h_da
HOCHSCHULE DARMSTADT
UNIVERSITY OF APPLIED SCIENCES

fbi
FACHBEREICH INFORMATIK

Sortieralgorithmen

Tobias Schneider, Fatih Kahraman

26. Mai 2017

Inhaltsverzeichnis

1 Einleitung **1**
- 1.1 Abstract . . . 1
- 1.2 Leser Fangen . . . 1
- 1.3 Problem und Relevanz . . . 1

2 Hauptteil **1**
- 2.1 Grundlagen . . . 1
  - 2.1.1 O-Notation . . . 1
  - 2.1.2 In-Place . . . 1
  - 2.1.3 Stabilität . . . 1
  - 2.1.4 Heap/Stack Größe . . . 2
  - 2.1.5 Testumgebung . . . 2
- 2.2 Sortieralgorithmen . . . 2
  - 2.2.1 Vorstellung . . . 2
  - 2.2.2 Pseudo Code . . . 2
  - 2.2.3 Eigenschaften . . . 2
  - 2.2.4 Testfälle . . . 2
- 2.3 Evaluierung . . . 2
  - 2.3.1 Vergleich der Ergebnisse . . . 2
  - 2.3.2 Relevanz der O-Notation . . . 2
  - 2.3.3 Wie wichtig sind weitere Kriterien . . . 2

3 Schluss **3**
- 3.1 Zusammenfassung und Ausblick . . . 3
  - 3.1.1 Fazit . . . 3
  - 3.1.2 Anwendungstipps . . . 3

4 Literatur Alt **3**

5 Literaturverzeichnis **3**

1 Einleitung

1.1 Abstract

1.2 Leser Fangen

1.3 Problem und Relevanz

2 Hauptteil

2.1 Grundlagen

2.1.1 O-Notation

Mithilfe der O-Notation, auch Landau-Notation gennant, wird eine Laufzeitberechnung anhand der gegebenen Eingabelänge n bestimmt. Dabei wird ein ungefähres Wachstumsverhalten des Algorithmus als mathematische Funktion definiert. Dies geschieht durch Analyse des Quellcodes und wird üblicherweise für drei Fälle durchgeführt:

Worst-Case: Es wird nach der maximalen Laufzeit des Algorithmus gesucht. Dies geschieht indem eine obere Schranke aufgestellt wird, über die die Laufzeit nicht steigt. Dieser Fall ist Sortieralgorithmus abhängig und kann nicht immer eine in umgekehrter Reihenfolge sortierte Liste sein.
Best-Case: Stellt die minimale Laufzeit da und wird durch eine untere Schranke realisiert. Auch hier fällt die Laufzeit nicht unter die Schranke. Der Best-Case ist nicht immer eine bereits aufsteigend sortierte Liste sein.
Average-Case: Es wird eine durchschnittliche Laufzeit aufgestellt.

Ein Vorteil der Landau-Notation ist, dass sie komplett unabhängig von Hardware und Betriebssystem ist. Mit ihr kann man die Laufzeiten von Algorithmen miteinander vergleichen. Ein großer Nachteil ist das die Funktionen nur angenähert sind und so nur eine grobe Einschätzung liefern. Weiterhin existiert keine Berücksichtigung auf Speicher Allokationen oder Zeitdauer von rekursive aufrufe.

2.1.2 In-Place

Diese Eigenschaft beschreibt ob der Algorithmus neben der zu sortierenden Liste noch weiteren Speicherplatz benötigt oder nicht. Eine Ausnahme ist der Tausch-Speicher der beim tausch zweier Werte benötigt wird.

2.1.3 Stabilität

Ein stabiler Algorithmus behält die Reihenfolge von äquivalenten Werten bei.

2.1.4 Heap/Stack Größe

2.1.5 Testumgebung

SSD vs HDD , CPU, Compiler, Sprache ...

2.2 Sortieralgorithmen

Vorstellungen von unterschiedlichen SA. min 5 bis (Textlimit erreicht ;D)

2.2.1 Vorstellung

Im folgenden Abschnitt werden wir uns hauptsächlich die Sortieralgorithmen im Detail anschauen. Für unsere Recherche haben wir uns auf drei grundlegende Verfahren eingeschränkt: Bubblesort, Quicksort und Mergesort. Der wohl am häufigsten verwendete Sortieralgorithmus ist der Quicksort. Seine Entstehung blickt weit zurück bis in die 60er Jahre und seither wird der Quicksort von vielen Forschern untersucht. Einer seiner haupt Eigenschaften ist es, dass er in-place abläuft, also keinen weiteren Speicher benötigt, um die Sortierung durchzuführen. Der Grunde, wieso Quicksort so schnell ist, ist, dass seine innere Schleife sehr kurz ist und somit einfach optimiert werden kann. Für die Sortierung von n Elementen wird im Durchschnitt n log (n) Operationen erfordet. Der Nachteil des Algorithmus ist, dass er rekursiv ist, also im worst-case braucht er n^2 Operationen. Ausgehend von diesen Behauptungen aus dem Netz, werden wir Tests durchführen, um zu schauen, in welchen Gebieten oder unter welchen Umständen diese genannten Aussagen gelten.

2.2.2 Pseudo Code

2.2.3 Eigenschaften

Wie sind die O-Notation, In-Place, Stabilität zu diesem SA

2.2.4 Testfälle

Worst Case, Average Case, Best Case, nearly sorted, festplattenart, genug Speicher, zuwenig Speicher, unterschiedliche Datentypen - Integer versus Klassenobjekte

2.3 Evaluierung

2.3.1 Vergleich der Ergebnisse

2.3.2 Relevanz der O-Notation

2.3.3 Wie wichtig sind weitere Kriterien

In-Place, Stabilität, Speicherplatz ...

3 Schluss

3.1 Zusammenfassung und Ausblick

3.1.1 Fazit

3.1.2 Anwendungstipps

4 Literatur Alt

Sortieralgorithmen unter mathematischen Gesichtspunkten `http://www.christian-rehn.de/wp-content/uploads/2009/08/Sortieralgorithmen.pdf`

Analysis and Testing of Sorting Algorithms on a Standard Dataset `http://ieeexplore.ieee.org/document/7280062/`

Rheinwerk OpenBook C von A bis Z `http://openbook.rheinwerk-verlag.de/c_von_a_bis_z`

Rheinwerk OpenBook Java ist auch eine Insel `http://openbook.rheinwerk-verlag.de/javainsel`

läuft das jetzt [1] wirklich?

5 Literaturverzeichnis

Literatur

[1] C. Rehn, "Sortieralgorithmen unter mathematischen gesichtspunkten."