

# ***Algorithmik***

Sebastian Iwanowski  
FH Wedel

3. Vorlesungswoche

# Algorithmik 3

## Das Auswahlproblem (Order Statistics)

**SELECT** ( $k, A$ ): **Element**

Gesucht wird das  $k$ -te Element eines  $n$ -elementigen unsortierten Arrays  $A$ , also das Element  $x \in A$  mit der Eigenschaft, dass  $k$  Elemente aus  $A$  kleiner oder gleich sind.

### Einfache Lösungen

1. Sortiere alles und bestimme dann das  $k$ -te Element. Laufzeit  $\Theta(n \log n)$   
w.c. und a.c.
2.  $k = 1$  oder  $k = n$ : Laufzeit  $\Theta(n)$  w.c. und a.c.  
Einfacher Durchlauf mit Merken des Minimums bzw. Maximums

### Referenzen zum Nacharbeiten und Vertiefen:

Alt, S. 23

Cormen, Kap. 9.1

# Algorithmik 3

## Das Auswahlproblem (Order Statistics)

**SELECT** ( $k, A$ ): **Element**

Gesucht wird das  $k$ -te Element eines  $n$ -elementigen unsortierten Arrays  $A$ , also das Element  $x \in A$  mit der Eigenschaft, dass  $k$  Elemente aus  $A$  kleiner oder gleich sind.

**Randomisierter Algorithmus**      Laufzeit  $\Theta(n^2)$  w.c.,  $\Theta(n)$  a.c.

1. Wähle zufällig eine Position  $j$  aus  $\{1, \dots, n\}$  und bestimme Element  $a = A[j]$ .
2. Vertausche die Elemente aus  $A$  mit Quicksort-Partition derart, dass  $a$  an der richtigen Position steht und alle kleineren links von  $a$  und alle größeren rechts von  $a$ .  
Teile das Array  $A$  in drei hintereinanderliegende Teile  $A_{<}$ ,  $A_{=}$  und  $A_{>}$  auf.
3.  $|A_{<}| < k \leq n - |A_{>}| \Rightarrow$  return  $a$   
 $k \leq |A_{<}| \Rightarrow$  return **SELECT** ( $k, A_{<}$ )  
 $k > |A_{>}| \Rightarrow$  return **SELECT** ( $k - (n - |A_{>}|), A_{>}$ )

**Referenzen zum Nacharbeiten und Vertiefen:**

Alt, S. 23 – 26

Cormen, Kap. 9.2

# Algorithmik 3

## Das Auswahlproblem (Order Statistics)

**SELECT (k, A): Element**

Gesucht wird das k-te Element eines n-elementigen unsortierten Arrays A, also das Element  $x \in A$  mit der Eigenschaft, dass k Elemente aus A kleiner oder gleich sind.

**Deterministischer Algorithmus** Laufzeit  $\Theta(n)$  w.c. und a.c.

1. Teile A in  $\lceil n/5 \rceil$  Teilfolgen der Länge 5 auf.
2. Sortiere jede Teilfolge i und bestimme dann den Median  $x_i$ .
3. Bestimme den Median x der Mediane durch  $\text{SELECT}(\lfloor n/2 \rfloor, \{x_1, \dots, x_{\lceil n/5 \rceil}\})$
4. Vertausche die Elemente aus A mit Quicksort-Partition derart, dass x an der richtigen Position steht und alle kleineren links von x und alle größeren rechts von x.  
Teile das Array A in drei hintereinanderliegende Teile  $A_{<}$ ,  $A_{=}$  und  $A_{>}$  auf.
5.  $|A_{<}| < k \leq n - |A_{>}| \Rightarrow$  return a  
 $k \leq |A_{<}| \Rightarrow$  return  $\text{SELECT}(k, A_{<})$   
 $k > |A_{>}| \Rightarrow$  return  $\text{SELECT}(k - (n - |A_{>}|), A_{>})$

### Referenzen zum Nacharbeiten und Vertiefen:

Alt, S. 27 – 29

Cormen, Kap. 9.3

# Algorithmik 3

## Suchen in Sortierten Feldern

**SELECT (k, A): Element**

Vorteil gegenüber dynamischen Datenstrukturen: weniger Speicherplatz

- |                            |   |
|----------------------------|---|
| 1. Binärsuche              | Laufzeit $\Theta(\log n)$ w.c. und a.c.                   |
| 2. Interpolationssuche     | Laufzeit $\Theta(n)$ w.c. und $\Theta(\log(\log n))$ a.c. |
| 3. Quadratische Binärsuche | Laufzeit $\Theta(n)$ w.c. und $\Theta(\log(\log n))$ a.c. |

Hausaufgabe (zum 08.05.):

1. Rekursionsgleichung für Binärsuche aufstellen und Laufzeit beweisen
2. Rekursionsgleichung für quadratische Binärsuche im Skript erarbeiten und Laufzeit beweisen

**Referenzen zum Nacharbeiten und Vertiefen:**

Alt, S. 30 – 35