OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

INFORMATIK UND
MATHEMATIK

Algorithmen und Datenstrukturen

- Grundlagen (Komplexität) -

Prof. Dr. Klaus Volbert

Wintersemester 2018/19
Regensburg, 15. Oktober 2018

Beispiel MaxTeilSum

- Eingabe: $a_0, \ldots, a_{n-1} \in \mathrm{Z}$ (n ganze Zahlen)
- Ausgabe: Maximale Teilsumme, d.h.

$$s = \max_{0 \le i \le j \le n-1} \sum_{k=i}^{j} a_k$$

- Anwendungen
  - Erkennung von grafischen Mustern
  - Analyse von Aktienkursen
    (täglich neue Kurse: Ermittlung bester Ein-/Ausstiegszeitpunkt)
- Beispiel:
  - Eingabe: -13, 25, 34, 12, -3, 7, -87, 28, -77, 11
  - Ausgabe: 75 (ergibt sich aus $i = 1, j = 5$)

MaxTeilSum4 (Divide-&-Conquer)

```
int MaxTeilsum4(int a[],int f, int l) { int n = l - f + 1;
    if (n == 1) return a[f];
      else {
         int newn = (n % 2 == 0 ? n / 2 : n / 2 + 1);

         int MaxBorderSum1=a[f+newn-1],i=f+newn-2,currVal=MaxBorderSum1;
         while (i>=f) { currVal+=a[i];
                if (currVal>MaxBorderSum1) MaxBorderSum1=currVal;
                i--; }

         int MaxBorderSum2=a[f+newn],i=f+newn+1,currVal=MaxBorderSum2;
         while (i<=l) { currVal+=a[i];
                if (currVal>MaxBorderSum2) MaxBorderSum2=currVal;
                i++; }

         return max(MaxTeilsum4(a,f,f+newn-1),
                 max(MaxTeilsum4(a,f+newn,l),MaxBorderSum1+
                                                  MaxBorderSum2)); }}
```

Triviallösung

Divide: Teile a in zwei Teile

Conquer: Berechne die Teillösungen

Merge: Füge die Einzelergebnisse zusammen

- Laufzeit: $T(n) = \Theta(n \log n)$

Beispiel MaxTeilSum4 I

Folge	MBS1	MBS2	Σ	MT1	MT2	Max
-13 25 34 12 -3 7 -87 28 -77 11	68	7	75	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11	59	12	71	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11	25	34	59	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11	-13	25	12	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11						-13
-13 25 34 12 -3 7 -87 28 -77 11						25
-13 25 34 12 -3 7 -87 28 -77 11	-13	25	12	-13	25	25
-13 25 34 12 -3 7 -87 28 -77 11						34
-13 25 34 12 -3 7 -87 28 -77 11	25	34	59	25	34	59
-13 25 34 12 -3 7 -87 28 -77 11	12	-3	9	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11						12

Beispiel MaxTeilSum4 II

Folge	MBS1	MBS2	Σ	MT1	MT2	Max
-13 25 34 12 -3 7 -87 28 -77 11						-3
-13 25 34 12 | -3 7 -87 28 -77 11	12	-3	9	12	-3	12
-13 25 34 | 12 -3 7 -87 28 -77 11	59	12	71	59	12	71
-13 25 34 12 -3 7 -87 28 | -77 11	28	-66	-38	?	?	?
-13 25 34 12 -3 7 -87 | 28 -77 11	-80	28	-52	?	?	?
-13 25 34 12 -3 7 | -87 28 -77 11	7	-87	-80	?	?	?
-13 25 34 12 -3 7 -87 28 -77 11						7
-13 25 34 12 -3 7 -87 28 -77 11						-87
-13 25 34 12 -3 7 | -87 28 -77 11	7	-87	-80	7	-87	7
-13 25 34 12 -3 7 -87 28 -77 11						28
-13 25 34 12 -3 7 -87 | 28 -77 11	-80	28	-52	7	28	28

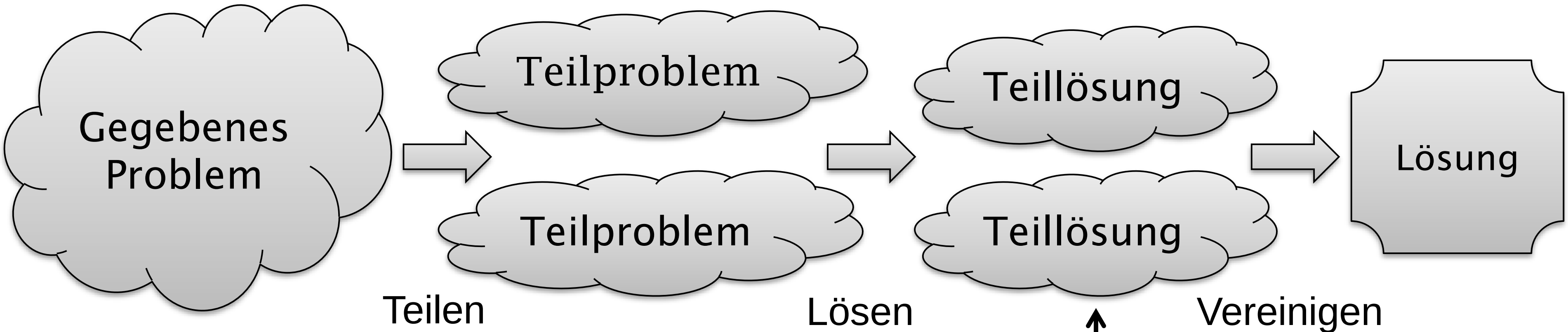
Beispiel MaxTeilSum4 III

<table>
<tr><th>Folge</th><th>MBS1</th><th>MBS2</th><th>Σ</th><th>MT1</th><th>MT2</th><th>Max</th></tr>
<tr><td>-13 25 34 12 -3 7 -87 28 -77 | 11</td><td>-77</td><td>11</td><td>-66</td><td>?</td><td>?</td><td>?</td></tr>
<tr><td>-13 25 34 12 -3 7 -87 28 -77 11</td><td></td><td></td><td></td><td></td><td></td><td>-77</td></tr>
<tr><td>-13 25 34 12 -3 7 -87 28 -77 11</td><td></td><td></td><td></td><td></td><td></td><td>11</td></tr>
<tr><td>-13 25 34 12 -3 7 -87 28 -77 | 11</td><td>-77</td><td>11</td><td>-66</td><td>-77</td><td>11</td><td>11</td></tr>
<tr><td>-13 25 34 12 -3 7 -87 28 | -77 11</td><td>28</td><td>-66</td><td>-38</td><td>28</td><td>11</td><td>28</td></tr>
<tr><td>-13 25 34 12 -3 | 7 -87 28 -77 11</td><td>68</td><td>7</td><td>75</td><td>71</td><td>28</td><td>75</td></tr>
</table>

- …und jetzt mit:
  - Eingabe: -5, 13, -32, 7, -3, 17, 23, 12, -35, 19
  - Ausgabe: ?

Entwurfsprinzip: Divide & Conquer

- Schema: Teilen und Herrschen



- Typische Einschränkungen
  - Teilprobleme müssen unabhängig voneinander lösbar sein
  - Gesamtlösung muss aus Teillösungen entstehen können (Vereinigung)
- Teilung bis zum Erreichen des Trivialproblems (oft rekursiv)
- Beispiele
  - Quicksort, Schnelle Fouriertransformation (FFT)

Anmerkungen

- Korrektheitsbeweise häufig mit vollständiger Induktion
  - Identifikation einer Bedingung, die nach allen Schleifendurchläufen gilt (Schleifeninvariante, Analyse: vor, während, nach)
  - Bedingung für das Verlassen der Schleife zusammen mit der Schleifeninvariante liefern das gewünschte Ergebnis
- Komplexitätsabschätzung durch Aufstellen von
  - Komplexitätsgleichungen ($T(n) = \ldots$)
  - Rekursionsgleichungen mit Rekursionsbasis ($T(n) = \ldots, T(a) = b$)
- Lösen von Rekursionsgleichungen durch
  - Substitutionsmethode (Lösung raten, Korrektheit beweisen)
  - Iterationsmethode (sukzessives Einsetzen liefert Abschätzung)
  - Master-Methode (jetzt)

Master-Methode

- Rekursionsgleichungen haben oft die Form
  - $T(1) = 1, T(n) = aT\left(\frac{n}{b}\right) + f(n)$ mit $a \geq 1, b > 1, f\colon \mathrm{IN}_0 \to \mathrm{IN}_0$
- Das Master-Theorem gibt an, wie man solche Gleichungen lösen kann:
  - 1. Fall: Falls $f(n) = O(n^{\log_b a-\varepsilon})$ für ein $\varepsilon > 0$, dann: $T(n) = \Theta\left(n^{\log_b a}\right)$
    Anmerkung: $f(n)$ wächst schwächer als $aT\left(\frac{n}{b}\right)$
  - 2. Fall: Falls $f(n) = \Theta(n^{\log_b a})$, dann: $T(n) = \Theta\left(n^{\log_b a} \log n\right)$
    Anmerkung: $f(n)$und $aT\left(\frac{n}{b}\right)$ wachsen gleich, dazu kommt: $\log$-Faktor
  - 3. Fall: Falls $f(n) = \Omega(n^{\log_b a+\varepsilon})$ für ein $\varepsilon > 0$ und falls $af\left(\frac{n}{b}\right) \leq cf(n)$ für ein $c < 1$ und alle $n \geq n_0$, dann: $T(n) = \Theta\left(f(n)\right)$
    Anmerkung: f wächst stärker als $aT\left(\frac{n}{b}\right)$
- Bemerkungen:
  - Aussagen gelten auch für … $T\left(\left\lceil\frac{n}{b}\right\rceil\right)$ … und … $T\left(\left\lfloor\frac{n}{b}\right\rfloor\right)$ …
  - Beweise können in Cormen, Leiserson, Rivest, Stein: Introduction to Algorithms, 3rd Ed., MIT Press, 2009 nachgelesen werden (Kapitel 4)

Beispiele zur Master-Methode I

- $T(n) = 2T\left(\frac{n}{2}\right) + n$ (Rekursionsgleichung MaxTeilSum4)
  - $a = 2, b = 2, f(n) = n$, es gilt $\log_b a = \log_2 2 = 1$
  - D.h. $f(n) = \Theta\left(n^{\log_b a}\right) = \Theta(n)$ und nach Fall 2:

$$T(n) = \Theta(n \log n)$$

- $T(n) = 9T\left(\frac{n}{3}\right) + n$
  - $a = 9, b = 3, f(n) = n$, es gilt $\log_b a = \log_3 9 = 2$
  - D.h. $f(n) = O\left(n^{\log_b a - \varepsilon}\right)$ für ein $\varepsilon > 0$ und nach Fall 1:

$$T(n) = \Theta(n^2)$$

Beispiele zur Master-Methode II

- $T(n) = 2T\left(\frac{n}{2}\right) + n\log n$
  - $a = 2, b = 2, f(n) = n\log n$, es gilt $\log_b a = \log_2 2 = 1$
  - Zusätzlich gilt: $f(n) = \Omega\left(n^{\log_b a+\varepsilon}\right)$ für ein $\varepsilon > 0$, da $n\log n \geq cn^{1+\varepsilon}$
  - Was zunächst für Fall 3 spricht, aber gilt $af\left(\frac{n}{b}\right) \leq cf(n)$ für ein $c < 1$?
  - Aus $2\frac{n}{2}\log\frac{n}{2} \leq cn\log n$ folgt $1 - \frac{1}{\log n} \leq c$. D.h. aber $c < 1$ existiert nicht
  - Daher ist Fall 3 nicht anwendbar (alle anderen Fälle auch nicht)
- Beobachtung:
  - Zwischen Fall 2 und Fall 3 existiert eine Lücke
  - Man kann erweitern:

    Wenn $f(n) = \Theta(n^{\log_b a}(\log n)^k)$ für $k \geq 0$, dann $T(n) = \Theta(n^{\log_b a}(\log n)^{k+1})$

Beispiele zur Master-Methode III

- Lösen Sie folgende Rekursionsgleichungen mit der Master-Methode:

1. $T(n) = 8T\left(\frac{n}{3}\right) + n^2$

2. $T(n) = 9T\left(\frac{n}{3}\right) + n^2$

3. $T(n) = 10T\left(\frac{n}{3}\right) + n^2$

Beispiele zur Master-Methode III

- Lösen Sie folgende Rekursionsgleichungen mit der Master-Methode:

1. $T(n) = 8T\left(\frac{n}{3}\right) + n^2$
   - $a = 8,\ \ b = 3,\ \ f(n) = n^2 = \Omega\left(n^{\log_3 8 + \varepsilon}\right)$ für $\varepsilon > 0$ (also evtl. Fall 3)
   - Prüfe: $8 \cdot f\left(\frac{n}{3}\right) = 8 \cdot \frac{n^2}{9} \leq \frac{8}{9} \cdot n^2 = c \cdot f(n)$ mit $c = \frac{8}{9} < 1$ für alle $n \geq n_0 = 1$
   - Also nach Fall 3: $T(n) = \Theta\left(f(n)\right) = \Theta(n^2)$
2. $T(n) = 9T\left(\frac{n}{3}\right) + n^2$
   - $a = 9,\ \ b = 3,\ \ f(n) = n^2 = \Theta\left(n^{\log_3 9}\right) = \Theta\left(n^{\log_3 3^2}\right) = \Theta(n^2)$
   - Also nach Fall 2: $T(n) = \Theta\left(n^{\log_3 3^2} \cdot \log(n)\right) = \Theta(n^2 \cdot \log(n))$
3. $T(n) = 10T\left(\frac{n}{3}\right) + n^2$
   - $a = 10,\ b = 3,\ f(n) = n^2 = O\left(n^{\log_3 10 - \varepsilon}\right)$ für $\varepsilon > 0$
   - Also nach Fall 1: $T(n) = \Theta\left(n^{\log_3 10}\right) = \Theta(n^{2{,}0959})$

Überblick

- Einführung und Organisation
- Grundlagen
  - Begriffe
    - Algorithmus, Datentyp, Datenstruktur, Datenstruktur Stapel
  - Korrektheit und Komplexität
    - Totale Korrektheit, Iterationen, Rekursionen
    - RAM, Church'sche These, O-Notation (O, Ω, Θ)
  - Rekursionsgleichungen
    - Iterationsmethode, Substitutionsmethode, Master-Methode
  - Entwurfsmethode Divide & Conquer
- Sortieralgorithmen