

Kursveränderungen einer Aktie:

7 $\underbrace{-11 \quad 18 \quad 10}_{17}$ -23 -3 27 -1

Wann kaufen? Wann wieder verkaufen?

Gewinn: $18 + 10 - 23 - 3 + 27 = 29$

Problem: Maximum Subarray Sum (MSS)

Eingabe: $a_1, \dots, a_n \in \mathbb{Z}$ (ganze Zahlen), $n \geq 1$

Ausgabe: grösstmögliche Teilsumme

$$S^* = a_i + \dots + a_j \quad (i, j \in \{1, \dots, n\}, i \leq j)$$

$$S^* = 0 \quad \text{falls alle Zahlen negativ (leere Teilsumme)}$$

naiver Algorithmus: berechne alle Teilsummen

$$S_{i,j} := a_i + \dots + a_j \quad (i \leq j)$$

$(S_{i,j} := 0 \text{ für } i > j)$

Anzahl Additionen: $A(n)$ (bestimmend für Laufzeit)

Pseudo code:

For $i : 1 \dots n$

Additionen

$$S_{i,i} \leftarrow a_i$$

0

$$S_{i,i+1} \leftarrow a_i + a_{i+1}$$

+1

$$S_{i,i+2} \leftarrow a_i + a_{i+1} + a_{i+2}$$

+2

$\vdots$

$\vdots$

$$S_{i,n} \leftarrow a_i + \dots + a_n$$

+ (n-i)

$$\left. \begin{array}{l} 0 \\ +1 \\ +2 \\ \vdots \\ +(n-i) \end{array} \right\} \frac{(n-i) \cdot (n-i+1)}{2}$$

$\frac{(n-i)^2}{2} \leq \frac{(n-i+1)^2}{2}$

Total ($n \geq 2$)

siehe Übungen

$$A(n) \leq \frac{1}{2} (n^2 + (n-1)^2 + \dots + 2^2 + 1^2) = \frac{n \cdot \overbrace{(n+\frac{1}{2})}^{\leq 2n} \cdot \overbrace{(n+1)}^{\leq 2n}}{6} \leq n^3$$

$$A(n) \geq \frac{1}{2} ((n-1)^2 + (n-2)^2 + \dots + 1^2 + 0^2) = \frac{\overbrace{(n-1)}^{\geq n/2} \cdot \overbrace{(n-\frac{1}{2})}^{\geq n/2} \cdot n}{6} \geq \frac{n^3}{24}$$

Also: $A(n) \leq O(n^3)$ und $n^3 \leq O(A(n))$ ($n \geq 2$)

Schreibweise: $A(n) = \Theta(n^3)$ "gleiche Ordnung"

geht es besser?

Idee: Addition assoziativ

$$S_{i,j} = (a_i + \dots + a_{j-1}) + a_j = S_{i,j-1} + a_j$$

Pseudo code:

For $i : 1 \dots n$

Additionen

$$S_{i,i} \leftarrow a_i$$

$$S_{i,i+1} \leftarrow S_{i,i} + a_{i+1}$$

$$S_{i,i+2} \leftarrow S_{i,i+1} + a_{i+2}$$

$\vdots$

$$S_{i,n} \leftarrow S_{i,n-1} + a_n$$

0
+1
+1
 $\vdots$
+1

} $n-i$

Total: $A(n) = (n-1) + (n-2) + \dots + 1 = \frac{n \cdot (n-1)}{2} \begin{cases} \leq n^2/2 \\ \geq n^2/4 \end{cases}$

$$\leadsto A(n) = \Theta(n^2)$$

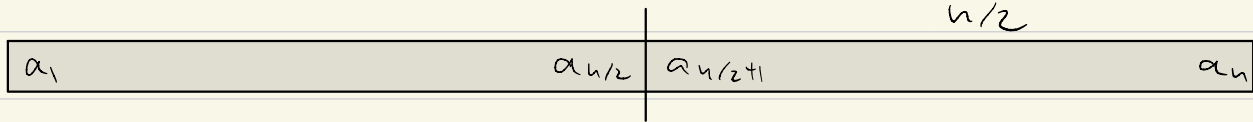
(da $\frac{n}{2} \leq n-1 \leq n$)

geht es besser?

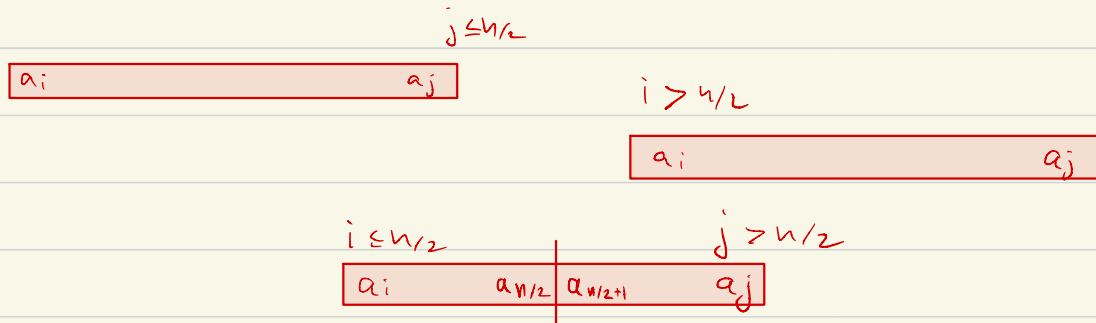
müssen wir alle $\Theta(n^2)$ Teilsummen berechnen?

MSS rekursiv

führe Lösung zurück auf Lösungen
(vgl. Karatsuba) mit $\frac{n}{2}$
Zahlen



3 Fälle für S^*



Algorithmus: (n gerade)

1. löse beide Hälften rekursiv (fertig in Fällen 1, 2)
2. berechne $\max \{ S_{i,j} \mid i \leq n/2 < j \}$ (fertig in Fall 3)
3. Ausgabe: beste der 3 Teillösungen (deckt alle Fälle ab)

Wie geht Schritt 2?

$$\max_{i \leq \frac{n}{2} < j} S_{i,j} = \max_{i \leq \frac{n}{2} < j} \left(S_{i, \frac{n}{2}} + S_{\frac{n}{2}+1, j} \right)$$

wähle i und j voneinander unabhängig

$$= \left(\max_{i \leq \frac{n}{2}} \text{---} \right) + \left(\max_{j > \frac{n}{2}} \text{---} \right)$$

- berechne $(S_{\frac{n}{2}+1, j})_{j=\frac{n}{2}+1}^n$ wie zuvor $\frac{n}{2}-1$ Additionen
- berechne $(S_{i, \frac{n}{2}})_{i=1}^n$ analog $\frac{n}{2}-1$ Additionen
- $\leadsto$ Schritt 2 geht mit $2 \cdot (\frac{n}{2}-1) + 1 = n-1$ Additionen

siehe Pseudocode mit $i=\frac{n}{2}+1$

Rekurrenz: $A(n) = 2 \cdot A\left(\frac{n}{2}\right) + n - 1$

Tabelle: $n = 2^k, k \in \mathbb{N}$

Anzahl	Länge	Additionen
1	2^k	$+(2^k - 1)$
2	2^{k-1}	$+ 2 \cdot (2^{k-1} - 1) = 2^k - 2$
2^2	2^{k-2}	$+ 2^2 \cdot (2^{k-2} - 1) = 2^k - 2^2$
$\vdots$		
2^{k-1}	2^1	$+ 2^{k-1} \cdot (2^1 - 1) = 2^k - 2^{k-1}$
2^k	2^0	0
		$k \cdot 2^k - (2^k - 1) = (k-1) \cdot 2^k + 1$
		$\leq k \cdot 2^k$

$$= n \cdot \log_2(n)$$

$$\leadsto A(n) = \Theta(n \log_2 n) \quad (n \geq 2)$$

geht es besser?

MSS anders rekursiv

- Idee:
- führe zurück auf Lösung für $n-1$ Zahlen
 - passe dafür Lösungsbegriff an

Randmaxima: $R_j := \max_{i \leq j} S_{ij}$

Beispiel

j	1	2	3	4	5	6	7	8
a_j	7	-11	18	10	-23	-3	27	-1
R_j	7	-4	18	28	5	2	29	28

Algorithmus

- berechne $R_1, \dots, R_{n-1}$ rekursiv mit
$$R_j \leftarrow \begin{cases} R_{j-1} + a_j & \text{falls } R_{j-1} \geq 0 \\ a_j & \text{sonst} \end{cases}$$
- Ausgabe: $S^* \leftarrow \max \{ R_1, \dots, R_n \}$

Analyse

n	1	2	3	...	n	Total
$A(n)$	0	+1	+1	...	+1	$n-1$

Laufzeit $O(n)$ (- Laufzeit durch Additionen bestimmt)