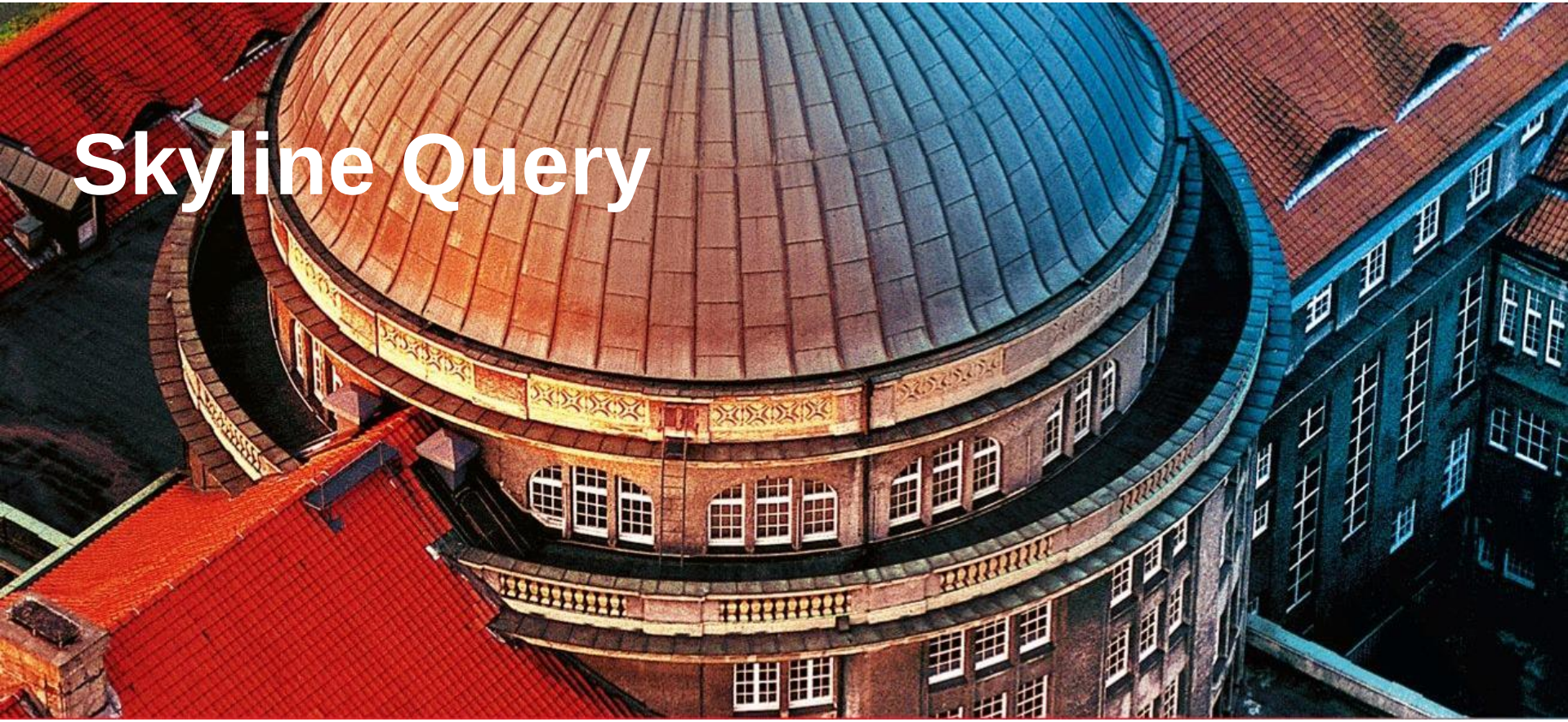
Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

Skyline Query



Agenda

1. Anwendung
2. Skyline Eigenschaften
3. Skyline Beispiel
4. Algorithmen
    - 4.1. Block-Nested-Loop Algorithmus
    - 4.2. Divide & Conquer Algorithmus

1. Anwendung

- Umgang mit großen Datenmengen
- Unscharfe Selektionskriterien
- Nutzerpräferenzen in relationale Abfragen einbauen
- Interest Points aus einem Datenset herausfiltern
- Versucht die optimale Lösung auf Basis von multiplen und konkurrierenden Zielen zu finden
- Oft keine einzige optimale Antwort, die genau die Präferenzen des Anwenders widerspiegelt, sondern mehrere Möglichkeiten, die in Betracht gezogen werden können → Pareto optimal Set
- Entscheidungsunterstützung aus der der Nutzer je nach Prioritäten seine Lösung aussuchen kann

1. Anwendung

- Customer Information Services
- Entscheidungsunterstützung
- Decision Making Systems
- Kann bei einer großen Varietät an Datentypen verwendet werden
  - ungeordnet
  - Unvollständige und unsichere Daten

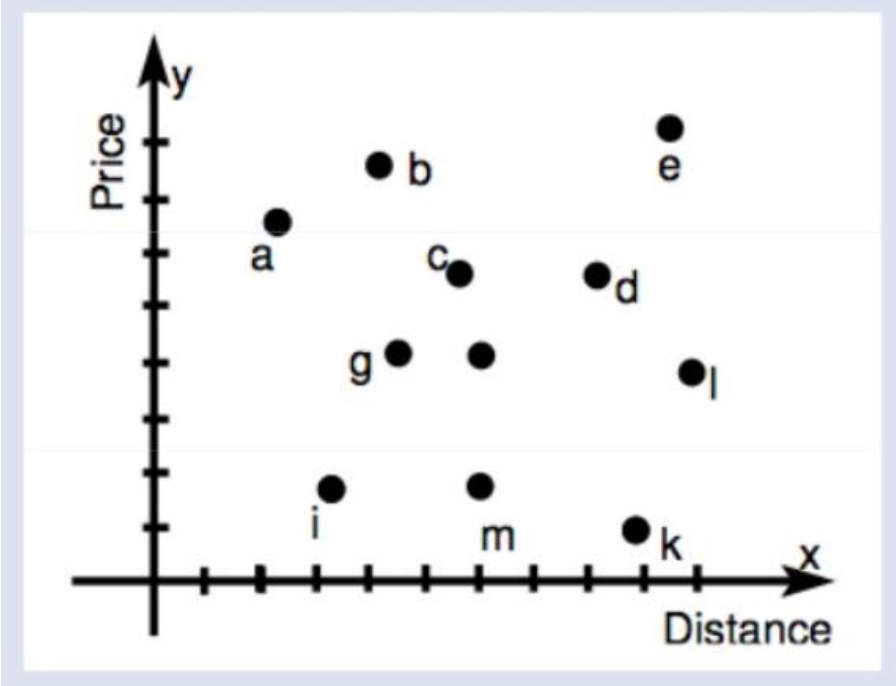
2. Skyline

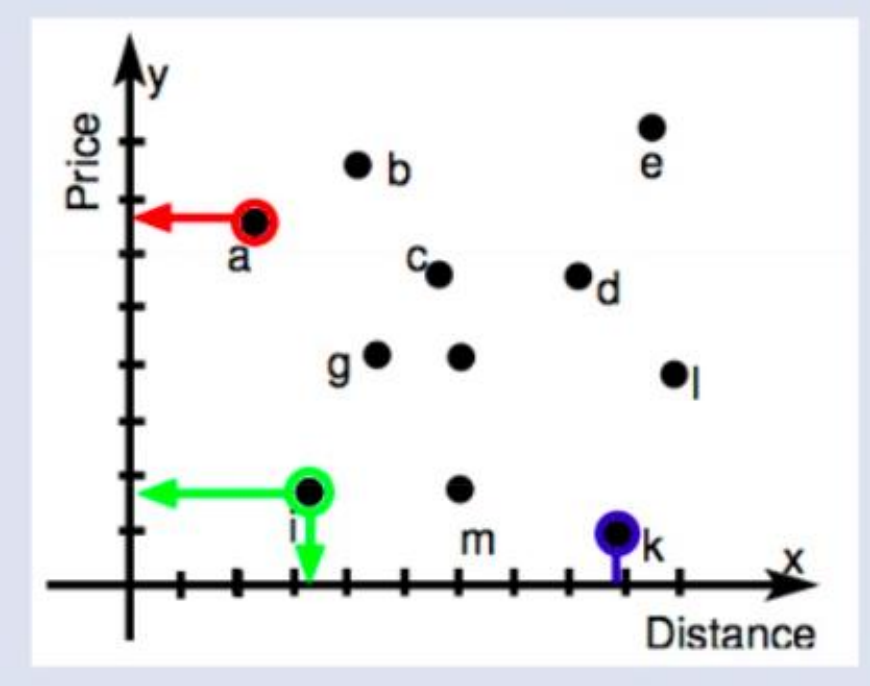
- Alle Punkte, die nicht von einem anderen dominiert werden
- Ein Punkt dominiert einen anderen wenn er
  - In mind. Einer Dimension besser und
  - In allen anderen Dimensionen mind. Genauso gut ist
- Ein Punkt in der Skyline muss unvergleichbar zu allen anderen Punkten der Skyline sein

3. Skyline Beispiel

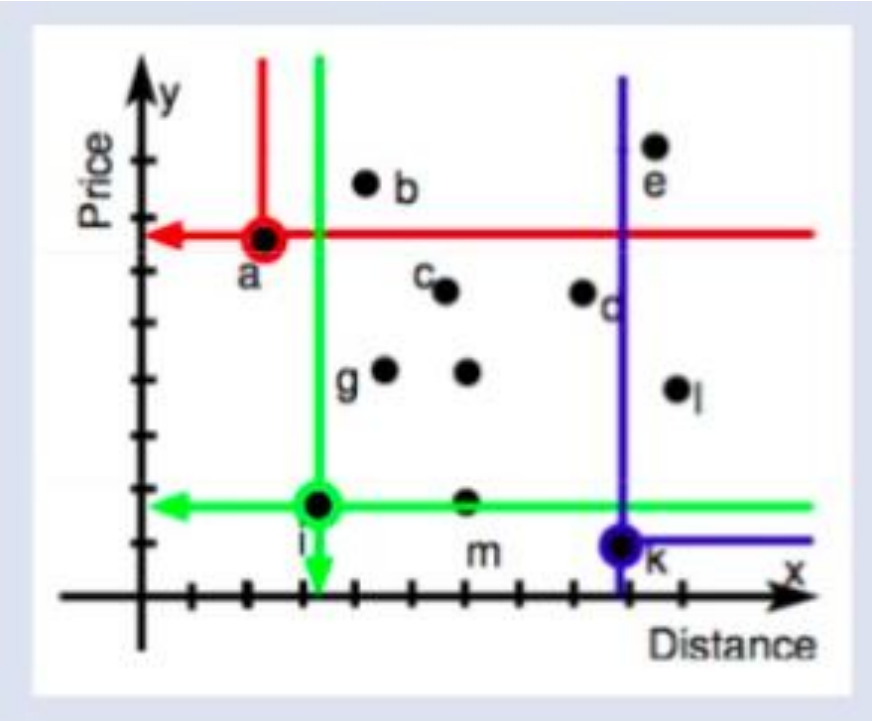
- Szenario: Hotelsuche
- 2 Kriterien
  - Preis
  - Nähe zum Strand

3. Skyline Beispiel





3. Skyline Beispiel



- a hat die kleinste Distanz zum Strand
- k hat den günstigsten Preis
- i hat weder die kleinste Distanz noch den günstigsten Preis
  - Günstiger als a
  - Dichter am Strand als k
  - Von keinem der beiden Punkte dominiert
- Alle anderen Punkte werden von a, i oder k dominiert

4. Algorithmen

- Block- Nested-Loop Algorithmus
- Divide & Conquer Algorithmus

4.1 Block-Nested-Loop Algorithmus

- Naiver Nested-Loop Algorithmus ineffizient
- BNL baut einen Block aus mehreren Skyline Tupeln in jeder Iteration
- Legt ein Fenster an, in das die nichtvergleichbaren (Skyline) Tupel gespeichert werden
- Jedes neue Tupel (p) wird mit allen Tupeln des Fensters verglichen
- 3 Möglichkeiten
  - P wird von einem Tupel des Fensters dominiert
  - P dominiert ein oder mehrere Tupel des Fensters
  - P ist nicht vergleichbar mit den Tupeln des Fensters

    → p wird im Fenster gespeichert wenn dort noch Platz ist ansonsten in temp Ordner und bei der nächsten Iteration ins Fenster

4.1 Block-Nested-Loop Algorithmus

- Am Ende jeder Iteration werden die Tupel des Fensters ausgelesen, die gespeichert wurden als der temp Ordner noch leer war
  - Teil der Skyline
- Alle Tupel im Fenster und im temp Ordner bekommen einen Zeitstempel
  - Wenn Tupel mit Zeitstempel t ausgelesen wird, können alle Tupel mit einem Zeitstempel < t ausgelesen werden
  - Stellt sicher, dass Algorithmus irgendwann aufhört
- Algorithmus funktioniert am besten bei einem kleinen Datenset

4.2 Divide & Conquer Algorithmus

- Gesamtproblem rekursiv in Teilprobleme zerlegen um aus Teillösungen eine Lösung für das Gesamtproblem zu konstruieren
- 1. Median m einer Dimension berechnen
- 2. Datenset in 2 Partitionen unterteilen
  - P1: alle Tupel mit einem Wert besser als m
  - P2: alle Tupel mit einem schlechteren Wert als m
- Für beide Partitionen wird die Skyline berechnet
- Es wird so lange partitioniert bis nur noch ein oder wenige Tupel in einer Partition enthalten sind

4.2 Divide & Conquer Algorithmus

- Skylines S1 und S2 werden verglichen und miteinander gemerged um die gesamte Skyline zu erhalten
- Median einer 2. Dimension zur Partitionierung von S1 und S2
  - S1.1, S1.2, S2.1 und S2.2
  - Mergen der Skyline Partitionen
    - S1.2 mit S2.2
    - S1.1 mit S2.2
    - S1.1 mit S2.1
  - So lange bis alle Dimensionen berücksichtigt oder eine der Partitionen leer ist