Aufgabenblatt 11

Besprechung am Dienstag, den 31. Januar 2006, 8:00 Uhr V38.01

Aufgabe 1 (Laufzeit von Programmen)

Bestimmen Sie das asymptotische Laufzeitverhalten der folgenden Programmstücke.

```
procedure m1(in n:integer) is begin
  if (n >= 0) then begin
    for  i  in 1.. n; loop
    put(i);
    m1(n - 1000 );
  end if;
end m1;
```

```
procedure m2(in n:integer) is
summe, i :integer;
begin
  summe:= 0; i:=1;
  while(i <= 2*n*n) loop
    summe = summe + 1;
    i = i + 1;
  end loop;
end m2;
```

Aufgabe 2 (Programmanalyse)

Ermitteln Sie für das im Folgenden angegebene Programm zunächst die genaue Anzahl der ausgeführten Instruktionen (eine Instruktion entspricht jeweils der "Abarbeitung" einer Zeile (ohne "end;"-Zeilen)) und geben Sie hierauf basierend die Laufzeitkomplexität mittels der O-Notation an, wobei die Eingabegröße durch die mit einem entsprechenden Wert vorbelegte Variable n repräsentiert sei!

```
a := 0;
while a <> n do
        a := a+1;
        b := 0;
        while b <> n do
                a := a+1;
                b := b+1;
        end;
        a := a-1;
end;
```

Aufgabe 3 (Programmanalyse)

Gegeben Sei folgendes Programmstück mit konstantem n und bekanntem $z < 2^n$.

```
procedure Aufg3 is
  a:array[1..n] of natural :=(0| others => 0);
  begin
    for i in (1 .. z) loop
      for j in (1 .. n) loop
        if A(j)=1 then A(j):=0
                  else A(j):=1; j:=n;
        end if;
      end loop;
    end loop;
  end Aufg3;
```

Was tut dieses Programmstück? Schätzen Sie seine Laufzeit möglichst genau ab.

Aufgabe 4 ehem. Prüfungsaufgabe

Zeigen Sie:

$$O(f(n) + g(n)) = O(\max\{f(n), g(n)\}) \qquad \text{für Funktionen } f, g : \mathbb{N} \to \mathbb{N}$$

Aufgabe 5

Wir definieren:

$f_1(n) = \log\log\log n$	$f_8(n) = n \log_2 n$	$f_{15}(n) = 2^n$
$f_2(n) = \log\log n$	$f_9(n) = n^{\frac{3}{2}}$	$f_{16}(n) = e^n$
$f_3(n) = \sqrt{\log n}$	$f_{10}(n) = n^2$	$f_{17}(n) = n!$
$f_4(n) = \log_2 n$	$f_{11}(n) = n^{10}$	$f_{18}(n) = n^n$
$f_5(n) = (\log_2 n)^2$	$f_{12}(n) = n^{\log_2 n}$	$f_{19}(n) = 2^{n^2}$
$f_6(n) = n^{\frac{1}{3}}$	$f_{13}(n) = 2^{\sqrt{n}}$	
$f_7(n) = n$	$f_{14}(n) = 1.1^n$	$f_{20}(n) = 2^{2^n}$

Zeigen Sie für $1 \leq i \leq 19$:

$$\lim_{n \to \infty} \frac{f_i(n)}{f_{i+1}(n)} = 0.$$

Aufgabe 6

Geben Sie einfache $O(\dots)$-Ausdrücke für die folgenden Funktionen an und $\Omega(\dots)$-Ausdrücke, soweit möglich.

(a) $g(n) = 10\log_2 n + 4\sqrt{3n}$

(b) $g(n) = 2n(\log_2 n + 5\log_2\log_2 n)$

(c) $g(n) = n^2\left(\lceil\log_2(n+5)\rceil + 7\right)$

(d) $g(n) = n^2 2^n + 2n^4 3^n + 10n^n$

(e) $g(n) = n^{10} + n^2 \cdot 3^n + n^5 \cdot 2^n \cdot \log n$

(f) $g(n) = |\sin n| + |\cos n|$

(g) $g(n) = e^n - \lfloor e^n \rfloor$

Aufgabe 7 (Eigenschaften von Graphen)

Ein ungerichteter Graph (V, E) heißt *bipartit*, wenn es eine Aufteilung der Knotenmenge V in zwei disjunkte Mengen V_1 und V_2 mit $V_1 \cup V_2 = V$ gibt, so dass jede Kante aus E einen Endpunkt in V_1 und einen in V_2 hat.

Beschreiben Sie einen möglichst effizienten Algorithmus, der für einen gegebenen ungerichteten Graphen entscheidet, ob dieser bipartit ist, und ggf. eine entsprechende Partition von V berechnet. Dabei können Sie annehmen, dass der Graph durch sein Knotenarray und seine Adjazenzlisten gegeben ist. Zeigen Sie die Korrektheit des Algorithmus und bestimmen Sie seine Laufzeit in O-Notation.

Beschreiben Sie einen möglichst effizienten Algorithmus, der entscheidet, ob ein ungerichteter Graph (V, E) einen Kreis ungerader Länge enthält. Begründen Sie seine Korrektheit und bestimmen sie seine Laufzeit in O-Notation.

Aufgabe 8 Programmanalyse

Analysieren Sie das folgende Programm. Welchen Aufwand hat es bei Eingabe von n Elementen?

```
PROCEDURE Exchange IS NEW Swap_Generic(ValueType => ElementType);

AnotherPassNeeded: Boolean := True;
CurrentBottom: IndexType := List'Last;

BEGIN
  WHILE AnotherPassNeeded LOOP
    AnotherPassNeeded := False;
    FOR Current IN List'First .. CurrentBottom - 1 LOOP
      IF KeyOf(List(Current + 1)) < KeyOf(List(Current)) THEN
        Exchange(List(Current), List(Current + 1));
        AnotherPassNeeded := True;
      END IF;
    END LOOP;
  END LOOP;
END;
```

Aufgabe 9 (Abschätzung in O - Notation - ehem. Prüfungsaufgabe)

Mit O(g) bezeichnen wir, wie in der Vorlesung, die Menge der Funktionen, die von der Ordnung g sind:

$$O(g) = \{f: N_0 \to R_+ | \text{ es gibt ein } c > 0 \text{ und ein } n_0 \in N_0 \text{ mit } f(n) \leq c\, g(n) \text{ für alle } n \geq n_0\}.$$

Entscheiden Sie, welche der folgenden Behauptungen richtig und welche falsch sind:
(für jede falsche Antwort wird ein Punkt abgezogen!!!)

a) $3n^2 + 17n + 23 \in O(n^3)$
b) $3n^2 + 17n + 23 \in O(n^2)$
c) $3n^2 + 17n + 23 \in O(n)$
d) $2^n + 25n \in O(n^{200})$
e) $10^n \in O(2^n)$
f) $115n^{17} + 12n^{12} + 4n^5 + 123 \in O(n^{20})$

Aufgabe 10 Ordnen von Funktionen nach asymptotischem Verhalten

Sortieren Sie die folgenden Funktionen in aufsteigender Reihenfolge gemäß der O-Notation:

$$\log_2(\frac{1}{2}n), \quad \sqrt{n^3}, \quad n!, \quad 10^4 n, \quad n^2 + \log_2 n^n, \quad \ln(2), \quad n^2, \quad n^3 + 5n^2, \quad 2^n$$

Ordnen Sie folgende Funktionen bezogen auf die O-Notation und begründen Sie Ihre Wahl:

$$f_1(n) = \begin{cases} n & \text{falls } n \text{ ungerade} \\ 3 * n^3 & \text{sonst} \end{cases} \qquad f_2(n) = \begin{cases} 3 * n^3 & \text{falls } n \text{ ungerade} \\ n^2 & \text{sonst} \end{cases}$$

$$f_3(n) = 2^n + 3^n \qquad f_4(n) = \sqrt{\frac{n}{2}}$$

Aufgabe 11 Beweise von Aussagen über asymptotisches Verhalten

(a) Zeigen oder widerlegen Sie folgende Behauptungen:

(i) $o(g) \cap \mathcal{O}(g) = \emptyset$

(ii) $f \in \mathcal{O}(g) \Rightarrow f \in o(g)$

(iii) $\sum_{i=0}^{n} i^2 \in \mathcal{O}(n^2)$

(b) Zeigen Sie folgende Behauptungen.

(i) $(n+a)^b \in \Theta(n^b)$, wobei a, b feste Konstanten sind (keine Kenngrößen), $a, b \in \mathbb{N}$, $a, b \geq 1$

(ii) $\lceil n^{1.5} + n \log_e n \rceil \in \mathcal{O}(\lceil n^{1.5} \rceil)$

Geben Sie eine geeignete Konstante c und ein geeignetes n_0 an, so dass die Aussage folgt.

Zeigen Sie mit Hilfe der Definition: Sind $f, g : \mathbf{N} \to \mathbf{N}$ mit $f(n) = O(n^k)$ und $g(n) = O(n^l)$ und Konstanten $k, l \in \mathbf{N}$ gegeben, so folgt $g(f(n)) = O(n^{kl})$.

Allgemeine Hinweise:

- Bei weiteren Fragen, wenden Sie sich bitte an W. Schmid (sltsoftware@yahoo.de).
- Weitere Hinweise finden Sie auf unserer Veranstaltungswebseite unter:
  http://www.info2.de.vu
  http://www.zusatzkurs.de.vu