

3AA

Sortieren Direkte Sortierverfahren

30.01.06

Prof. Dr. Wolfgang P. Kowalk
Universität Oldenburg

WS 2005/2006

Laufzeitkomplexität

- Hängt die Bearbeitungszeit $Z(N)$ eines Problems von Komplexitätsparameter N ab, so dass für ein $N_0 > 0$ und $c, C > 0$ gilt:
 - $c \cdot f(N) \leq Z(N) \leq C \cdot g(N)$ für alle $N > N_0, C > 0, c > 0,$
- so wird $Z(N)$ bezeichnet als
 - „ $Z(N)$ ist nach unten durch $o(f(N))$ und nach oben durch $O(g(N))$ beschränkt“
 - $O(g(N))$ heißt „Von der Ordnung $g(N)$ “

Laufzeitkomplexität

- Spezielle Bezeichnungen
 - $O(1)$ oder **konstant**, falls $g(N) = 1$ eine Konstante ist;
 - $O(N)$ oder **linear**, falls $g(N) = N$;
 - $O(N^2)$ oder **quadratisch**, falls $g(N) = N^2$;
 - **polynomial**, falls $g(N)$ ein Polynom in N ist;
 - $O(\log(N))$ oder **logarithmisch**, falls $g(N) = \log(N)$;
 - $O(a^N)$ oder exponentiell, falls $g(N) = a^N$ ist für eine Konstante $a > 1$;
 - $(N \cdot \log(N))$ oder „ N mal $\log N$ “, falls $g(N) = N \cdot \log(N)$.
- Komplexität eines Problems
 - Komplexität des „schnellsten“ Algorithmus, der dieses Problem löst.

Sortieren – Grundbegriffe

- Eine Folge von N Zahlen lässt sich in einem Feld abspeichern:

- **integer array** **Feld** [1..N];

- Unsortiertes Feld

8	15	3	6	5	2	16	13	9	4	1	12	11	14	10	7
---	----	---	---	---	---	----	----	---	---	---	----	----	----	----	---

- Sortiertes Feld

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
---	---	---	---	---	---	---	---	---	----	----	----	----	----	----	----

- **Permutieren**

Vertauschen zweier Elemente des Feldes

8	2	3	6	5	15	16	13	9	4	1	12	11	14	10	7
---	---	---	---	---	----	----	----	---	---	---	----	----	----	----	---

Sortieren – Grundbegriffe

- **Stabiles Sortierverfahren**

- $i < j \wedge \text{Feld}[i] = \text{Feld}[j] \Rightarrow i_s < j_s$

- Ordnung gleicher Feldelemente bleibt erhalten.

- Wichtig für

- Sortierung bei verschiedenen Schlüsseln
(Vorname, Nachname)

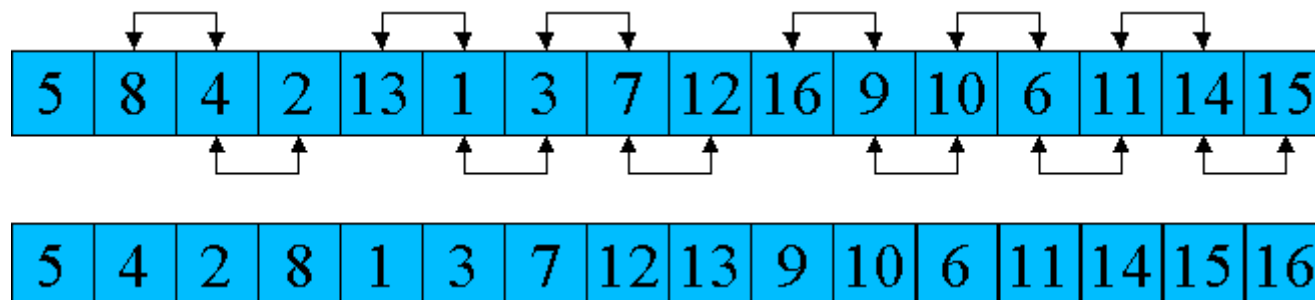
-

- **Ordnungsverträgliches Sortierverfahren**

- Laufzeit eines Sortierverfahrens wächst monoton mit Inversionszahl jeder Permutation
 - Effizientere Verfahren sind häufig nicht oder nur teilweise ordnungsverträglich

Direkte Sortierverfahren

- Sortieren durch Vertauschen
 - Bubblesort
- Vertausche benachbarte Element, solange noch welche vertauscht sind
- Verschiedene Verbesserungsmöglichkeiten, aber
 - Bubblesort 'schlechtestes' Sortierverfahren
 - nicht einfacher zu implementieren als andere Verfahren
 - unprofessionell



Direkte Sortierverfahren

```
1      // sortieren durch Vertauschen
2
3      adr ende := con feld0 // Feld [0]
4 FuegHinzu adr ende += con 1 // Feld [0,...,ende]
5      if val ende >= con feldEnde then goto con Fertig
6      adr vergleich := val ende // vergleich = ende
7      // vertausche bis an richtiger Stelle
8 Vertausch if val vergleich <= con feld0 then goto con FuegHinzu
9      adr v1 := val vergleich - con 1 // v1 = i-1
10     // prüfe, ob an richtiger Stelle
11     if val val v1 <= val val vergleich then goto con FuegHinzu
12     adr wert1 := val val v1 // vertausche w<=[v1]
13     adr val v1 := val val vergleich // [v1] = [vergleich]
14     adr val vergleich := val wert1 // [vergleich] = w
15     adr vergleich -= con 1 // vergleich--
16     goto con Vertausch
17 Fertig stop

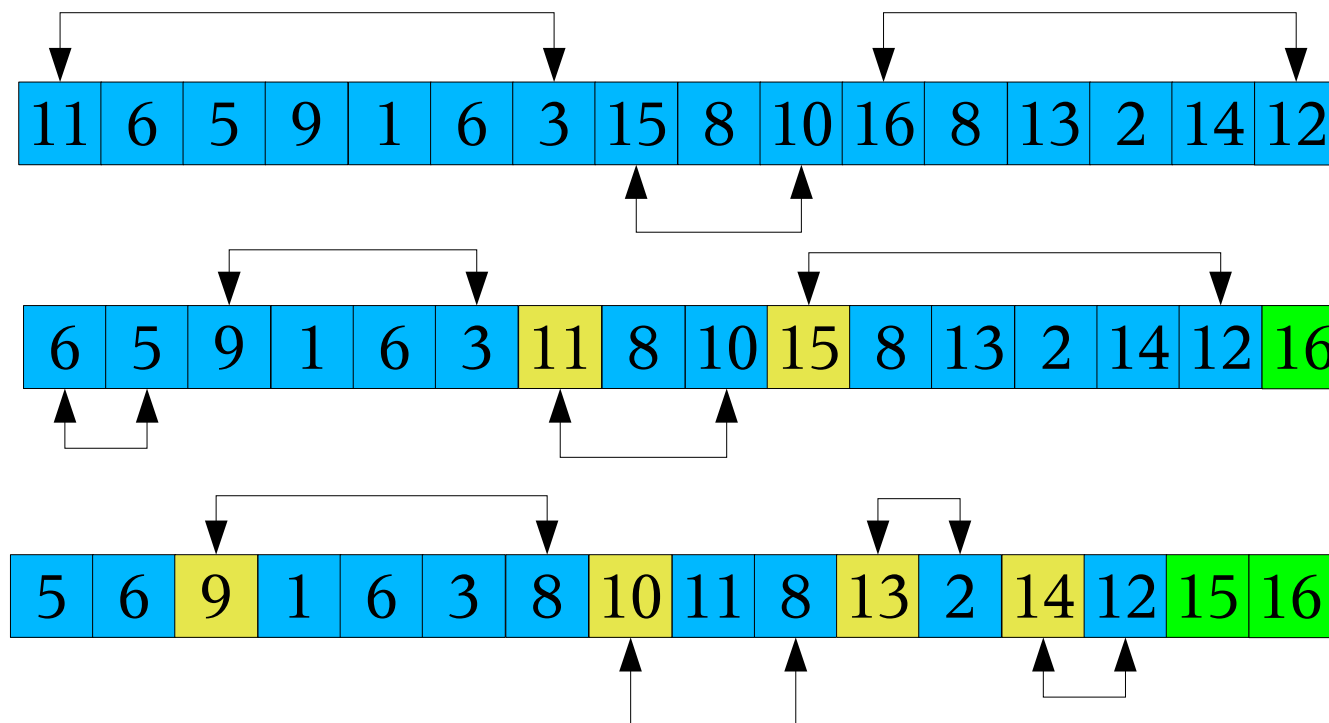
26 feld0 con 12
27 feld1 con 4
...
47 feldEnde
```

Direkte Sortierverfahren

- Die Laufzeit von Bubblesort ist quadratisch
 - Vergleiche in jeder Zeile $(N-1)$ Mal
 - Nach jedem Durchlauf liegt größtes Element an richtiger Stelle
 - Nach $N-1$ Durchläufen stehen alle Elemente an richtiger Stelle
 - Laufzeit: $(N-1)^2$ Vergleiche, ca. $(N-1)^2/2$ Vertauschungen
 - Laufzeitkomplexität: $O(N^2)$.

Direkte Sortierverfahren

- Verbesserungen der Laufzeit von Bubblesort
 - Verkleinere obere Grenze:
 $N-1 + N-2 + N-3 + \dots + 2 + 1 = N \cdot (N-1) / 2$ Vergleiche

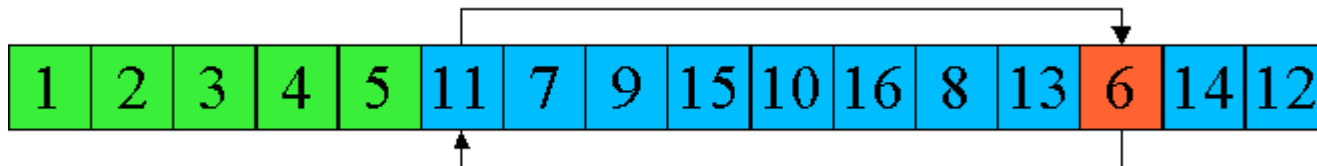


Direkte Sortierverfahren

- Verbesserungen der Laufzeit von Bubblesort
 - Beende Durchlauf, wenn alle Elemente sortiert
 - meist keine Verbesserung, da zusätzlicher Overhead
- Weitere Verbesserungen
 - Statt Vertauschen: Suche größtes Element und bringe es nach oben → **Sortieren durch Auswählen**
 - Vergrößere Menge um jeweils ein Element; vertausche dieses an richtige Stelle → **Sortieren durch Einfügen**
- Alle 'direkten' Sortierverfahren haben quadratische Laufzeitkomplexität!

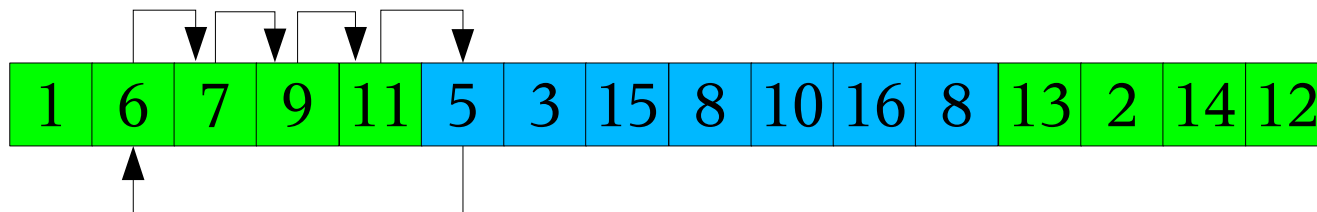
Direkte Sortierverfahren

- Sortieren durch Auswählen
 - Vertausche kleinstes Element an den Anfang der Liste
 - i. Suche kleinstes Element
 - ii. Vertausche kleinstes Element mit erstem Element
 - iii. Verkleinere Liste um richtig positioniertes Element
 - iv. Verfahre bei i. fort, bis nur noch ein Element übrig



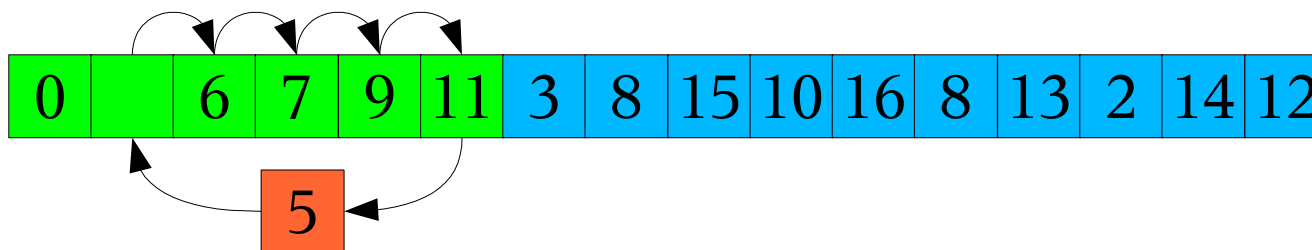
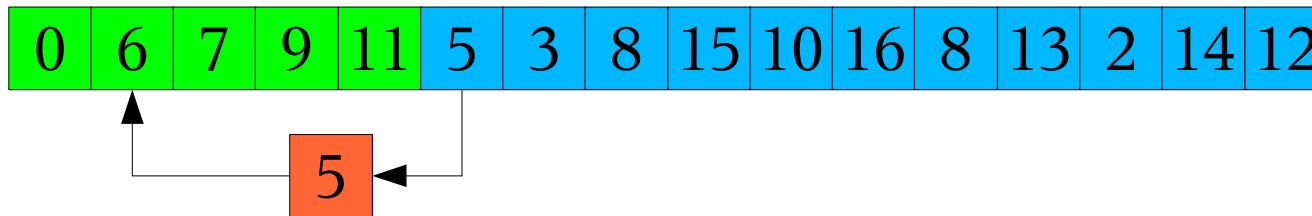
Direkte Sortierverfahren

- Sortieren durch Einfügen
 - Suche zu jedem Element den richtigen Platz
 - Verschiebe erstes unsortiertes Element nach links, bis das linke Element kleiner oder gleich diesem
 - i. **Sortierte Menge** [1..1] ist sortiert
 - ii. Füge ein Element zur **Sortierten Menge** hinzu
 - iii. Vertausche dieses mit links benachbartem, solange bis linkes Element kleiner oder gleich diesem.



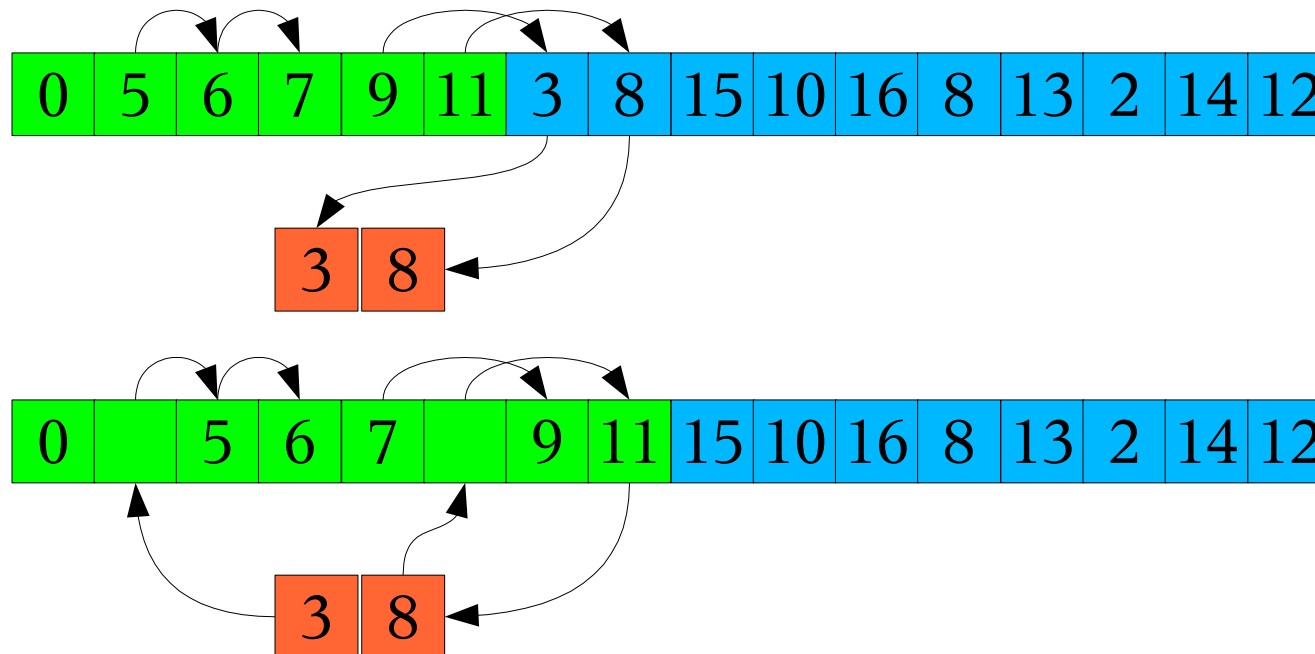
Direkte Sortierverfahren

- Sortieren durch Einfügen
 - Verbesserungsmöglichkeiten
 - Statt Vertauschen, verschiebe, bis kleineres gefunden



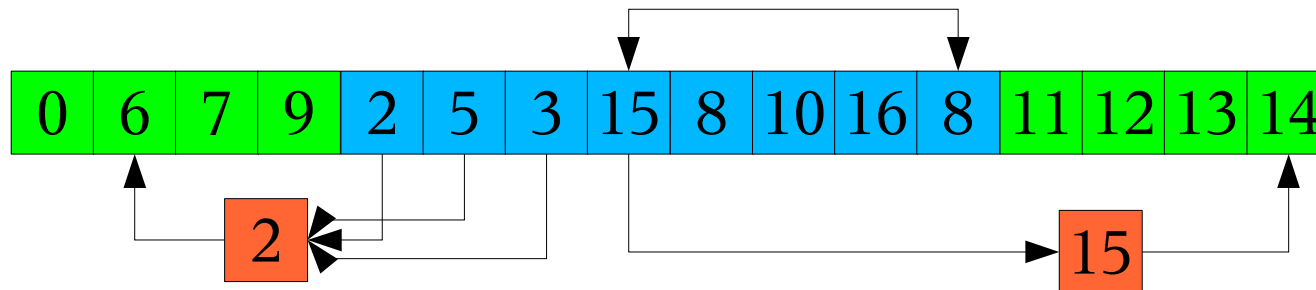
Direkte Sortierverfahren

- Sortieren durch Einfügen
 - Verbesserungsmöglichkeiten
 - Verschiebe gleichzeitig 2 (oder mehrere) Element



Direkte Sortierverfahren

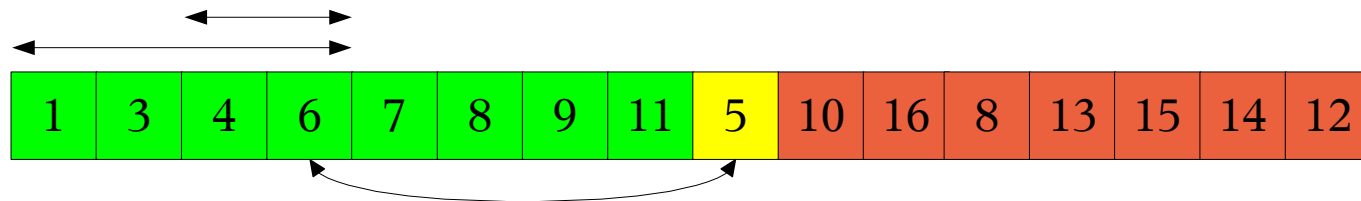
- Sortieren durch Einfügen
 - Verbesserungsmöglichkeiten
 - Bringe kleine Element nach vorne, große nach hinten
 - Pivot-Element (10)



- Quadratische Laufzeit: $O\left(2 \cdot \left(\frac{N}{2}\right)^2\right) = \frac{1}{2} \cdot O(N^2)$

Direkte Sortierverfahren

- Sortieren durch Einfügen
 - Verbesserungsmöglichkeiten
 - Kombiniere diese Verfahren
 - Suche Einfügestelle durch Binärteilung

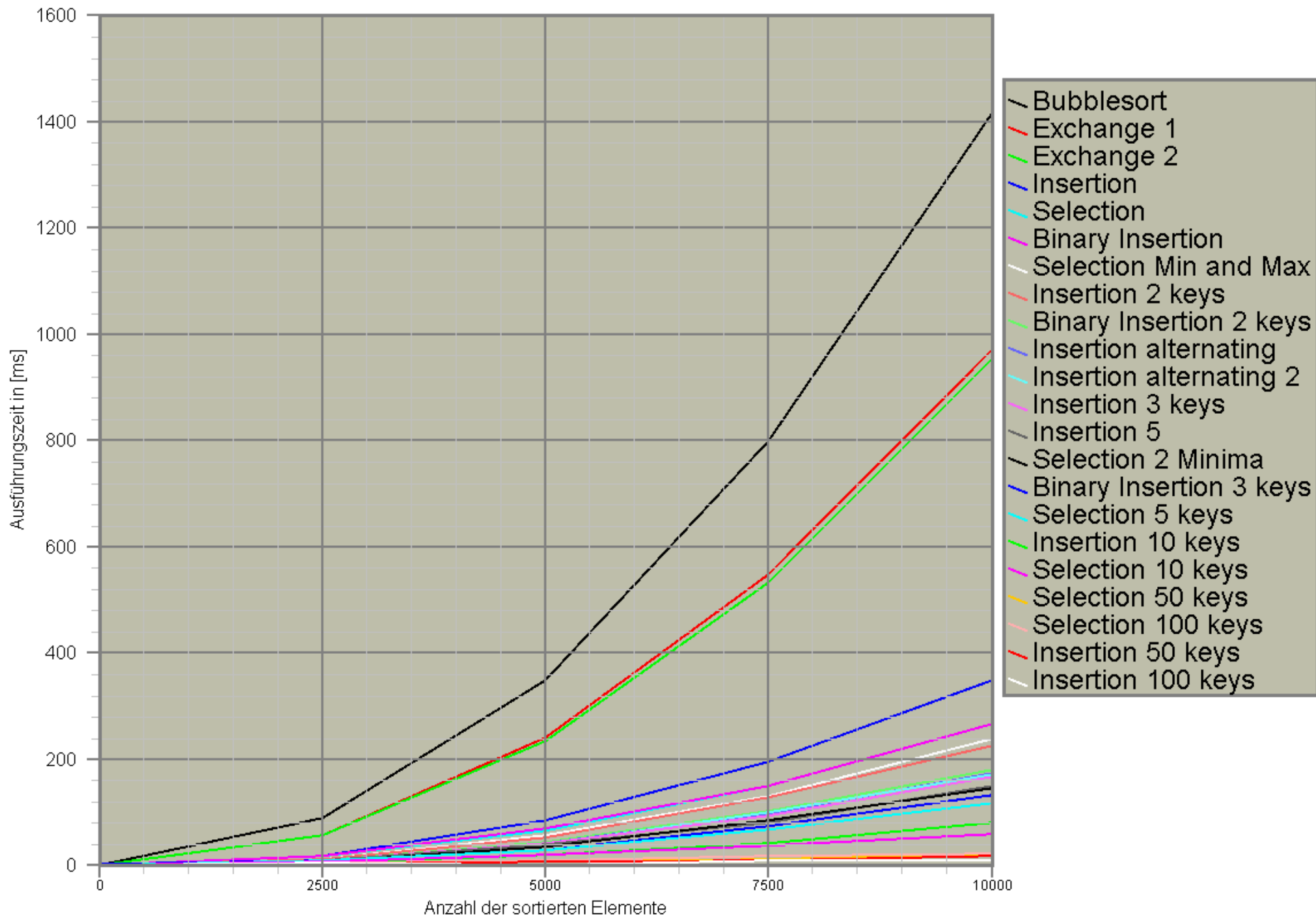


- Verringert Zahl der Vergleiche

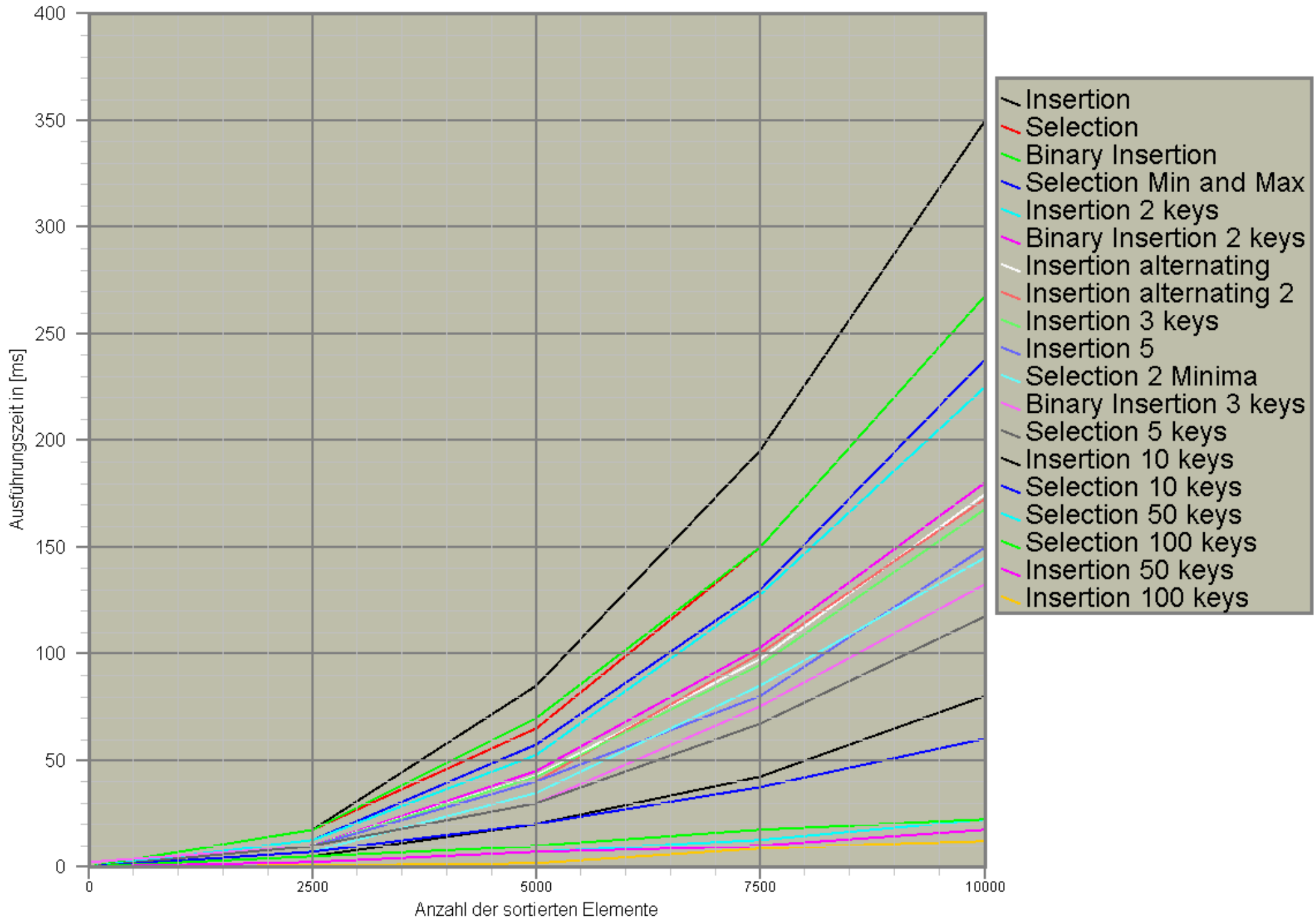
$$\log_2 2 + \log_2 3 + \dots + \log_2 (N-1) = \log_2 (N-1)! < \log_2 ((N-1)^{N-1}) = (N-1) \cdot \log_2 (N-1)$$

- Zahl der Schiebeoperationen bleibt unverändert

Vergleich der Laufzeiten der Sortieralgorithmen



Vergleich der Laufzeiten der Sortieralgorithmen



Direkte Sortierverfahren

- Direktes Sortieren
 - Quadratischer Aufwand häufig zu groß
 - Bessere Verfahren
 - Shellsort
 - Heapsort
 - Quicksort
 - Mergesort
 - Mischen (externes Verfahren)
 - Demnächst in diesem Theater!