Algorithmen I

Tutorium 1 - 1. Sitzung

Dennis Felsing

dennis.felsing@student.kit.edu
www.stud.uni-karlsruhe.de/~ubcqr/algo

2011-04-18

Vorstellung

Ich

- Dennis Felsing
- Informatik Bachelor 4. Semester
- Algorithmen vor einem Jahr bei Sanders
- Hobby: Radfahren

Und ihr?

Tutorium

Sinn des Tutoriums

- Vorlesungsstoff vertiefen
- Auf Übungsblatt vorbereiten
- Fragen beantworten
- Fragen, Vorschläge gerne auch per E-Mail an `dennis.felsing@student.kit.edu`

Informationen

- Vorlesung: VAB im Vorlesungsverzeichnis
- Tutorium: `www.stud.uni-karlsruhe.de/~ubcqr/algo`

Übungsblätter

Termine

- Bearbeitung: Zwei Wochen Zeit
- Abgabe: Mittwochs 12:00 in Kasten im Untergeschoss Informatik-Hauptgebäude (wo wir gerade sind)

Bearbeitung

- **Freiwillig**: Keine Klausurvoraussetzung, Bonuspunkte, ...
- Gruppenarbeit und -abgabe erlaubt
- Auf Deckblatt: Name, **Tutorium 1**
- Leserlich schreiben

Modulprüfung

Klausur

- Notwendig für Modul
- Anmeldung im Studienportal
- Hauptklausur: **19.07.2011** (Dienstag nach Vorlesungszeit)
- Nachklausur: 10.10.2011

Einführung

Was ist ein Algorithmus?

Ein **Algorithmus** ist eine Handlungsvorschrift zur Lösung eines Problems.

Überblick

Werkzeuge

Pseudocode

Definition

Pseudocode ist Programmcode, der zur Veranschaulichung eines Algorithmus dient.

- Ähnlich zu normalem Code (Java, C++, Python, ...)
- Wird nicht von Computer verarbeitet
- Kann Ausdrücke in natürlicher Sprache enthalten
- Kurz und präzise
- Keine feste Norm

Pseudocode

Beispiel

SUMME(A)

```
1  sum = 0
2  for i = 1 to A.length
3        // Füge A[i] zu sum hinzu
4        sum = sum + A[i]
5  return sum
```

Aufgabe

Entwerfe einen Algorithmus MAX(A), der die größte Zahl im Feld A zurückgibt.

Pseudocode

Aufgabe

Entwerfe einen Algorithmus $\textsc{Max}(A)$, der die größte Zahl im Feld A zurückgibt.

Lösung

$\textsc{Max}(A)$

```
1   max = 0
2   for i = 1 to A.length
3         if A[i] > max
4               max = A[i]
5   return max
```

O-Kalkül

Definitionen

$$O(g(n)) = \{f(n) \mid \exists c > 0 : \exists n_0 > 0 : \forall n \geq n_0 : f(n) \leq c \cdot g(n)\}$$
$$o(g(n)) = \{f(n) \mid \forall c > 0 : \exists n_0 > 0 : \forall n \geq n_0 : f(n) < c \cdot g(n)\}$$

$$\Omega(g(n) = \{f(n) \mid \exists c > 0 : \exists n_0 > 0 : \forall n \geq n_0 : f(n) \geq c \cdot g(n)\}$$
$$\omega(g(n) = \{f(n) \mid \forall c > 0 : \exists n_0 > 0 : \forall n \geq n_0 : f(n) > c \cdot g(n)\}$$

$$\Theta(g(n)) = \{f(n) \mid \exists c_1, c_2 > 0 : \exists n_0 > 0 : \forall n \geq n_0 : c_1 \cdot g(n) \leq f(n) \leq c_2 \cdot g(n)\}$$

$$\Theta(g(n)) = O(g(n)) \cap \Omega(g(n))$$

O-Kalkül

Aufgaben

Zeige oder widerlege:

- $3n^2 + n \in O(n^2)$ ✓
- $2n \in \omega(n)$ ×
- $\log n \in o(n)$ ✓
- $0.00000001n \in \Theta(n)$ ✓
- $2^{n+1} \in O(2^n)$ ✓

Suchen und Sortieren

Insertionsort

INSERTION-SORT(A)

```
1  for j = 2 to A.length
2      key = A[j]
3      // Füge A[j] in die sortierte Sequenz A[1..j − 1] ein.
4      i = j − 1
5      while i > 0 and A[i] > key
6          A[i + 1] = A[i]
7          i = i − 1
8      A[i + 1] = key
```

Mergesort

Grundlegende Idee

Es wird eine Zahlenfolge eingegeben.

1. Teile die Zahlenfolge in zwei Hälften
2. Sortiere die beiden Hälften rekursiv
3. Füge die beiden Hälften zusammen

Mergesort

$\textsc{Merge-Sort}(A, p = 1, r = A.length)$

```
1  if p < r
2        q = ⌊(p + r)/2⌋
3        Merge-Sort(A, p, q)
4        Merge-Sort(A, q + 1, r)
5        Merge(A, p, q, r)
```

Mergesort

MERGE(A, p, q, r)

```
 1  n_1 = q − p + 1; n_2 = r − q
 2  seien L[1..n_1 + 1] und R[1..n_2 + 1] zwei neue Felder
 3  for i = 1 to n_1
 4      L[i] = A[p + i − 1]
 5  for j = 1 to n_2
 6      R[j] = A[q + j]
 7  L[n_1 + 1] = ∞
 8  R[n_2 + 1] = ∞
 9  i = j = 1
10  for k = p to r
11      if L[i] ≤ R[j]
12          A[k] = L[i]
13          i = i + 1
14      else A[k] = R[j]
15          j = j + 1
```

Quicksort

Grundlegende Idee

Es wird eine Zahlenfolge eingegeben.

1. Wähle ein Pivot-Element p
2. Partitioniere die Folge in zwei Teile $a = \langle x : x < p \rangle$, $b = \langle x : x > p \rangle$
3. Sortiere a und b rekursiv
4. Füge die sortierten Felder zusammen

Quicksort

$\textsc{Quicksort}(A, p = 1, r = A.length)$

```
1  if p < r
2        q = Partition(A, p, r)
3        Quicksort(A, p, q − 1)
4        Quicksort(A, q + 1, r)
```

$\textsc{Partition}(A, p, r)$

```
1  x = A[r]
2  i = p − 1
3  for j = p to r − 1
4        if A[j] ≤ x
5              i = i + 1
6              vertausche A[i] mit A[j]
7  vertausche A[i + 1] mit A[r]
8  return i + 1
```

Übersicht

Achtung

Das Tutorium nächste Woche (Ostermontag) entfällt!