2.3 Sortieren

Sortierproblem

- Bringe eine gegebene Folge von Datenobjekten in eine wohldefinierte Reihenfolge.
- Folge ≠ Menge !! (doppelte Objekte)
- Reihenfolge (Ordnungsrelation)

Ordnungsrelationen

- Menge M, Relation $\leqslant$
- Partielle Ordnung
  - Reflexivität $x \leqslant x$
  - Transitivität $x \leqslant y \land y \leqslant z \Rightarrow x \leqslant z$
  - Antisymmetrie $x \leqslant y \land y \leqslant x \Rightarrow x = y$
- Strikter Anteil
  - $x < y :\Leftrightarrow x \leqslant y \land x \neq y$

Ordnungsrelationen

- Menge M, Relation $\leqslant$
- Totale Ordnung
  - Trichotomie

    $\forall x,y: x < y \vee x = y \vee x > y$
- Sortierschlüssel
  - key[objekt]
  - z.B. Adresse nach Nachnamen sortiert

Ordnungsrelationen

- Skalare Typen besitzen in der Regel eine totale Ordnung
- Mengen von Integer $2^{\mathbb{N}}$, Mengeninklusion $\subseteq$
  - Partielle Ordnung … {1,2}, {2,3}
- Punkte in der Ebene $\mathbf{R}\times\mathbf{R}$
  - Keine Ordnung

    $(x,y) \leqslant (u,v) \Leftrightarrow x^2 + y^2 \leqslant u^2 + v^2$
  - Totale Ordnung (Lexikographisch)

    $(x,y) \leqslant (u,v) \Leftrightarrow x < u \vee (x=u \wedge y \leqslant v)$

Sortierproblem

- Gegeben eine Folge von Objekten

  $R_1,\ldots,R_n$

- Schreibweise

  $R_i \leqslant R_j :\Leftrightarrow key[R_i] \leqslant key[R_j]$

- Finde eine Permutation π, so dass

  $\forall\, i=1,\ldots,n-1: R_{\pi(i)} \leqslant R_{\pi(i+1)}$

- Insbesondere

  $\forall\, i,j: i < j \Rightarrow R_{\pi(i)} \leqslant R_{\pi(j)}$

Warum sortieren?

- Bringe Objekte mit gleichen oder ähnlichen Schlüsseln zusammen
- Detektiere doppelte Objekte
- Vorbereitung zum Suchen
  - $O(n) \rightarrow O(\log n)$
  - Lohnt sich nur, wenn

    $m \times n \times C_1 > C_{sort}(n) + m \times \log(n) \times C_2$

Begriffe

- Direktes / Indirektes Sortieren
  - Direkt: Umkopieren der Objekte R_i
  - Indirekt: Erzeuge Permutationstabelle π
    - Initialisierung $\pi(i) := i$
    - Zugriff per $R_{\pi(i)}$
    - Sortieren bzgl. mehrerer Schlüssel / Relationen
- Stabilität (Reihenfolge von Objekten mit gleichem Schlüssel bleibt erhalten)

Kriterien

- Berechnungsaufwand
  - $O(n^2)$, $O(n \times \log n)$, $O(n)$
- Worst vs. Average Case
  - Ausnutzen von Vorsortierungen
- Speicherbedarf
  - Kopieren
  - In-place