Informatik II
Sortieralgorithmen

G. Zachmann
Clausthal University, Germany
zach@tu-clausthal.de

Motivation

- Preprocessing fürs Suchen
- Sind für kommerzielle Anwendungen häufig die Programmteile, die die meiste Rechenzeit verbrauchen
- Viele raffinierte Methoden wurden im Laufe der Zeit entwickelt, von denen wir ein paar kennenlernen wollen

Die Sortieraufgabe

- Eingabe: Datensätze (records) aus einem File, der Form
  - Key und *satellite/payload* data

- Sortierschlüssel kann aus einem oder mehreren Feldern des Datensatzes bestehen (z.B. Nachname + Vorname)
- Bedingung: auf den Keys muß eine totale Ordnungsrelation $\preceq$ definiert sein, d.h., es gilt
  - Trichotomie: für alle Keys a,b gilt genau eine Relation

$$a \prec b, \quad a = b, \quad a \succ b$$

  - Transitivität:

$$\forall a, b : a \prec b \wedge b \prec c \Rightarrow a \prec c$$

- Aufgabe: bestimme eine Permutation $\Pi = (p_1, \ldots, p_n)$ für die Records, so daß die Keys in nicht-fallender Ordnung sind:

$$K_{p_1} \preceq \ldots \preceq K_{p_n}$$

- Implementierung üblicherweise als Klasse mit eingebautem Vergleichsoperator:

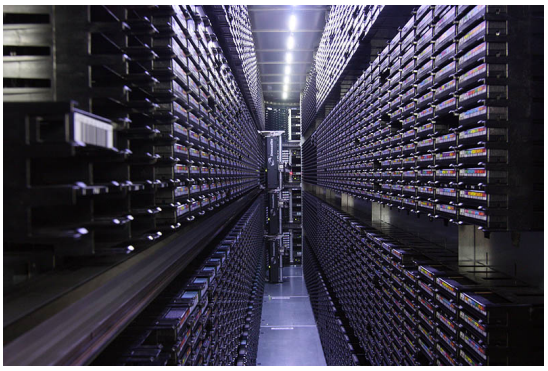
```
class MyData:
    def __init__( self, key, value ):
        self.key = key
        self.value = value
    def __cmp__( self, other ):
        if self.key < other.key:
            return -1
        elif self.key > other.key:
            return 1
        else:
            return 0
a = MyData(…)
b = MyData(…)
if a < b:
   ...
```

Klassifikation / Kriterien von Sortierverfahren

- Interne Sortierverfahren:
  - Alle Datensätze befinden sich im Hauptspeicher
  - Es besteht random access auf den gesamten Datenbestand
  - Bekannte Verfahren: Bubblesort, Insertionsort, Selectionsort, Quicksort, Heapsort
- Externe Sortierverfahren:
  - Die Datensätze befinden sich in einem Hintergrundspeicher (Festplatte, Magnetband, etc.) und können nur sequentiell verarbeitet werden
  - Bekanntes Verfahren: Mergesort

Exkurs: Tape Libraries

- Wird auch heute noch für Datenarchive gerne verwendet
- Beispiel (Deutsches Klimarechenzentrum, Stand 2010):
  - 8 robots per library
  - 500 TeraByte disk cache
  - Total capacity: 60 PetaByte
  - Projected fill rate: 10 PetaByte/year



Fortsetzung Klassifikation

- Vergleichsbasiert (comparison sort): zulässige Operationen auf den Daten sind nur Vergleich und Umkopieren
- Zahlenbasiert: man darf/kann auf den Keys auch rechnen
  - Diese Unterscheidung ist analog zu der bei den Suchalgorithmen
- Stabil (stable): Gleiche Keys haben nach dem Sortieren die selbe relative Lage zueinander wie vorher
- Array-basiert vs. Listen-basiert: Können Datensätze beliebig im Speicher angeordnet sein (Liste), oder müssen sie hintereinander im Speicher liegen (Array)
- In-Place (in situ): Algorithmus braucht nur konstanten zusätzlichen Speicher (z.B. Zähler; aber keine Hilfsarrays o.ä.)
  - Konsequenz: Ausgabe-Array = Eingabe-Array