Übungsblatt 1

Uli Köhler (10580373), Tobias Harrer (10575835)

29. April 2013

1 Aufgabe 2

Idee: Erweiterung des 'Cleveren Algorithmus', sodass in der letzten Fallunterscheidung auch rMaxScores zugelassen werden, die gleichgroß wie der bestehende maxScore sind. Die Start- und Endpositionen jedes weiteren maxScore werden in einer Liste gespeichert:

```
MSS Clever (int[] a, int n)
begin
    int maxscore := 0;
    int rmaxscore := 0; rstart := 1;
    int[] lStart; int[] rEnd; int MSScounter := 0;
    for (i := 1; i ≤ n; i++) do
        if (rmaxscore > 0) then
            rmaxscore := rmaxscore + a[i];
        else
            rmaxscore := a[i]; rstart := i;
        if (rmaxscore ≥ maxscore) then
            maxscore := rmaxscore;
            lStart[MSScounter] := rstart;
            rEnd[MSScounter] := i;
            MSScounter++;
end
```

Die Laufzeit bleibt bei $\mathcal{O}(n)$, da nur in der letzten Fallunterscheidung 3 Operationen hinzukommen, die in konstanter Laufzeit geschehen (Schreiben in Array, Addieren zu einem Integer). Werden hingegen lStart und rEnd als Listen implementiert, benötigt das Hinzufügen zur Liste lineare Laufzeit, die Gesamtlaufzeit würde in $\mathcal{O}(n^2)$ liegen.

2 Aufgabe 3

Die Einträge sind die Eingabegrößen, die der jeweilige Algorithmus in der jeweiligen Zeit bewältigt:

	1s	60s	3,600s	86,400s	2,592,000s
T_1	2	120	7,200	172,800	5,184,000
T_2	≈7	≈164	≈5,763	≈103,700	≈2,445,000
T_3	≈32	≈245	≈1,897	≈9,295	≈50,912
T_4	≈46	≈182	≈711	≈2,052	≈6,376
T_5	≈13	≈16	≈20	≈23	≈26

Die Einträge wurden folgendermaßen berechnet (z.B. 1s, T_1):
$1s * 1000 * \frac{1}{s} = 500 * n \Leftrightarrow n = 2$

3 Aufgabe 4

3.1 Algorithmen mit polynomieller Laufzeit:

Da T_1 linear wächst ($T_1 \in \mathcal{O}(n)$), kann auch die Eingabegröße um das 100-fache steigen.
T_3 wächst quadratisch ($T_1 \in \mathcal{O}(n^2)$), daher kann bei 100-facher Eingabegröße die Eingabe um das $\sqrt{100} = 10$-fache steigen.
$T_4 \in \mathcal{O}(n^3)$, daher kann die Eingabegröße um das $\sqrt[3]{100} \approx 4.6$-fache steigen.
Die Steigerung der Eingabegröße kann mithilfe der x-ten Wurzel berechnet werden, da sich die Eingabegröße folgendermaßen errechnet:
$t * o = a * n^x \Leftrightarrow n = \sqrt[x]{\frac{t*o}{a}}$ (o = Elementaroperationen pro Sekunde). Dabei handelt es sich um die jeweilige Umkehrfunktion.

3.2 Algorithmen mit exponentieller Laufzeit:

$t * o = a * x^n \Leftrightarrow n = \log_x(\frac{t*o}{a})$
Werden die Elementaroperationen pro Sekunde verhundertfacht, so wird auch das Argument des Logarithmus mit dem 100-fachen multipliziert. Des weiteren gilt:
$\log(a * b) = \log(a) + \log(b)$. Da in diesem Fall durch das Verhundertfachen der Rechenleistung lediglich das Argument des Logarithmus beeinflusst wird, können folglich konstant zusätzlich $\log_3(100) \approx 4$ Operationen zusätzlich ausgeführt werden.

3.3 Algorithmen mit logarithmischer Laufzeit:

$T_2(n) = n * \log(n) = \log(n^n)$ (da: $\log(a * b) = \log(a) + \log(b)$).
Dann gilt: $x = log_2(n^n) \Leftrightarrow n^n = 2^x \Leftrightarrow n = \sqrt[n]{2^x}$. Da n auf beiden Seiten der Gleichung steht, ist die Lösung nicht ohne Weiteres ersichtlich.