Laufzeitverhalten

a.k.a. Anzahl der Rechenschritte / Vergleichsoperation in Abhängigkeit der Menge der Eingaben (n)

Notation: „Big-O-Notation" ⇒ z.B. $O(n)$ → lineare Abhängigkeit

(auch Deutsch: Landau-Symbol)

Ein paar konkrete Beispiele:

- Datenstruktur „Liste" ↔ sortiert: $O(n)$ → Faktor $f \approx \frac{1}{2}$
  unsortiert: $O(n)$ → $\frac{1}{2} < f \leq 1$
- Datenstruktur „Baum" → sortiert: $O(\log n)$ → $f \approx \log_2 n$ (= Höhe des Baumes)
  unsortiert: $O(n)$ → $f \leq 1$
- Tiefensuche: $O(n!)$ → Jeder Knoten muss min. 1 mal besucht werden → tendiert zu fakultativ
- Breitensuche: $O(n^2)$ → Dijkstra: rekursiv von jedem Knoten aus denkbar → immer nur kürzesten Weg verfolgen

} Suche nach kürzestem Weg

Sortieralgorithmen:

	Best	Avg	Worst	
• binary-tree-sort	$n \log n$	$n \log n$	n^2	→ 1. Binärbaum erzeugen: $O(n)$ 2. Suchen: $O(n) \Leftrightarrow O(\log n)$
• Heap-sort	$n \log n$	$n \log n$	$n \log n$	→ 1. Sort Heap erzeugen: $O(n)$ 2. Suchen: $O(\log n)$
• Insertion-sort	n	n^2	n^2	→ Random Elemente der Eingabe entnehmen und einsortieren
• Selection-sort	n^2	n^2	n^2	→ z.B. kleinstes Element wird einer nach vorherigen kleinsten gestellt
• Gnome-sort	n	n^2	n^2	→ „Gnome" läuft Liste ab und nimmt größere Elemente mit
• Merge-sort	$n \log n$	$n \log n$	$n \log n$	→ braucht viel Speicher → Teilen und intelligentes mergen
• natural Merge-sort	n	$n \log n$	$n \log n$	→ viel Speicher → ebenfalls Sortierung vor dem Teilen
• Bubble-sort	n	n^2	n^2	→ größere Elemente „schwimmen" nach „oben"
• Shaker-sort	n	n^2	n^2	→ Bubble-Sort von beiden Seiten

• Quicksort	$n \log n$	$n \log n$	n^2	→ meist genutzt → „intelligentes Teilen"
• Bucket-sort	n	n	$n \log n$	→ logische, grobe Sortierung in „Buckets" → Sortierung der Buckets

⇒ Worst-Case sind meist bereits sortierte Listen die nicht als solche erkannt werden

Passwörter → Arten der Verschlüsselung

Symmetrische Verschlüsselung

„One password to rule them all"

Ver- und Entschlüsselung mit dem selben Schlüssel

Bsps: Caesar-Verschlüsselung,
Blowfish-Verschlüsselung,
AES

Problem: „Wie übergebe ich das Passwort?"

Vorteil

Asymmetrische Verschlüsselung

„Frodo, we still need the private-Ring"

Ver- und Entschlüsselung mit unterschiedlichen Keys!

- public-Key → öffentlich → geht aus privat Key hervor!
- private Key → geheim

Bsps: RSA („Länge des Keys")

Angriffsmöglichkeiten:

Brute-Force
„planloses" Ausprobieren aller möglichen Zeichenkombinationen

Dictionary-Attack
geplantes Ausprobieren aller sinnvollen Zeichenkombinationen

Max. Angriffsdauer berechnen:

Zeit pro Zeichen: 1ns ⇒ für 4-Dez.-Zahlen: $10^4 \cdot 1ns$

→ evtl. Umrechnung in Tage und Stunden