

- $O(n\sqrt{n})$ und $O(n \log^2(n))$
- $O(3^n)$ und $O(n^n)$
- $O(2^{n+1})$ und $O(2^n)$
- $O(2^{2n})$ und $O(2^n)$

Aufgabe 2: (O-Notation) zum Votieren, 1 Punkt

Finden Sie einen Fehler im folgenden Argument.

Behauptung $f(n) = 2^n \in O(1)$.

Beweis Vollständige Induktion nach n. Die Behauptung ist richtig für $n = 1$, denn $2^1 = 2 \in O(1)$. Die Induktionsannahme ist nun also, dass $2^{n-1} \in O(1)$. Zu zeigen ist demnach, dass dann auch $2^n \in O(1)$. Das ist einfach, weil

$$2^n = 2 \cdot 2^{n-1} = 2 \cdot O(1) = O(1).$$

Aufgabe 3: (Analyse von Algorithmen) **schriftlich**

Gegeben ist folgender in Ada implementierter Algorithmus:

```
type Array_Of_Natural is array (Natural range <>) of Natural;

procedure Algorithm(A : in out Array_Of_Natural) is
      subtype Range_Type is Natural range A'RANGE;
      Tmp : Natural;
      Min : Range_Type;
begin
   for I in A'RANGE loop
      Min := I;                  -- (Z1)
      for J in I+1 .. A'LAST loop
         if A(J) < A(Min) then   -- (V1)
            Min := J;            -- (Z2)
         end if;
      end loop;

      if I /= Min then           -- (V2)
         Tmp := A(Min);          -- (Z3)
         A(Min) := A(I);         -- (Z4)
         A(I) := Tmp;            -- (Z5)
      end if;
   end loop;
end Algorithm;
```

a. (1 Punkt) Was bewirkt der Algorithmus?

b. (5 Punkte) **Uniforme Komplexität:** Wieviele Vergleiche bei (V1/2) und wieviele Zuweisungen bei Z(1–5) werden (abhängig von der Zahl n der Feldelemente in A) im schlechtesten Fall durchgeführt? Wieviele im besten Fall? In welcher Reihenfolge müssen die Feldelemente stehen, damit der beste bzw. schlechteste Fall eintritt? Geben Sie die Effizienz des Algorithmus in der O-Notation an.

Aufgabe 4: (Summen und Rekursionsgleichungen) **zum Vot.**

Bei der Analyse von Algorithmen treten häufig Summen und Rekursionsgleichungen wie die folgenden auf:

a. (2 Punkte) Welchen Wert haben die folgenden Summen?

(i) $\sum_{k=0}^{n} k^3$, für $n \in \mathbb{N}_0$

(ii) $\sum_{k=0}^{\infty} kx^k$, $x \in \mathbb{R}$, $0 < x < 1$

b. (2 Punkte) Bringen Sie die folgenden Rekursionsgleichungen in eine geschlossene Form, d.h. geben Sie $T(n)$ derart an, dass auf der rechten Seite der Gleichung T nicht mehr auftaucht.

(i) $T(0) = 0$ und $T(n) = 2T(n-1) + 1$ für $n \geq 1$.

(ii) $T(1) = 1$ und $T(n) = 2T(\frac{n}{2}) + n$ für $n = 2^k, k \in I\!N_0$.

(iii) $T(1) = 1$ und $T(n) = 2T(\frac{n}{2}) + n^2$ für $n = 2^k, k \in I\!N_0$.

Aufgabe 5: (Entwurf und Analyse eines Algorithmus) schriftlich

Ein Münzenproblem: Unsere Währung hat Centmünzen mit den Werten 1, 2, 5, 10, 20 und 50 Cent. Damit lassen sich alle Werte von 1 bis 100 Cent mit maximal 6 Münzen darstellen. Folgende Greedy-Strategie führt dabei zum optimalen Ergebnis: Solange der Wert noch nicht erreicht ist, nehme jeweils die Münze mit dem höchstmöglichen Wert. Für 59 Cent würde man z.B. mit 50 Cent anfangen, dann eine 5 Cent und zwei 2 Cent Münzen nehmen.

a. (2 Punkte) Schreiben Sie ein Ada 95 Programm, das für die Werte 1 bis 100 Cent mit obiger Greedy-Strategie berechnet, wieviel Münzen minimal zur Darstellung nötig sind. Geben Sie den Mittelwert der benötigten Münzenanzahl für die Werte 1 bis 100 Cent an.

b. Bei anderen Aufteilungen - z.B. 1, 2, 5, 10, 24 und 50 Cent - führt die Greedy-Strategie nicht immer zum richtigen Ergebnis: Bei 58 Cent ermittelt die Greedy-Strategie 4 Münzen (50+5+2+1), obwohl die Darstellung auch mit 3 Münzen möglich ist (2*24+10).

(5 Punkte) Schreiben Sie ein Ada 95 Programm, das 6 Münzwerte einliest und dann für alle Werte 1 bis 100 Cent jeweils korrekt ermittelt, wieviel Münzen zur Darstellung mindestens benötigt werden. Geben Sie aus, für wieviel Werte weniger, gleich bzw. mehr Münzen gebraucht werden im Vergleich zur Darstellung mit 1, 2, 5, 10, 20 und 50 Cent Münzen. Geben Sie auch jeweils den Mittelwert an.

c. Zusatzaufgabe (1 Punkt): Versuchen Sie durch Probieren sechs Münzwerte zu finden, mit denen sich die Werte 1 bis 100 Cent im Mittel mit weniger Münzen darstellen lassen als es mit 1, 2, 5, 10, 20 und 50 Cent möglich ist.

d. Zusatzaufgabe (schwer! - bis zu 4 Punkte): Finden Sie notwendige und hinreichende Bedingungen, damit die minimale Darstellung mit den gegebenen Münzwerten für beliebige Werte mit der Greedy Strategie gefunden werden kann.

Es werden nur 20 Punkte des Aufgabenblattes gewertet. Die Zusatzaufgaben sind zum Verständnis des Vorlesungsstoffes nicht notwendig und werden in den Übungen nicht besprochen.

Das Eintragen in die Übungsgruppen ist abgeschlossen. Die Listen (incl. Rauminformation) sind weiterhin unter

https://inf2.informatik.uni-stuttgart.de/uebungsgruppen-bin/inf2/groups

einzusehen. Ein- und Umtragen ist dann nur noch bei freier Kapazität direkt über Stefan.Lewandowski@informatik.uni-stuttgart.de möglich.

Die Folien zur Vorlesung liegen zum Kopieren im Semesterapparat und bei der Fachschaft, sowie online unter

http://www.informatik.uni-stuttgart.de/ifi/fk/lehre/ss02/info_II_02.html

Dort werden auch alle weitere Informationen zur Vorlesung und den Übungen bekanntgegeben, insbesondere gibt es dort Informationen zu dem System eClaus, über das die Abgabe der Übungen erfolgt.

Fragen zur Vorlesung und den Übungen, sowie Anregungen und Kritik können auf dem Schwarzen Brett

http://fachschaft.informatik.uni-stuttgart.de/forum/

diskutiert werden.