Algorithmen und Datenstrukturen

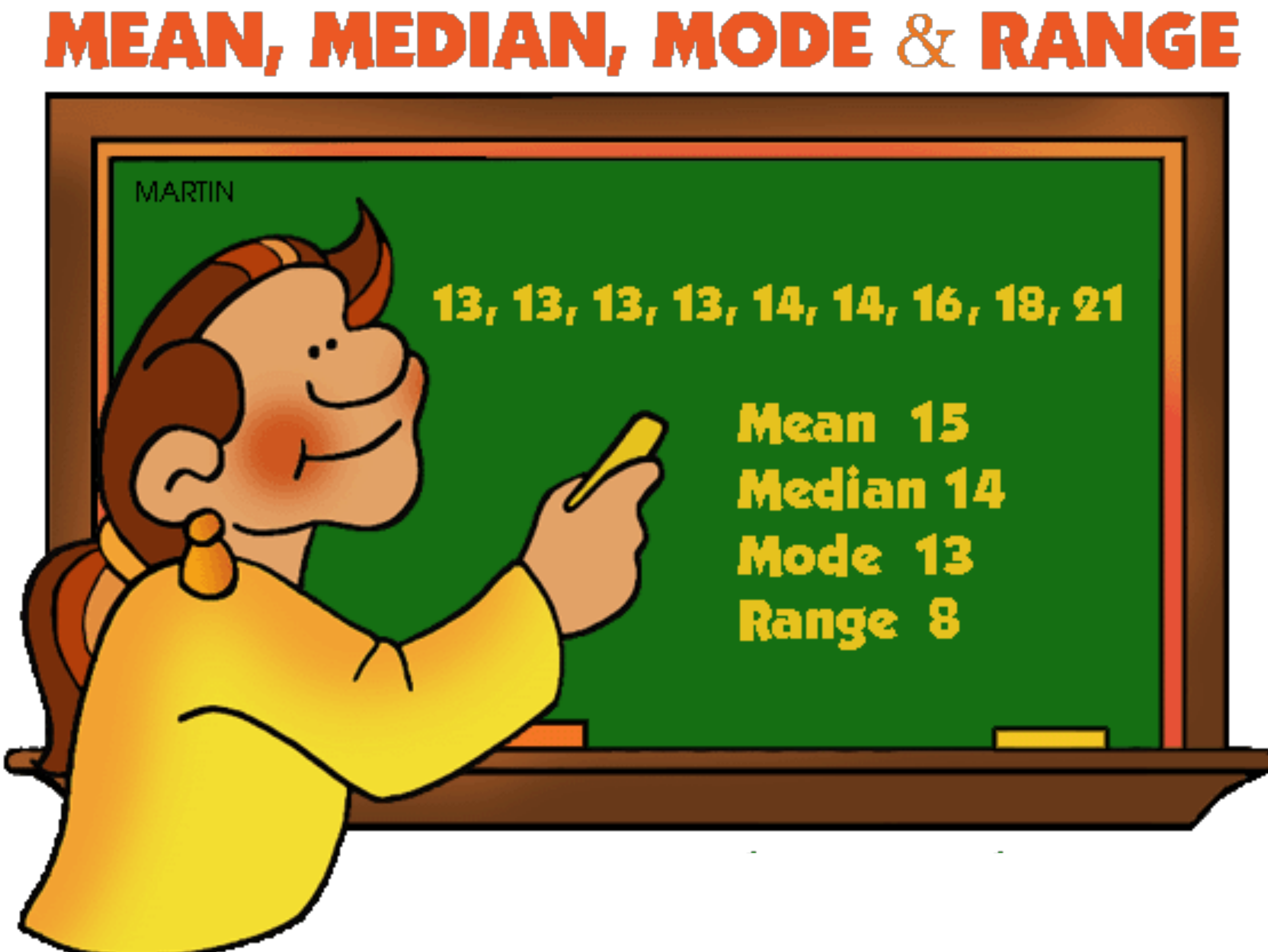
Prof. Dr. Ralf Möller

Universität zu Lübeck

Institut für Informationssysteme

Magnus Bender und Malte Luttermann
(Übungen)

sowie viele Tutoren


MEAN, MEDIAN, MODE & RANGE
MARTIN
13, 13, 13, 13, 14, 14, 16, 18, 21
Mean 15
Median 14
Mode 13
Range 8

Danksagung

Die nachfolgenden 4 Präsentationen übernommen aus der Vorlesung „Effiziente Algorithmen und Datenstrukturen“ (Kapitel 5: Sortieren und Selektieren) gehalten von Christian Scheideler an der TUM

http://www14.in.tum.de/lehre/2008WS/ea/index.html.de

Selektion

Problem: Finde k-kleinstes Element in einer Folge von n Elementen

Lösung: Sortiere Elemente (z.B. Mergesort), gib k-tes Element aus! Zeit $O(n \log n)$

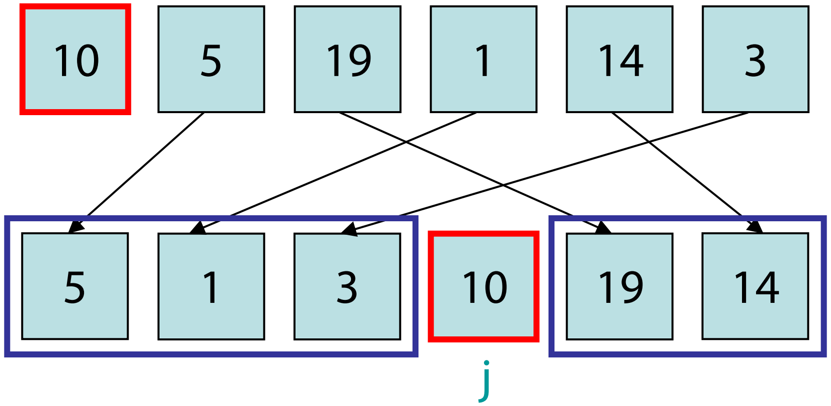
Geht das auch schneller??

Ganz sicher für $k=1$ (min) und $k=n$ (max)

Ausschluss von $n-1$ Elementen nötig, also $O(n)$

Selektion

Ansatz: Verfahre ähnlich zu Quicksort



- j: Position des Pivotelements nach Partitionierung
- k<j: mach mit linker Teilfolge weiter
- k>j: mach mit rechter Teilfolge weiter

Selektion

```
function quickselect(A, l, r, k)
  # A[l..r]: Restfeld, k: k-kleinstes Element, l≤k≤r
  if r == l return A[l] end
  z = rand(l:r)
  A[z], A[r] = A[r], A[z] # tausche A[z] und A[r]
  v = A[r]; i = l; j = r
  while true # ordne Elemente in [l, r - 1] nach Pivot v
    while A[i] < v   i += 1 end
    while A[j] >= v && j != l   j -= 1 end
    if i < j
      A[i], A[j] = A[j], A[i]
    end
    if j <= i   break end
  end
  A[i], A[r] = A[r], A[i]
  if k < i return quickselect(A, l, i - 1, k)
  elseif k > i return quickselect(A, i + 1, r, k)
  else return A[k]
  end
end
```

Quickselect: Analyse

- Aufwand $T(n)$: erwartete Anzahl Vergleiche

Behauptung: $T(n) \in O(n)$

Begründung:

- Pivot ist gut: keine der Teilfolgen länger als $n \cdot 2/3$
- Sei p = Anteil der Fälle, in denen gilt: Pivot ist gut

- $p=1/3$
- Pivot gut: Restaufwand $\leq T(2n/3)$
- Pivot schlecht: Restaufwand $\leq T(n)$

Quickselect: Analyse

$$T(n) \leq cn + p \cdot T(n \cdot 2/3) + (1-p) \cdot T(n)$$

$$p \cdot T(n) \leq cn + p \cdot T(n \cdot 2/3)$$

$$T(n) \leq cn/p + T(n \cdot 2/3)$$

$$\leq cn/p + c \cdot (n \cdot 2/3)/p + T(n \cdot (2/3)^2)$$

... wiederholtes Einsetzen

$$\leq (cn/p)(1 + 2/3 + 4/9 + 8/27 + \ldots)$$

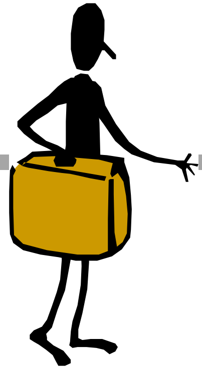
$$\leq \frac{cn}{p} \sum_{i \geq 0} (2/3)^i$$

geometrische Reihe mit $a_0 = 1$:

$$\sum_{k=0}^{\infty} a_0 q^k = \frac{a_0}{1-q}$$

$$\leq \frac{cn}{1/3} \cdot \frac{1}{1-2/3} = 9cn \in O(n)$$

Überblick



- Bisher behandelt:
  - Sortieren durch Vergleichen (vorige Sitzungen)
  - Lineares Sortieren
  - Prioritätswarteschlangen
  - Selektion von Elementen aus einem Feld (z.B. Median)
- Es kommt:
  - Realisierung von Mengen
  - Assoziation von Objekten (über sog. Hashtabellen)