Claus/Hagenah/Lewandowski — Sommersemester 1999

Aufgabenblatt 1

Übungen zur Vorlesung „Entwurf und Analyse effizienter Algorithmen“

---

1. a) Gegeben seien Funktionen $f, g, h : \mathbb{N} \to \mathbb{N}$. Beweisen Sie bzw. widerlegen Sie folgende Behauptungen unter der Voraussetzung, daß $f, g \in O(h)$ gilt:

$$f + g \in O(h) \qquad f - g \in o(h) \qquad f \in O(g)$$

Gilt für beliebige $f, g : \mathbb{N} \to \mathbb{N}$ stets $f \in O(g) \;\vee\; g \in O(f)$?

b) Zeigen Sie für eine beliebige Funktion $f : \mathbb{N} \to \mathbb{N}$:

(i) $f \in \Theta(2^{\lceil \log f \rceil})$

(ii) $\Theta(f) = \{g : \mathbb{N} \to \mathbb{N} \mid \exists c > 0 \; \exists n_0 \; \forall n \geq n_0 : \; 1/c\, f(n) \leq g(n) \leq c\, f(n)\}$

2. Ordnen Sie die folgenden Funktionen nach der Ordnung $f \leq g :\Leftrightarrow f \in O(g)$:

$$2^{\sqrt{n}}, \quad 3^{\log(n^3)}, \quad n^n, \quad n^{3.01}, \quad 2^{n^2}, \quad \log(n)$$
$$n^3 \log(n), \quad \log^3(\log(n)), \quad n^2 2^n, \quad (n+2)^3,$$
$$n^{1.5}, \quad n^3 \log(n^3) + 1, \quad \sqrt{n!}, \quad n^{3\log(n)}, \quad n^2 \log(n!)$$

3. Gegeben sei die Rekursionsgleichung

$$f(n+2) = a\, f(n+1) + b\, f(n)$$

mit Anfangswerten $f(0), f(1)$, wobei $a, b \in \mathbb{N}$ mit $b \neq 0$ gilt. Die Gleichung 2-ten Grades $x^2 = a\,x + b$ wird als charakteristische Gleichung bezeichnet. Zeigen Sie:

(i) Falls die charakteristische Gleichung zwei verschiedene Lösungen $r_1 \neq r_2$ besitzt, so lautet die Lösung der Rekursionsgleichung $f(n) = C_1\, r_1^n + C_2\, r_2^n$, wobei die Konstanten C_1, C_2 die Lösungen des folgenden Gleichungssystems sind:

$$C_1 + C_2 = f(0) \qquad C_1\, r_1 + C_2\, r_2 = f(1)$$

(ii) Falls $r_1 = r_2$ gilt, so lautet die Lösung der Rekursionsgleichung $f(n) = r_1^n\,(C_1 + n\, C_2)$, wobei $C_1 = f(0)$ und $C_2 = \frac{f(1) - r_1\, f(0)}{r_1}$ gelten.

4. Bestimmung der maximalen Teilfolgensumme: Gegeben sei ein Feld $A(1..n)$von n Integerwerten (positiv und negativ). Gesucht ist der Wert

$$b = \max\{b_{l,r} : b_{l,r} = \Sigma_{i=l}^{r} A(i), 1 \leq l \leq r \leq n\}$$

Es sei $b = 0$, falls A nur aus negativen Zahlen besteht. Die beiden folgenden Algorithmen berechnen die maximale Teilfolgensumme. Zeigen Sie für beide Algorithmen die Korrektheit, geben Sie die Zeit- und Platzkomplexität an und begründen Sie diese. (siehe auch im Skript Seite 19ff.)

```
N : constant Integer := 1000;
type Bereich is range 1..N;

A : array(Bereich) of Integer;

function Max(X,Y: Integer) return Integer is
begin
   if X>Y then return X; else return Y; end if;
end Max;

function Recursive_Linear return Integer is
   T,P,S,B : Integer;

   procedure Tpsb(L,R: Bereich; T,P,S,B: in out Integer) is
      C: Bereich;
      T_L, T_R, P_L, P_R, S_L, S_R, B_L, B_R: Integer;
   begin
      if L=R then
         T := A(L);
         S := Max(0,A(L));
         P := Max(0,A(L));
         B := Max(0,A(L));
      else
         C := Bereich((Integer(R)+Integer(L))/2);
         Tpsb(L, C,   T_L, P_L, S_L, B_L);
         Tpsb(C+1, R, T_R, P_R, S_R, B_R);
         T := T_L+T_R;
         P := Max(P_L, T_L+P_R);
         S := Max(S_L+T_R, S_R);
         B := Max(Max(B_L, B_R), S_L+P_R);
      end if;
   end Tpsb;

begin
   Tpsb(Bereich'First, Bereich'Last,T,P,S,B);
   return B;
end Recursive_Linear;

function Iterative_Square return Integer is
   Best : Integer := 0;
   Sum : Integer;
begin
   for Left in Bereich loop
      Sum := 0;
      for Right in Left..Bereich'Last loop
         Sum := Sum+A(Right);
         Best := Max(Best,Sum);
      end loop;
   end loop;
   return Best;
end Iterative_Square;
```