

QuickSort: Sortieren durch Zerlegen

Weiterführende Literatur:

- Algorithmen und Datenstrukturen: Tafelübung 11, WS 2018/19, Seite 55
- Wikipedia-Artikel „Quicksort“
- Saake und Sattler, Algorithmen und Datenstrukturen, Seite 135-139 (PDF 153-157)

Funktionsweise

Listen mit maximal *einem* Element sind *trivialerweise sortiert*. Falls die zu sortierende Liste mehr als ein Element beinhaltet wird ein sogenanntes *Pivot-Element* (vom Französischen *pivot* „Dreh-/Angelpunkt“) ausgewählt. Alle *kleineren* Elemente werden *vor* und alle *größeren hinter* das Pivot-Element verschoben. Der Algorithmus verfährt *rekursiv* mit den beiden Teillisten. Der Algorithmus arbeitet nach dem *Teile-Und-Herrsche-Prinzip*.

einem
trivialerweise sortiert
Pivot-Element
Dreh-/Angelpunkt
kleineren
vor
größeren
hinter
rekursiv
Teile-Und-Herrsche-Prinzip

Eigenschaften

- Laufzeitkomplexität:
 - $\mathcal{O}(n \cdot \log(n))$ (im Best-/Average-Case)
 - $\mathcal{O}(n^2)$ (im Worst-Case)
- in „klassischer“ Variante *instabil*
- durch Rekursion wachsender Aufrufstapel → out-of-place

Minimales Code-Beispiel zum Auswendiglernen

```
28 private int zerlege(int l, int r) {
29     int i, j;
30     int pw = a[(l + r) / 2];
31     i = l - 1;
32     j = r + 1;
33     while (true) {
34         do {
35             i++;
36         } while (a[i] < pw);
37
38         do {
39             j--;
40         } while (a[j] > pw);
41
42         if (i < j) {
43             vertausche(i, j);
44         } else {
45             return j;
46         }
47     }
48 }
```

Code-Beispiel auf Github ansehen: [src/main/java/org/beschlangaul/sortier/QuickMinimal.java](https://github.com/beschlangaul/sortier/QuickMinimal.java)

```
66 private int[] sortiereRekursiv(int l, int r) {
67     int p;
68     if (l < r) {
69         p = zerlege(l, r);
70         sortiereRekursiv(l, p);
71         sortiereRekursiv(p + 1, r);
72     }
73     return zahlen;
```

Code-Beispiel auf Github ansehen: [src/main/java/org/beschlangaul/sortier/QuickMinimal.java](https://github.com/beschlangaul/sortier/QuickMinimal.java)

Implementation nach Saake¹

```
14 /**
15  * Hilfsmethode zum Zerlegen der Folge. Diese Methode heit im Englischen auch
16  * oft "partition".
17  *
18  * @param links Die Index-Nummer der unteren Grenze.
19  * @param rechts Die Index-Nummer der oberen Grenze.
20  *
21  * @return Die endgltige Index-Nummer des Pivot-Elements.
22  */
23 private int zerlege(int links, int rechts) {
24     // Pivot-Element bestimmen
25     int pivotIndex = (links + rechts) / 2;
26     int pivotWert = zahlen[pivotIndex];
27     int pivotIndexEndgltig = links;
28     // Pivot-Element an das Ende verschieben
29     vertausche(pivotIndex, rechts);
30     for (int i = links; i < rechts; i++) {
31         if (zahlen[i] <= pivotWert) {
32             vertausche(pivotIndexEndgltig, i);
```

¹Saake und Sattler, *Algorithmen und Datenstrukturen*, Seite 138 (PDF 156).

```

33         pivotIndexEndgültig++;
34     }
35 }
36 // Pivot-Element an die richtige Position kopieren
37 vertausche(rechts, pivotIndexEndgültig);
38 // neue Pivot-Position zurückgeben
39 return pivotIndexEndgültig;
40 }
41
42 /**
43  * Hilfsmethode zum rekursiven Sortieren
44  *
45  * @param links Die Index-Nummer der unteren Grenze.
46  * @param rechts Die Index-Nummer der oberen Grenze.
47  */
48 private void sortiereRekursiv(int links, int rechts) {
49
50     if (rechts > links) {
51         // Feld zerlegen
52         int pivotIndexEndgültig = zerlege(links, rechts);
53         // und zerlegen sortieren
54         sortiereRekursiv(links, pivotIndexEndgültig - 1);
55         sortiereRekursiv(pivotIndexEndgültig + 1, rechts);
56     }
57 }

```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/sortier/QuickSaake.java](https://github.com/bschlangaul/sortier/QuickSaake.java)

Weitere Implementation

```
14     * verschoben wird.
15     */
16     public class QuickHorare extends Sortieralgorithmus {
17
18         /**
19          * Zerlege das Zahlen-Feld.
20          *
21          * @param links Die Index-Nummer ab dem das Zahlen-Feld zerlegt werden soll.
22          * @param rechts Die Index-Nummer bis zu dem das Zahlen-Feld zerlegt werden
23          * soll.
24          *
25          * @return Die Index-Nummer, an dem das Feld zerlegt werden soll.
26          */
27         private int zerlege(int links, int rechts) {
28             int i, j;
29             int pivotWert = zahlen[(links + rechts) / 2];
30             i = links - 1;
31             j = rechts + 1;
32             while (true) {
33                 do {
34                     i++;
35                 } while (zahlen[i] < pivotWert);
36
37                 do {
38                     j--;
39                 } while (zahlen[j] > pivotWert);
40
41                 if (i < j) {
42                     vertausche(i, j);
43                 } else {
44                     return j;
45                 }
46             }
47         }
48
49         /**
50          * Sortiere ein Zahlen-Feld mit Hilfe des Quicksort-Algorithmus.
51          *
52          * @param links Die Index-Nummer ab dem das Zahlen-Feld sortiert werden soll.
53          * @param rechts Die Index-Nummer bis zu dem das Zahlen-Feld sortiert werden
54          * soll.
55          *
56          * @return Das sortierte Zahlenfeld.
57          */
```

Code-Beispiel auf Github ansehen: [src/main/java/org/bschlangaul/sortier/QuickHorare.java](https://github.com/bschlangaul/sortier/QuickHorare.java)

Im Gegensatz zu der Implementation von Saake wird hier der Pivot-Wert nicht an den oberen Rand und dann wieder zurück kopiert.

8	2	1	5	9	7	3
8	2	1	5	9	7	3
2	1	3	8	5	9	7

2	1	3	8	5	9	7
1	2	3	5	7	8	9
1	2	3	5	7	8	9
1	2	3	5	7	8	9
1	2	3	5	7	8	9
1	2	3	5	7	8	9
1	2	3	5	7	8	9

Literatur

- [1] *Algorithmen und Datenstrukturen: Tafelübung 11, WS 2018/19.* https://www.studon.fau.de/file2567217_download.html. FAU: Lehrstuhl für Informatik 2 (Programmiersysteme).
- [2] Gunter Saake und Kai-Uwe Sattler. *Algorithmen und Datenstrukturen. Eine Einführung in Java.* 2014.
- [3] *Wikipedia-Artikel „Quicksort“.* <https://de.wikipedia.org/wiki/Quicksort>.