Algorithmen und Datenstrukturen

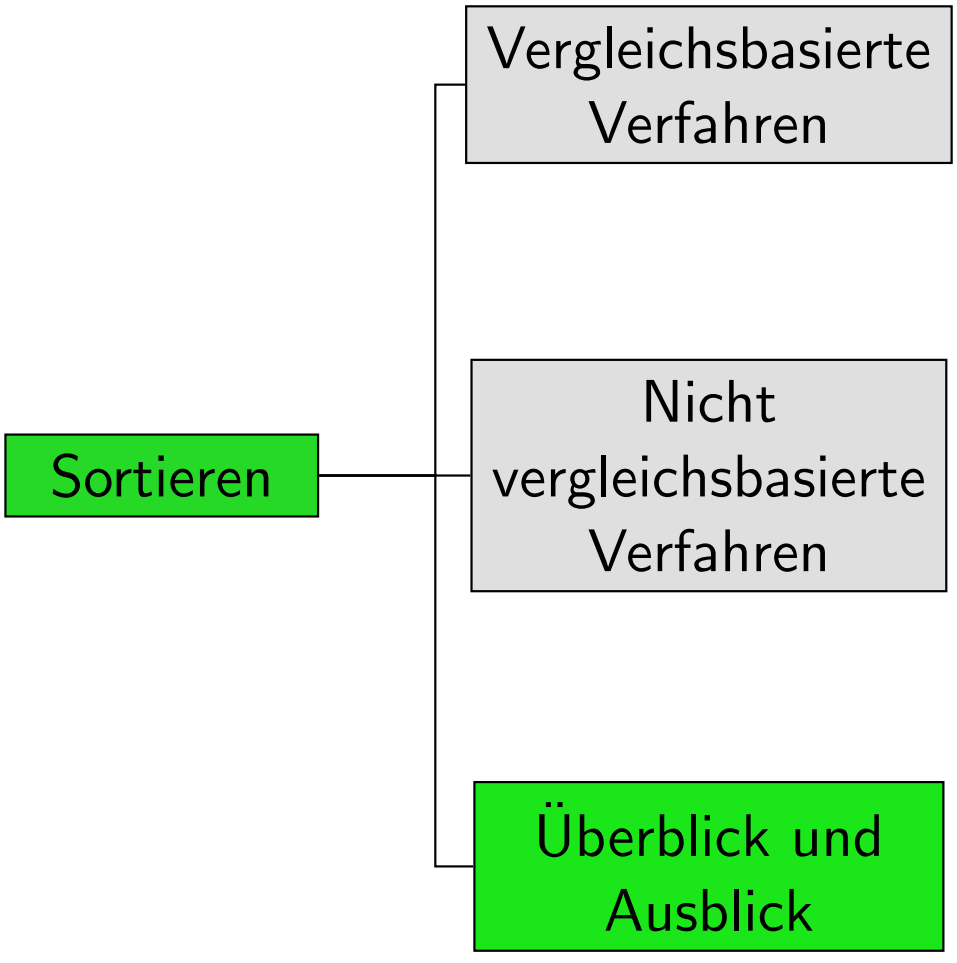
A14. Sortieren: Überblick & Ausblick

Marcel Lüthi and Gabriele Röger

Universität Basel

30. März 2022

Sortierverfahren



Überblick

Vergleichsbasierte Verfahren: Übersicht

Algorithmus	Laufzeit $O(\cdot)$	Speicherbedarf $O(\cdot)$	stabil
	best/avg./worst	best/avg./worst	
Selectionsort	n^2	1	nein
Insertionsort	$n/n^2/n^2$	1	ja
Mergesort	$n \log n$	n	ja
Quicksort	$n \log n/n \log n/n^2$	$\log n/\log n/n$	nein
Heapsort	$n \log n$	1	nein

Vergleichsbasierte Verfahren: Übersicht

Algorithmus	Laufzeit $O(\cdot)$	Speicherbedarf $O(\cdot)$	stabil
	best/avg./worst	best/avg./worst	
Selectionsort	n^2	1	nein
Insertionsort	$n/n^2/n^2$	1	ja
Mergesort	$n \log n$	n	ja
Quicksort	$n \log n/n \log n/n^2$	$\log n/\log n/n$	nein
Heapsort	$n \log n$	1	nein

Sehr schöne Visualisierung der Verfahren unter
`https://www.toptal.com/developers/sorting-algorithms/`

Vergleichsbasierte Verfahren: Bemerkungen

- Insertionsort ist auf kleinen Sequenzen sehr schnell und wird daher zum Beispiel zur Verbesserung von Mergesort und Quicksort für kurze Aufrufe eingesetzt.

Vergleichsbasierte Verfahren: Bemerkungen

- Insertionsort ist auf kleinen Sequenzen sehr schnell und wird daher zum Beispiel zur Verbesserung von Mergesort und Quicksort für kurze Aufrufe eingesetzt.
- Quicksort hat eine sehr kurze (= schnelle) innere Schleife. Mit Randomisierung tritt schlechtester Fall so gut wie nie auf.

Vergleichsbasierte Verfahren: Bemerkungen

- Insertionsort ist auf kleinen Sequenzen sehr schnell und wird daher zum Beispiel zur Verbesserung von Mergesort und Quicksort für kurze Aufrufe eingesetzt.
- Quicksort hat eine sehr kurze (= schnelle) innere Schleife. Mit Randomisierung tritt schlechtester Fall so gut wie nie auf.
- Mergesort ist dafür stabil. Zudem ist der Mergeschritt auch für externes Sortieren relevant
  Wird z.B. gerne bei Datenbankanwendungen eingesetzt.

Vergleichsbasierte Verfahren: Bemerkungen

- Insertionsort ist auf kleinen Sequenzen sehr schnell und wird daher zum Beispiel zur Verbesserung von Mergesort und Quicksort für kurze Aufrufe eingesetzt.
- Quicksort hat eine sehr kurze (= schnelle) innere Schleife. Mit Randomisierung tritt schlechtester Fall so gut wie nie auf.
- Mergesort ist dafür stabil. Zudem ist der Mergeschritt auch für externes Sortieren relevant
  Wird z.B. gerne bei Datenbankanwendungen eingesetzt.
- Heapsort ist in der Praxis etwas langsamer als Mergesort, als in-place-Verfahren aber dennoch interessant
  z.B. für eingebettete Systeme.

Vergleichsbasierte Verfahren: Bemerkungen

- Insertionsort ist auf kleinen Sequenzen sehr schnell und wird daher zum Beispiel zur Verbesserung von Mergesort und Quicksort für kurze Aufrufe eingesetzt.
- Quicksort hat eine sehr kurze (= schnelle) innere Schleife. Mit Randomisierung tritt schlechtester Fall so gut wie nie auf.
- Mergesort ist dafür stabil. Zudem ist der Mergeschritt auch für externes Sortieren relevant
  Wird z.B. gerne bei Datenbankanwendungen eingesetzt.
- Heapsort ist in der Praxis etwas langsamer als Mergesort, als in-place-Verfahren aber dennoch interessant
  z.B. für eingebettete Systeme.
- Gleiche asymptotische Laufzeit bedeutet nicht, dass Verfahren auch gleich lange brauchen (verschiedene Konstanten in $O(\cdot)$).
  Heapsort braucht doppelt so viele Vergleiche wie Mergesort.

Ausblick

Vorsortierte Daten

- Oftmals sind Teilsequenzen der Eingabe bereits vorsortiert.
- Insertionsort profitiert davon direkt.
- Von manchen Verfahren gibt es Varianten,
  die Vorsortierung ausnutzen
  z.B. natürliches 2-Wege-Mergesort.

Viele gleiche Schlüssel

- Tritt in praktischen Anwendungen häufig auf
  z.B. Sortieren von Studierendendaten nach Wohnort
- Von manchen Algorithmen gibt es spezialisierte Varianten
- Zum Beispiel 3-Wege-Partitionierung in Quicksort

$< P$	$= P$	$> P$

Sortieren komplexer Objekte

- Meist will man nicht nur Zahlen,
  sondern komplexe Objekte sortieren.
- Hier wäre es sehr teuer, bei jeder Vertauschung
  die ganzen Objekte zu kopieren.
- Stattdessen: Sortiere Elemente, die nur aus Schlüssel und Zeiger/Referenz auf das tatsächliche Objekt bestehen.

Weniger korrekte Verfahren

vollständiger Comic unter `https://xkcd.com/1185/`
(CC BY-NC 2.5)

Andere Probleme durch Sortieren lösen

k-kleinstes Element

- zum Beispiel Finden des Medians ($k = \lfloor n/2 \rfloor$)
- Verwende Quicksort, aber mache rekursiven Aufruf nur für den relevanten Bereich ($\rightarrow$ Quickselect).

Andere Probleme durch Sortieren lösen

k-kleinstes Element

- zum Beispiel Finden des Medians ($k = \lfloor n/2 \rfloor$)
- Verwende Quicksort, aber mache rekursiven Aufruf nur für den relevanten Bereich ($\rightarrow$ Quickselect).

Duplikate

- Wie viele verschiedene Schlüssel gibt es? Welcher Wert ist am häufigsten? Gibt es doppelte Schlüssel?
- Kann man direkt mit quadratischen Algorithmen beantworten.
- Oder – schlauer – erst sortieren und dann mit einem Durchlauf lösen.

Quiz

Quiz

kahoot.it