

Aufwandsbeurteilung: Wie "schnell" ist ein Algorithmus?

Wenn man ein Element in einer sortierten Liste sucht und den Aufwand vergleicht, je nachdem ob man die einfache Suche (der Reihe nach jedes Element ansehen) oder die **binäre Suche** verwendet, kann man zu den folgenden Erkenntnissen gelangen.

Angenommen, es dauert 1ms, um ein Element zu überprüfen - wie lange dauert es jeweils, in einer Liste mit 100 Elementen das gesuchte zu finden? Naja, das kommt drauf an - es könnte ja bei beiden Methoden sein, das bereits die erste Betrachtung das gesuchte Element findet, dann dauert es 1ms¹⁾, es zu finden. Bei der einfachen Suche kann es aber durchaus auch 100ms dauern, wenn das gesuchte Element als letztes überprüft wird.



Darum einigt man sich, dass man bei der Beurteilung von Algorithmen bevorzugt den schlechtesten Fall („worst case“) betrachtet, um einen Eindruck zu bekommen, wie effektiv ein Algorithmus arbeitet

Der schlechteste Fall bei der binären Suche dauert 7ms, also 15 mal so lang. Die binäre Suche ist also 15 mal schneller als die einfache Suche?! Stimmt das? Dazu betrachten wir die folgende Tabelle, in der die „worst case“-Zeiten dargestellt sind.

Zahl der Elemente	Einfache Suche	Binäre Suche
100	100ms	7ms
10.000	10 Sekunden	14ms
1.000.000.000	11 Tage	32ms

Man sieht, dass die Laufzeiten **mit der Zunahme der Zahl der Elemente sehr unterschiedlich zunehmen**.



Um diesen Umstand darzustellen, verwendet man die **Landau-Notation** (oder auch „Big-O-Notation“). Die Landau Notation ermöglicht es, die Anzahl der Operationen die Algorithmen zur Lösung eines Problems benötigen zu vergleichen - sie teilt dir mit wie schnell die Zahl der Operationen des Algorithmus **anwächst**.

Für unser Beispiel oben:

Einfache Suche	Binäre Suche
$O(n)$	$O(\log n)$

Übung

Du sollst auf einem Blatt Papier 16 rechteckige Kästchen erzeugen. Wie effizient sind die folgenden beiden Algorithmen?

1. Du zeichnest mit einem Stift nacheinander Kästchen für Kästchen auf das Blatt.
 2. Du faltest das Blatt „über Kreuz“ jeweils auf seine Hälfte und verwendest die Faltmarken als Ränder der Quadrate.
- Überlege zunächst, wieviele Operationen du für 16 Kästchen jeweils benötigst.
 - Wieviele Operationen benötigt man, um 32 oder 64 Kästchen zu erzeugen?
 - Wie effizient sind die beiden Vorgehensweisen in der Landau-Notation?
 - Josephins sagt: „Stopp! - wir haben doch gar nicht festgelegt, was beim Zeichnen einer Operation entspricht. Ist das das Zeichnen eines Rechtecks oder das Ziehen einer Linie? Im zweiten Fall benötigt das Zeichnen eines Rechtecks ja 4 Operationen!“ Was sagst du dazu?

"Typische" Laufzeiten von Algorithmen

- $O(\log n)$ - „*logarithmische Laufzeit*“, z.B. die binäre Suche
- $O(n)$ - „*lineare Laufzeit*“, z.B. die einfache Suche
- $O(n \cdot \log n)$, ein schneller Sortieralgorithmus
- $O(n^2)$, ein langsamer Sortieralgorithmus
- $O(n!)$, ein sehr langsamer Algorithmus (Problem des Handlungsreisenden, [12](#))

Zusammenfassung



- Die Geschwindigkeit eines Algorithmus wird **nicht** als Zeit (z.B. in Sekunden) angegeben, sondern durch die Betrachtung der **Zunahme der Anzahl der Operationen**.
- Wir schauen, wie schnell die Laufzeit zunimmt, wenn die Größe der Eingabe zunimmt.
- Die Laufzeit wird in der Landau Notation aufgeschrieben

¹⁾
wenn man annimmt, dass die anderen Operationen im Vergleich fast keine Zeit beanspruchen

From:
<https://wiki.qg-moessingen.de/> - QG Wiki

Permanent link:
https://wiki.qg-moessingen.de/faecher:informatik:oberstufe:algorithmen:big_o:start?rev=1595505058

Last update: 23.07.2020 13:50

