

# Info I – Übungsblatt 4

Joachim Breitner

mit Aufgaben von Martin Kiefel und Christian Maier

28. November 2005



Ich hätte die Rechnerübung tatsächlich vergessen...

# Entschuldigung

Soll nicht mehr vorkommen.

Falls doch: Meine Handynummer ist auf meiner Homepage.

# Unser Programm heute



- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen



- Wer ist aktiv ohne Übungsblätter abzugeben?
- Studiengänge beim nächsten Übungsblatt angeben!

- 1 **Übungsblatt 3**
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen

# Übungsblatt-Rückblick



- Etwas weniger erfreuliches Ergebnis:
  - Schnitt nur 17 von 23 Punkten
- Häufigste Fehler
  - Lebendigkeit nicht begründet oder falsch angegeben
  - Nicht auf Federn und Beton bzw. Materialunabhängigkeit eingegangen

# Errata



Beim letzten Tutorium habe ich zwei Fehler gemacht:

- `new` ist kein gültiger Bezeichner
- Fließkommazahlen in Java mit `.` statt `,`

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz**
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen





# Effizienzberechnung



## Problem

- Ist ein Algorithmus besser als ein anderer?
- Gilt das auch auf einer anderen Rechenmaschine?
- Gilt es auch, wenn die Datenmenge weiter wächst?

## Lösungsansatz

Wir abstrahieren die Effizienz von Algorithmen:

- Unabhängig von der Rechenmaschine
- In Abhängigkeit der Eingabelänge (meist  $n$ )
- Mathematisch fundiert und beweisbar



# Effizienzberechnung



## Problem

- Ist ein Algorithmus besser als ein anderer?
- Gilt das auch auf einer anderen Rechenmaschine?
- Gilt es auch, wenn die Datenmenge weiter wächst?

## Lösungsansatz

Wir abstrahieren die Effizienz von Algorithmen:

- Unabhängig von der Rechenmaschine
- In Abhängigkeit der Eingabelänge (meist  $n$ )
- Mathematisch fundiert und beweisbar



# O-Kalkül



## Definition

$$O(g(n)) = \{f(n) \mid \exists n_0 \in \mathbb{N} \exists c \in \mathbb{R} \forall n \geq n_0 : 0 \leq f(n) \leq c \cdot g(n)\}$$

## Umgangssprachlich

$O(g(n))$  enthält alle nicht-negativen Funktionen, die langsamer oder so schnell wie  $g(n)$  wachsen.

Dabei kümmern wir uns nicht

- darum, was am Anfang passiert ( $\exists n_0 \in \mathbb{N} \dots \forall n \geq n_0$ ).
- um einfache Faktoren ( $\exists c \in \mathbb{R} \dots c \cdot g(n)$ ).

# O-Kalkül



## Definition

$$O(g(n)) = \{f(n) \mid \exists n_0 \in \mathbb{N} \exists c \in \mathbb{R} \forall n \geq n_0 : 0 \leq f(n) \leq c \cdot g(n)\}$$

## Umgangssprachlich

$O(g(n))$  enthält alle nicht-negativen Funktionen, die langsamer oder so schnell wie  $g(n)$  wachsen.

Dabei kümmern wir uns nicht

- darum, was am Anfang passiert ( $\exists n_0 \in \mathbb{N} \dots \forall n \geq n_0$ ).
- um einfache Faktoren ( $\exists c \in \mathbb{R} \dots c \cdot g(n)$ ).



# Rechnen mit dem $O$ -Kalkül



Man kann aus der obigen Definition ein paar Regeln ableiten, mit denen der Umgang deutlich einfacher wird.

(Eigentlich rechnet man hier mit Mengen, und auch sonst ist die Schreibweise nicht mathematisch korrekt, aber das wird meistens unter den Teppich gekehrt)

## Faustregeln

- $O(1) + O(23) + O(4) = O(1234)$
- $O(f(n)) + O(f(n)) = O(f(n))$
- $O(a \cdot f(n)) = O(f(n)) \quad \forall a \in \mathbb{R}$
- $O(a \cdot n^2 + b \cdot n + c) = O(n^2) \quad \forall a, b, c \in \mathbb{R}$
- $n^a = O(p^n) \quad \forall a \in \mathbb{N}, p > 1$

„Der Stärkere gewinnt!“



# Rechnen mit dem $O$ -Kalkül



Man kann aus der obigen Definition ein paar Regeln ableiten, mit denen der Umgang deutlich einfacher wird.

(Eigentlich rechnet man hier mit Mengen, und auch sonst ist die Schreibweise nicht mathematisch korrekt, aber das wird meistens unter den Teppich gekehrt)

## Faustregeln

- $O(1) + O(23) + O(4) = O(1)$
- $O(f(n)) + O(f(n)) = O(f(n))$
- $O(a \cdot f(n)) = O(f(n)) \quad \forall a \in \mathbb{R}$
- $O(a \cdot n^2 + b \cdot n + c) = O(n^2) \quad \forall a, b, c \in \mathbb{R}$
- $n^a = O(p^n) \quad \forall a \in \mathbb{N}, p > 1$

„Der Stärkere gewinnt!“



# Effizienz von Selectionsort



```

Setze i=0;

Solange i<n {
  Setze j=i, min=∞, stelle=i;

  Solange j<n {
    Wenn min>liste[j] {
      Setze min=liste[j];
      Setze stelle=j;
    }
    Erhöhe j um 1;
  }

  Tausche liste[i], liste[stelle];
  Erhöhe i um 1;
}

```



# Effizienz von Selectionsort



Setze  $i=0$ ;  $O(1)$

```

Solange  $i < n$  {
  Setze  $j=i$ ,  $min=\infty$ ,  $stelle=i$ ;

  Solange  $j < n$  {
    Wenn  $min > liste[j]$  {
      Setze  $min=liste[j]$ ;
      Setze  $stelle=j$ ;
    }
    Erhöhe  $j$  um 1;
  }

  Tausche  $liste[i]$ ,  $liste[stelle]$ ;
  Erhöhe  $i$  um 1;
}

```





# Effizienz von Selectionsort



Setze  $i=0$ ;

$O(1)$

$0 \dots (n-1)$

```

Solange  $i < n$  {
  Setze  $j=i$ ,  $min=\infty$ ,  $stelle=i$ ;

  Solange  $j < n$  {
    Wenn  $min > liste[j]$  {
      Setze  $min=liste[j]$ ;
      Setze  $stelle=j$ ;
    }
    Erhöhe  $j$  um 1;
  }

  Tausche  $liste[i]$ ,  $liste[stelle]$ ;
  Erhöhe  $i$  um 1;
}

```

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
                                           0... (n-1)
Solange i<n {
    Setze j=i, min=∞, stelle=i;           O(1)

    Solange j<n {
        Wenn min>liste[j] {
            Setze min=liste[j];
            Setze stelle=j;
        }
        Erhöhe j um 1;
    }

    Tausche liste[i], liste[stelle];
    Erhöhe i um 1;
}
  
```

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
Solange i<n {                               0... (n-1)
    Setze j=i, min=∞, stelle=i;            O(1)
    Solange j<n {                           i... (n-1)
        Wenn min>liste[j] {
            Setze min=liste[j];
            Setze stelle=j;
        }
        Erhöhe j um 1;
    }
    Tausche liste[i], liste[stelle];
    Erhöhe i um 1;
}
  
```

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
Solange i<n {                               0... (n-1)
  Setze j=i, min=∞, stelle=i;              O(1)
  Solange j<n {                             i... (n-1)
    Wenn min>liste[j] {                     O(1)
      Setze min=liste[j];                  O(1)
      Setze stelle=j;                      O(1)
    }
    Erhöhe j um 1;                          O(1)
  }
  Tausche liste[i], liste[stelle];
  Erhöhe i um 1;
}

```

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
Solange i<n {                               0... (n-1)
    Setze j=i, min=∞, stelle=i;             O(1)
    Solange j<n {                           i... (n-1)
        Wenn min>liste[j] {
            Setze min=liste[j];
            Setze stelle=j;
        }
        Erhöhe j um 1;                       O(1)
    }
    Tausche liste[i], liste[stelle];        O(1)
    Erhöhe i um 1;                           O(1)
}
  
```

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
Solange i<n {                               0... (n-1)
    Setze j=i, min=∞, stelle=i;             O(1)
    Solange j<n {                             i... (n-1)
        Wenn min>liste[j] {
            Setze min=liste[j];
            Setze stelle=j;
        }
        Erhöhe j um 1;                       O(1)
    }
    Tausche liste[i], liste[stelle];        O(1)
    Erhöhe i um 1;                           O(1)
}
  
```

$$O\left(1 + \sum_{i=0}^{n-1} \left(1 + \sum_{j=i}^{n-1} (1 + 1 + 1) + 1\right) + 1 + 1\right)$$

# Effizienz von Selectionsort



```

Setze i=0;                                O(1)
Solange i<n {                               0... (n-1)
    Setze j=i, min=∞, stelle=i;             O(1)
    Solange j<n {                             i... (n-1)
        Wenn min>liste[j] {
            Setze min=liste[j];
            Setze stelle=j;
        }
        Erhöhe j um 1;                       O(1)
    }
    Tausche liste[i], liste[stelle];        O(1)
    Erhöhe i um 1;                           O(1)
}
  
```

$$\begin{aligned}
 O\left(1 + \sum_{i=0}^{n-1} \left(1 + \sum_{j=i}^{n-1} (1 + 1 + 1) + 1\right) + 1 + 1\right) \\
 = O(n^2)
 \end{aligned}$$

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung**
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen





# Regelung vs. Steuerung



## Steuerung

Ein Algorithmus oder eine Schaltung nimmt Einfluß auf eine beliebige Größe.

## Regelung

Wieder nimmt ein Algorithmus oder eine Schaltung Einfluß auf eine beliebige Größe.

Aber diesmal findet eine Rückkopplung statt und die beeinflusste Größe wird gemessen und fließt wieder in den Algorithmus ein.

# Beispiel: Heizung (1)



## Außentemperaturgesteuerte Heizung

Bei einer außentemperaturgesteuerten Heizung misst eine Schaltung über einen Sensor die Außentemperatur und ändert in Abhängigkeit davon die Leistung der Heizkörper, um z. B. bei kaltem Wetter den Innenraum stärker zu heizen.

Steuerung oder Regelung?



# Beispiel: Heizung (1)



## Außentemperaturgesteuerte Heizung

Bei einer außentemperaturgesteuerten Heizung misst eine Schaltung über einen Sensor die Außentemperatur und ändert in Abhängigkeit davon die Leistung der Heizkörper, um z. B. bei kaltem Wetter den Innenraum stärker zu heizen.

Steuerung oder Regelung?

## Begründung

Bei dieser Heizung wird die Leistung der Heizkörper in Abhängigkeit von der Außentemperatur geändert. Sie verfügt über keine Messeinrichtung für die tatsächliche Innentemperatur (Rückkopplung). Eventuell auftretende Störgrößen (z.B. offene Fenster) können hier nicht berücksichtigt werden.



# Beispiel: Heizung (1)



## Außentemperaturgesteuerte Heizung

Bei einer außentemperaturgesteuerten Heizung misst eine Schaltung über einen Sensor die Außentemperatur und ändert in Abhängigkeit davon die Leistung der Heizkörper, um z. B. bei kaltem Wetter den Innenraum stärker zu heizen.

## Steuerung

## Begründung

Bei dieser Heizung wird die Leistung der Heizkörper in Abhängigkeit von der Außentemperatur geändert. Sie verfügt über keine Messeinrichtung für die tatsächliche Innentemperatur (Rückkopplung). Eventuell auftretende Störgrößen (z.B. offene Fenster) können hier nicht berücksichtigt werden.



# Beispiel: Heizung (1)



## Außentemperaturgesteuerte Heizung

Bei einer außentemperaturgesteuerten Heizung misst eine Schaltung über einen Sensor die Außentemperatur und ändert in Abhängigkeit davon die Leistung der Heizkörper, um z. B. bei kaltem Wetter den Innenraum stärker zu heizen.

## Steuerung

## Begründung

Bei dieser Heizung wird die Leistung der Heizkörper in Abhängigkeit von der Außentemperatur geändert. Sie verfügt über keine Messeinrichtung für die tatsächliche Innentemperatur (Rückkopplung). Eventuell auftretende Störgrößen (z.B. offene Fenster) können hier nicht berücksichtigt werden.

# Beispiel: Heizung (2)



Wie machen wir daraus eine Regelung?

## Erweiterung zur Regelung

Man kann diese Heizung um einen Regler erweitern, an dem der Benutzer die gewünschte Innenraumtemperatur einstellen kann. Dazu wird die Innentemperatur gemessen, der Außentempersensor fällt stattdessen weg. Die Heizung kann nun ständig die gemessene Temperatur mit der gewünschten vergleichen (Ist-/ Sollwert-Vergleich) und bei Abweichungen entsprechend die Leistung der Heizkörper erhöhen bzw. verringern. Bei einer derartig modifizierten Heizung handelt es sich um eine Regelung.

# Beispiel: Heizung (2)



Wie machen wir daraus eine Regelung?

## Erweiterung zur Regelung

Man kann diese Heizung um einen Regler erweitern, an dem der Benutzer die gewünschte Innenraumtemperatur einstellen kann. Dazu wird die Innentemperatur gemessen, der Außentempersensor fällt stattdessen weg. Die Heizung kann nun ständig die gemessene Temperatur mit der gewünschten vergleichen (Ist-/ Sollwert-Vergleich) und bei Abweichungen entsprechend die Leistung der Heizkörper erhöhen bzw. verringern. Bei einer derartig modifizierten Heizung handelt es sich um eine Regelung.

# Beispiele



Entscheidet was man hier vorfindet:

- Die Sonne regelt das Wachstum auf der Erde.
- Ein Lautstärkeregler an einer Stereoanlage.
- Die  $\lambda$ -Sonde im Auto.
- Die Helligkeitseinstellung eines Monitors.



# Beispiele



Entscheidet was man hier vorfindet:

- Die Sonne regelt das Wachstum auf der Erde.
- Ein Lautstärkeregler an einer Stereoanlage.
- Die Lambdasonde im Auto.
- Die Helligkeitseinstellung eines Monitors.

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML**
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen

# Klassen und Objekte



## Klassen

Klassen sind Schablonen für Gegenständen. In ihnen sind die Gemeinsamkeiten einer Menge von Gegenständen vereint, vor allem

- Verhalten (→ Methoden)
- Eigenschaften (→ Attribute)

ClassName	
attrib1:	Type1
attrib2:	Type2
method1()	
method2()	

## Objekte

Objekte sind konkrete Instanzen einer Klasse. Erst Objekte haben Werte für die Attribute der Klasse, und kein „eigenes“ Verhalten.

Object: ClassName	
attrib1:	Value1
attrib2:	Value2



# Klassen und Objekte



## Klassen

Klassen sind Schablonen für Gegenständen. In ihnen sind die Gemeinsamkeiten einer Menge von Gegenständen vereint, vor allem

- Verhalten (→ Methoden)
- Eigenschaften (→ Attribute)

ClassName	
attrib1:	Type1
attrib2:	Type2
method1()	
method2()	

## Objekte

Objekte sind konkrete Instanzen einer Klasse. Erst Objekte haben Werte für die Attribute der Klasse, und kein „eigenes“ Verhalten.

Object: ClassName	
attrib1:	Value1
attrib2:	Value2

# UML Klassen- und Objektdiagramme



Modeliert folgendes Objekt und seine Klasse mit den genannten Eigenschaften. Bezieht auch get- und set-Methoden mit ein.

## Drucker

- 20 Seiten/Minute
- Laserdrucker
- 20000 €
- HB Moepjet 10001

Zusätzlich: Eine Methode zum Drucken

## UML-Drucker



Printer	
Name:	String
Price:	Float
PagePerMinute:	Float
PrinterType:	Integer
getName() setName() getPrice() setPrice() getPagePerMinute() setPagePerMinute() getPrinterType() setPrinterType() print()	

<u>MoepJet: Printer</u>	
Name:	"HB Moepjet 10001"
Price:	20000
PagePerMinute:	20
PrinterType:	0

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner**
- 6 Übungsblatt 4
- 7 Fragen

# Von-Neumann-Rechner



## Fragen

- Aus welchen Komponenten besteht ein von Neumann-Rechner?
- Beschreiben Sie das Problem des so genannten von-Neumann'schen Flaschenhalses.
- Wie könnte man diesen Flaschenhals abschwächen?

## Antworten



# Von-Neumann-Rechner



## Fragen

- Aus welchen Komponenten besteht ein von Neumann-Rechner?
- Beschreiben Sie das Problem des so genannten von-Neumann'schen Flaschenhalses.
- Wie könnte man diesen Flaschenhals abschwächen?

## Antworten

- Ein von-Neumann-Rechner besteht aus der Zentraleinheit (CPU, auch Prozessor) des Rechners, dem Speicher zur Aufbewahrung von binären Programm- und Dateninformationen, der Verbindungseinrichtung (Bus) sowie Ein- und Ausgabeeinheiten.

# Von-Neumann-Rechner



## Fragen

- Aus welchen Komponenten besteht ein von Neumann-Rechner?
- Beschreiben Sie das Problem des so genannten von-Neumann'schen Flaschenhalses.
- Wie könnte man diesen Flaschenhals abschwächen?

## Antworten

- Ein von-Neumann-Rechner besteht aus der Zentraleinheit (CPU, auch Prozessor) des Rechners, dem Speicher zur Aufbewahrung von binären Programm- und Dateninformationen, der Verbindungseinrichtung (Bus) sowie Ein- und Ausgabeeinheiten.
- Das Problem des so genannten von-Neumann'schen Flaschenhalses resultiert aus physikalischen Gegebenheiten. Der Rechnerkern kann heute Befehle um etwa den Faktor 10 schneller ausführen als Speicherzugriffe.

# Von-Neumann-Rechner



## Fragen

- Aus welchen Komponenten besteht ein von Neumann-Rechner?
- Beschreiben Sie das Problem des so genannten von-Neumann'schen Flaschenhalses.
- Wie könnte man diesen Flaschenhals abschwächen?

## Antworten

- Ein von-Neumann-Rechner besteht aus der Zentraleinheit (CPU, auch Prozessor) des Rechners, dem Speicher zur Aufbewahrung von binären Programm- und Dateninformationen, der Verbindungseinrichtung (Bus) sowie Ein- und Ausgabeeinheiten.
- Das Problem des so genannten von-Neumann'schen Flaschenhalses resultiert aus physikalischen Gegebenheiten. Der Rechnerkern kann heute Befehle um etwa den Faktor 10 schneller ausführen als Speicherzugriffe.
- Beispielsweise durch Einführung von Registern, das sind lokale Speicherzellen im Rechnerkern, Caches oder Verwendung getrennter Busse für den Zugriff auf Daten und Programm.

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4**
- 7 Fragen

# Hinweise zum Übungsblatt



- Algorithmen in Pseudocode
- Regelung & Steuerung: Begründung wichtig
- Animation nicht notwendig, Screenshots genügen

- 1 Übungsblatt 3
- 2 Algorithmen-Effizienz
- 3 Regelung und Steuerung
- 4 UML
- 5 Von-Neumann-Rechner
- 6 Übungsblatt 4
- 7 Fragen**

# Fragen



# Fragen?

