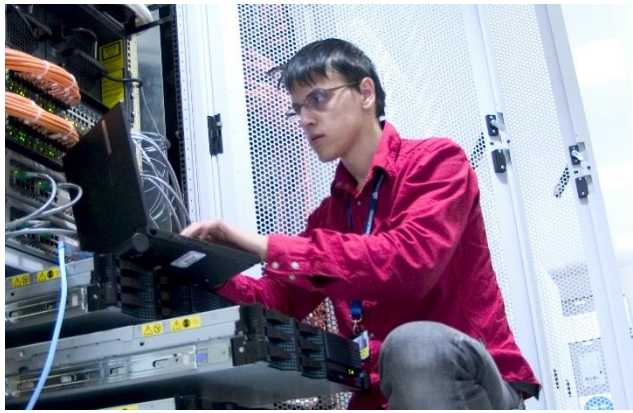
EMIL-VON-BEHRING GYMNASIUM

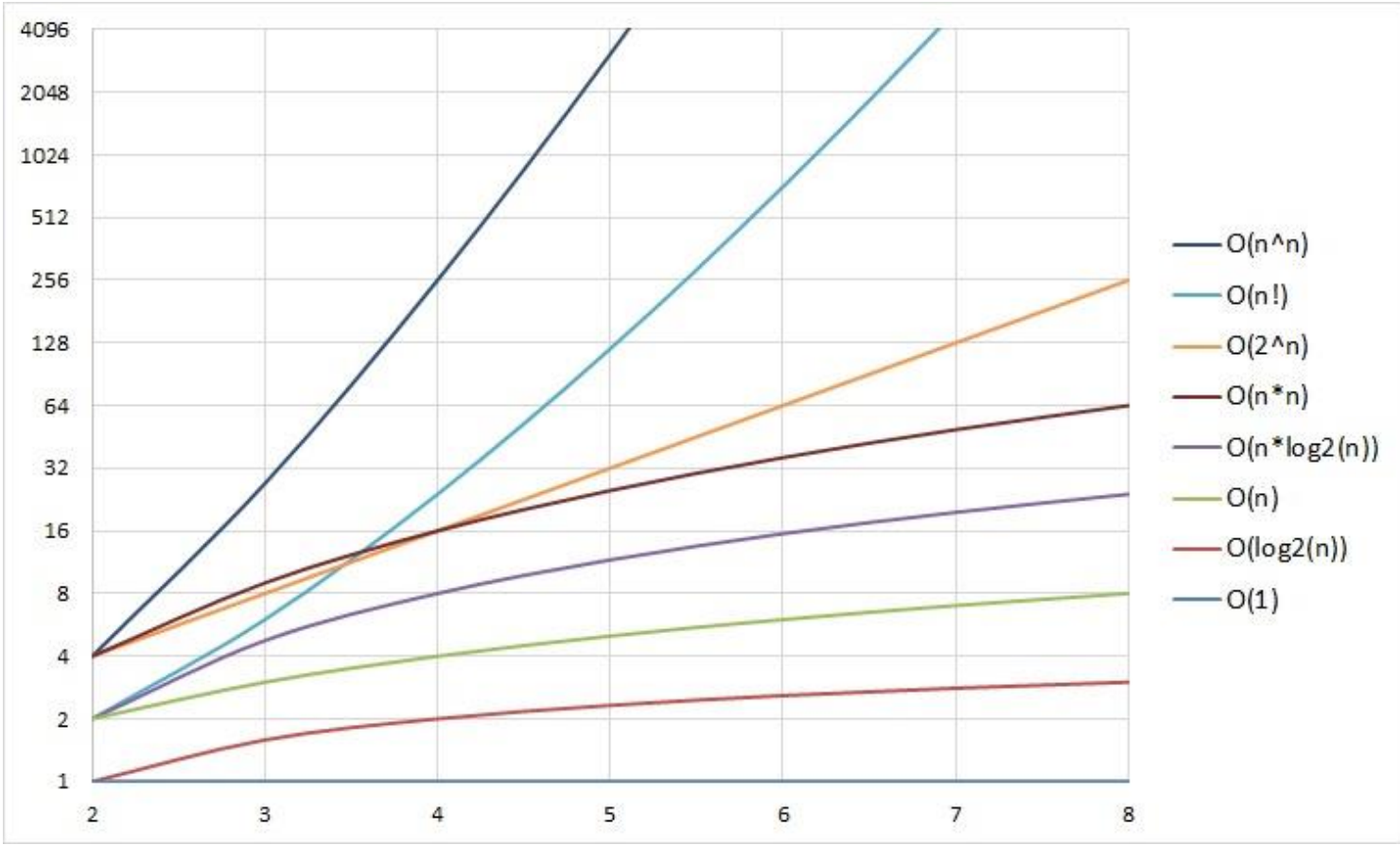
Experimentelle Abschätzung des Laufzeitaufwands von Sortieralgorithmen



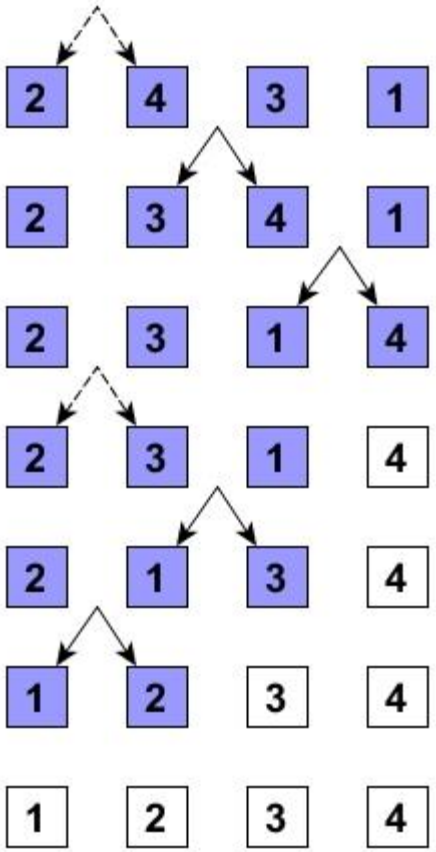
Grafiken © shutterstock.com

Bubblesort, Mergesort und Quicksort

- Bearbeite alle Sortieralgorithmen gemäß Arbeitsblatt



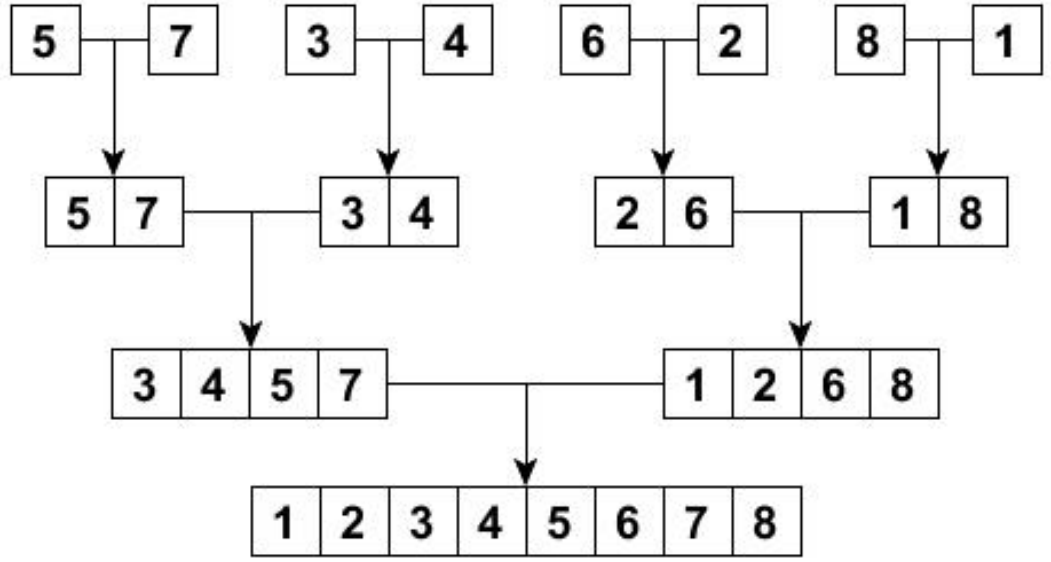
Aufwandsabschätzung von Bubblesort



- Im ersten Durchlauf werden $n - 1$ Vergleiche durchgeführt.
- Im zweiten Durchlauf werden $n - 2$ Vergleiche durchgeführt
- …
- Insgesamt werden somit
  $1 + 2 + \cdots + n - 2 + n - 1$
  Vergleiche durchgeführt.

$$\sum_{i=1}^{n-1} i = \frac{n \cdot (n-1)}{2} = \frac{n^2 - n}{2} \quad \Rightarrow \quad O(n^2)$$

Aufwandsabschätzung von Mergesort

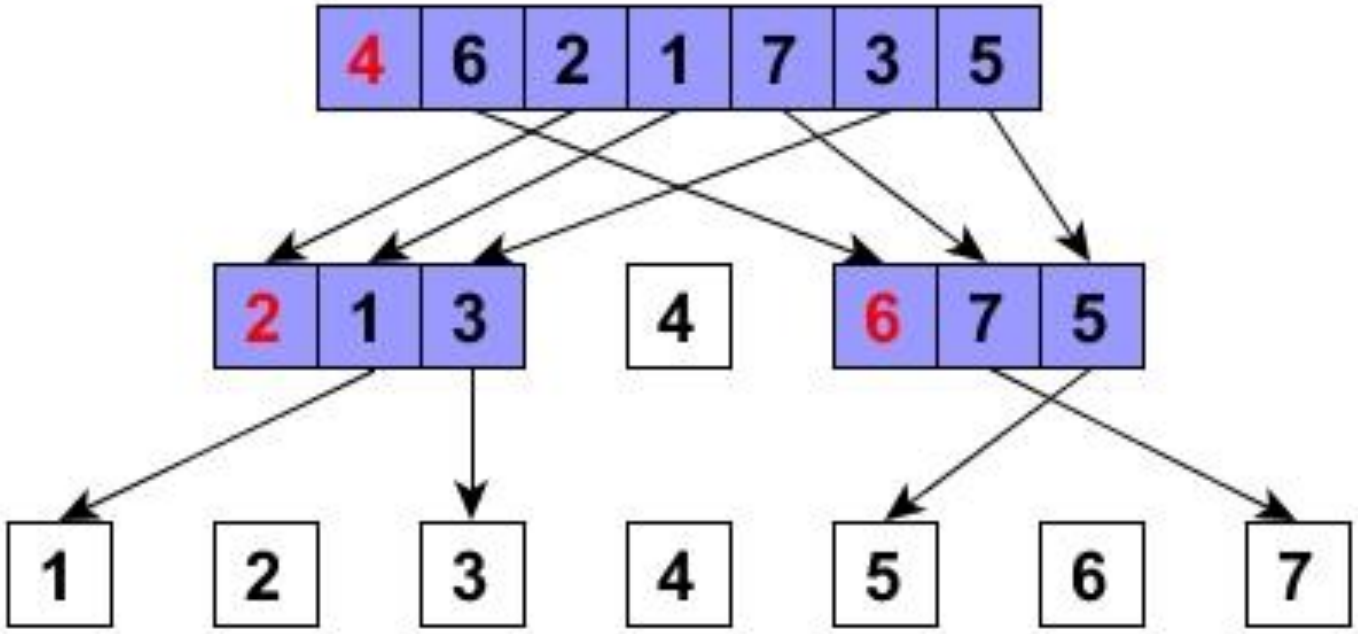


Anzahl Listen	Länge der Listen
8	1
4	2
2	4
1	8

- Pro Zeile: n zu sortierende Elemente
- $\log_2 n$ Durchläufe sind nötig
- $\Rightarrow O(n \cdot \log_2 n)$

Aufwandsabschätzung von Quicksort

- Wenn das Pivotelement den zu sortierenden Ausschnitt genau halbiert (also die Hälfte der Werte kleiner ist als das Pivotelement und die andere Hälfte der Werte größer ist als das Pivotelement), dann hat Quicksort den Aufwand $O(n \cdot \log_2 n)$.



Zusatzaufgaben

Zusatzaufgabe 1

Begründe, warum es zulässig ist, nur die Anzahl der Wiegungen zur Abschätzung des Laufzeitaufwands heranzuziehen.

- Der Vergleich von zwei Elementen liegt in O(1) und kann daher bei der Betrachtung im O-Kalkül vernachlässigt werden.

Zusatzaufgaben

Bubblesort: Zusatzaufgabe 2

Überlege, wie eine Startreihenfolge angeordnet sein muss, sodass der Algorithmus nach möglichst wenigen/vielen Schritten beendet wird. Bestimme, in welcher Laufzeitkategorie der Algorithmus mit diesen (speziellen) Startreihenfolgen liegen würde.

- Da beim ersten Durchlauf keine Elemente miteinander vertauscht werden, beendet sich der Algorithmus. In diesem ersten Durchlauf werden $n-1$ Vergleiche durchgeführt. Somit liegt der Algorithmus bei diesem Spezialfall in $O(n)$. Gegensätzlich sortiert -> man braucht eben alle $\frac{n^2-n}{2}$ Vergleiche.
- -> average case: $O(n^2)$ best case: $O(n)$ worst case: $O(n^2)$

Zusatzaufgaben

Mergesort: Zusatzaufgabe 2

Überlege, wie eine Startreihenfolge angeordnet sein muss, sodass der Algorithmus nach möglichst wenigen/vielen Schritten beendet wird. Bestimme, in welcher Laufzeitkategorie der Algorithmus mit diesen (speziellen) Startreihenfolgen liegen würde.

- Nein, der Algorithmus benötigt immer gleich viele Schritte bis alle Elemente sortiert sind. Aufgrund einer fehlenden Fallunterscheidung wird immer auf dieselbe Weise sortiert (sogar dann, wenn die Startreihenfolge schon richtig sortiert ist).
- -> average case/best case/worst case: $O(n \cdot \log_2 n)$

Zusatzaufgaben

Quicksort: Zusatzaufgabe 2

Überlege, wie eine Startreihenfolge angeordnet sein muss, sodass der Algorithmus nach möglichst wenigen/vielen Schritten beendet wird. Bestimme, in welcher Laufzeitkategorie der Algorithmus mit diesen (speziellen) Startreihenfolgen liegen würde.

- Die Startreihenfolge muss bereits aufsteigend/absteigend sortiert sein. Somit ist das gewählte Pivotelement immer das Kleinste/Größte der verbleibenden Elemente und bei der Sortierung bleibt die andere Teilliste leer. So wird dann jedes Element mit jedem anderen verglichen. -> worst case: $O(n^2)$
- best case: Pivotelement ist immer der Median $O(n \cdot \log_2 n)$
- average case: $O(n \cdot \log_2 n)$

Welcher Algorithmus ist nun der Beste?

- Bubblesort ist nicht der Beste, da dieser ja in einer höheren Laufzeitklasse liegt.
- Es scheint so, dass Mergesort besser sei als Quicksort, weil dieser immer $n \cdot \log_2 n$ Schritte benötigt, während der Quicksort-Algorithmus im schlimmsten Fall sogar n^2-viele Schritte.
- Aber im Durchschnitt (bei einer zufälligen durchschnittlichen Wahl des Pivotelements) kann man durch praktisches Austesten zeigen, dass der Quicksort-Algorithmus im Durchschnitt schneller (= weniger Berechnungsschritte) ist als Mergesort.