

# Effiziente und effektive Ähnlichkeitssuche auf komplexen Objekten

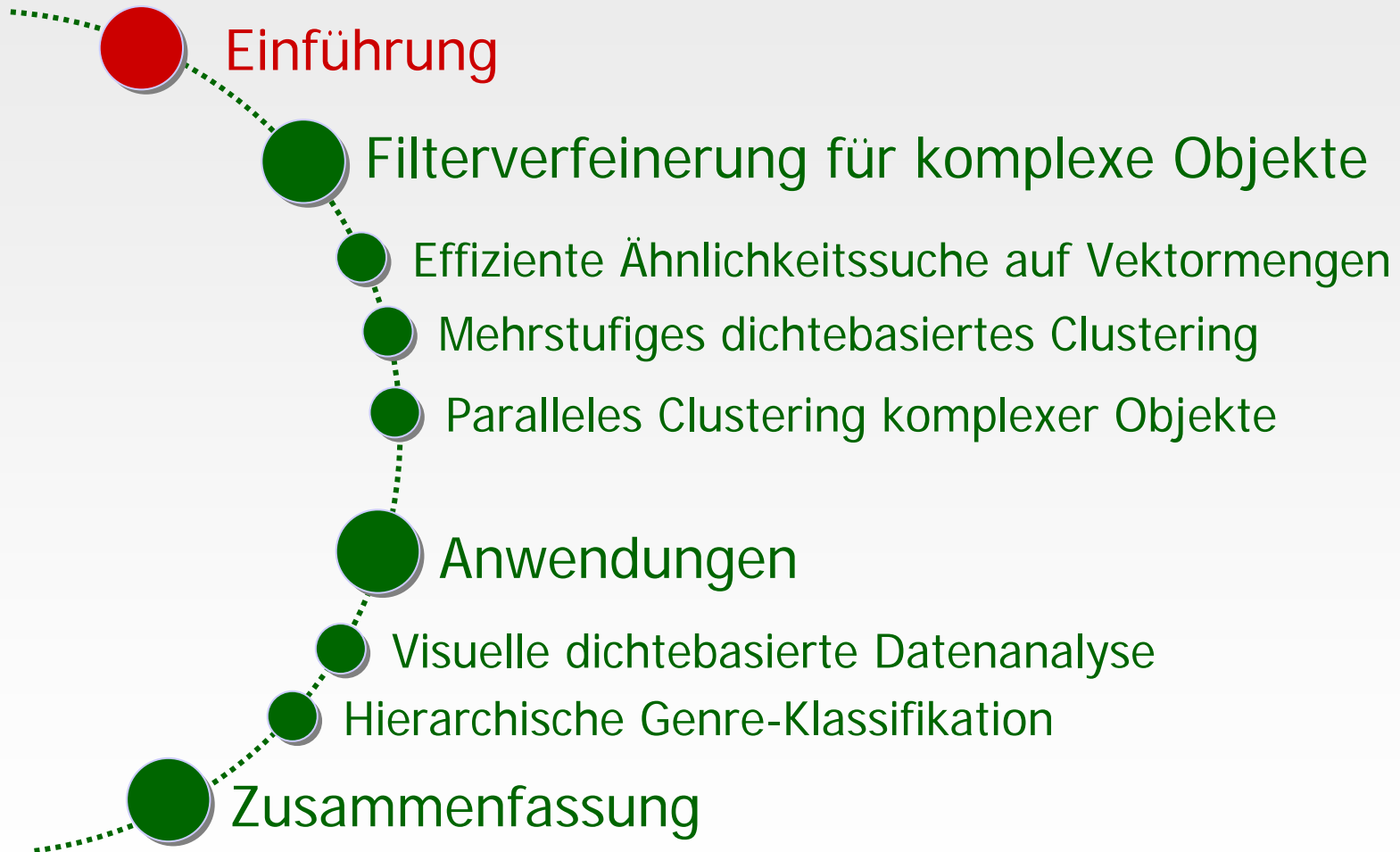
Stefan Brecheisen

Ludwig-Maximilians-Universität München  
Department „Institut für Informatik“  
Lehr- und Forschungseinheit für Datenbanksysteme

22. Februar 2007

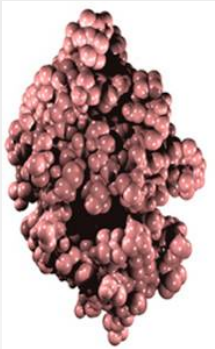
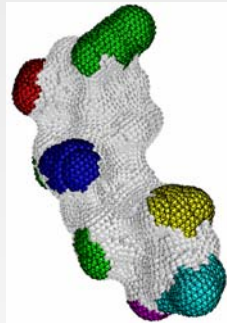
# Überblick

---

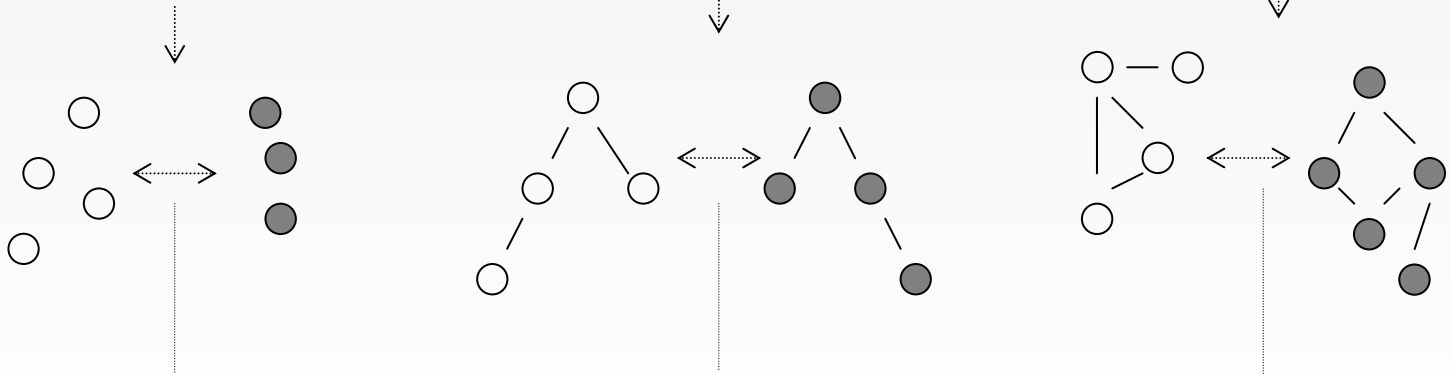


# Komplexe Objekte

komplexe Objekte



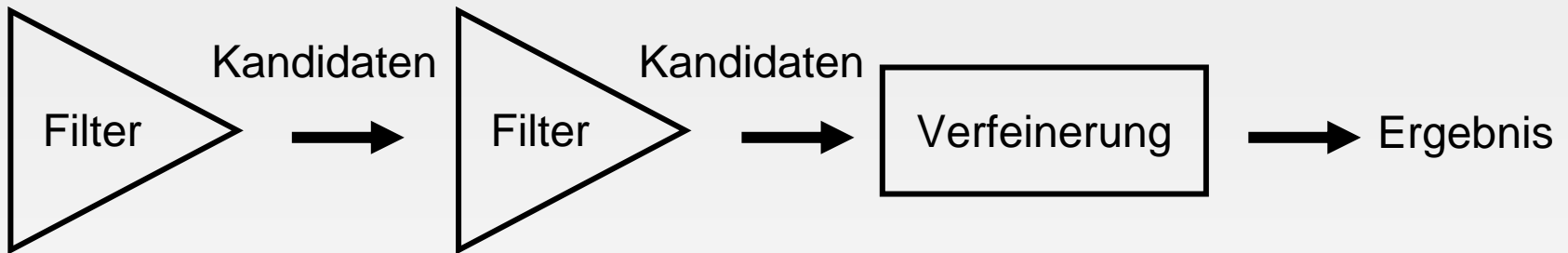
komplexe Modelle



komplexe Distanzmaße

# Mehrstufige Anfragebearbeitung

Effizienzsteigernde Maßnahmen:



- Mehrstufige Anfragebearbeitung  
→ Untere-Schranke-Eigenschaft

$$\forall o_1, o_2 \in O : d_f(o_1, o_2) \leq d_o(o_1, o_2)$$

- Indexstrukturen  
→ Metrische Indexstrukturen, z.B. M-tree

# Überblick

---



# Mengen von Feature-Vektoren

---

Featurebasierte Ähnlichkeit:

- Extraktion von  $d$ -dimensionalen Feature-Vektoren
- Ähnlichkeitsmaß ist die Distanz im Feature-Raum

Distanzbasierte Ähnlichkeit:

- Ähnlichkeitsmaß ist direkt auf den Objekten definiert, z.B. Bäume, Graphen

Mittelweg:

- Fasse pro Objekt  $k$  Feature-Vektoren zu einer Menge zusammen
- Ähnlichkeitsmaß ist die Distanz zwischen den Vektormengen

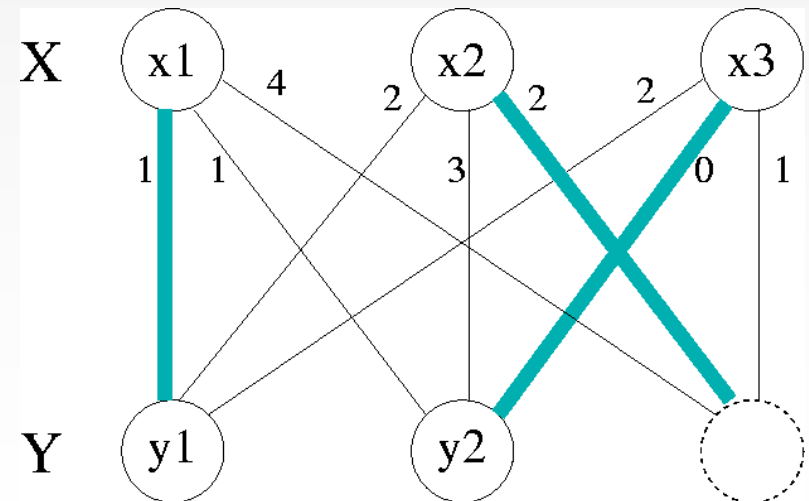
# Distanzmaße auf Vektormengen

Anforderungen an Distanzmaß:

- Eignung für Ähnlichkeitssuche
- Metrik

→ Bestimme maximales Matching mit minimalem Gewicht  
(Minimal-Matching-Distanz)

- Gegeben Vektormengen  $X$  und  $Y$ ,  
 $|Y| \leq |X| \leq k$
- Konstruiere vollständigen bipartiten  
Graph  $G = (X \cup Y, X \times Y)$
- Das Gewicht jeder Kante  
 $(x, y) \in X \times Y$  ist  $dist(x, y)$
- Gewichtsfunktion  $w$  für nicht  
zugeordnete Knoten, falls  $|X| \neq |Y|$



# Distanzmaße auf Vektormengen

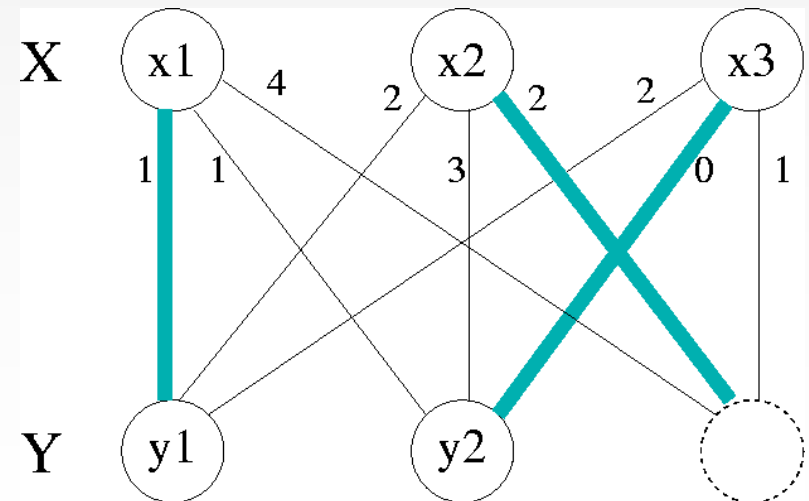
Partielle Minimal-Matching-Distanz:

- Bestimme Zuordnung zwischen  $s \leq |X|$  Vektoren

Zeitaufwand:

- für vollständiges Matching  $O(k^3 + k^2d)$
- für partielles Matching  $O(\binom{k}{s}sk^2 + k^2d)$

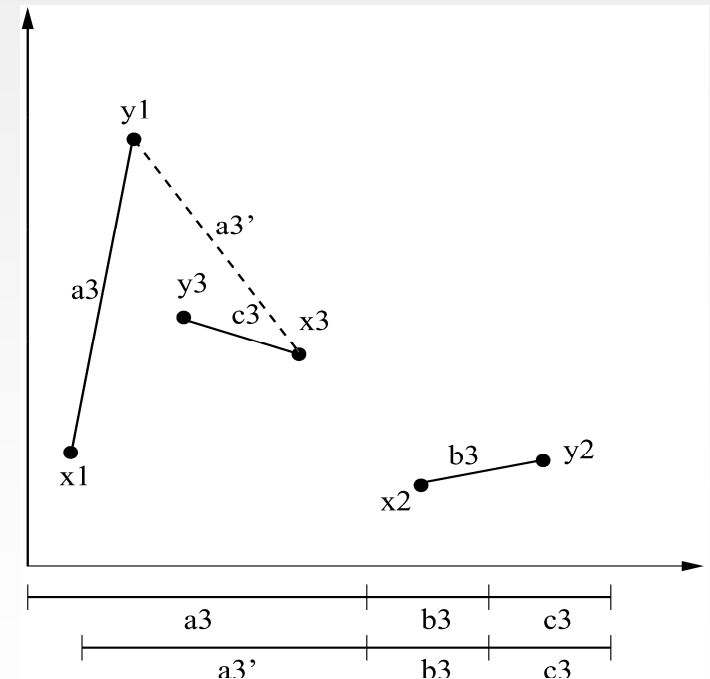
- Gegeben Vektormengen  $X$  und  $Y$ ,  
 $|Y| \leq |X| \leq k$
- Konstruiere vollständigen bipartiten  
Graph  $G = (X \cup Y, X \times Y)$
- Das Gewicht jeder Kante  
 $(x,y) \in X \times Y$  ist  $dist(x,y)$
- Gewichtsfunktion  $w$  für nicht  
zugeordnete Knoten, falls  $|X| \neq |Y|$



# Filter auf Vektormengen

## Closest-Pair-Ansatz:

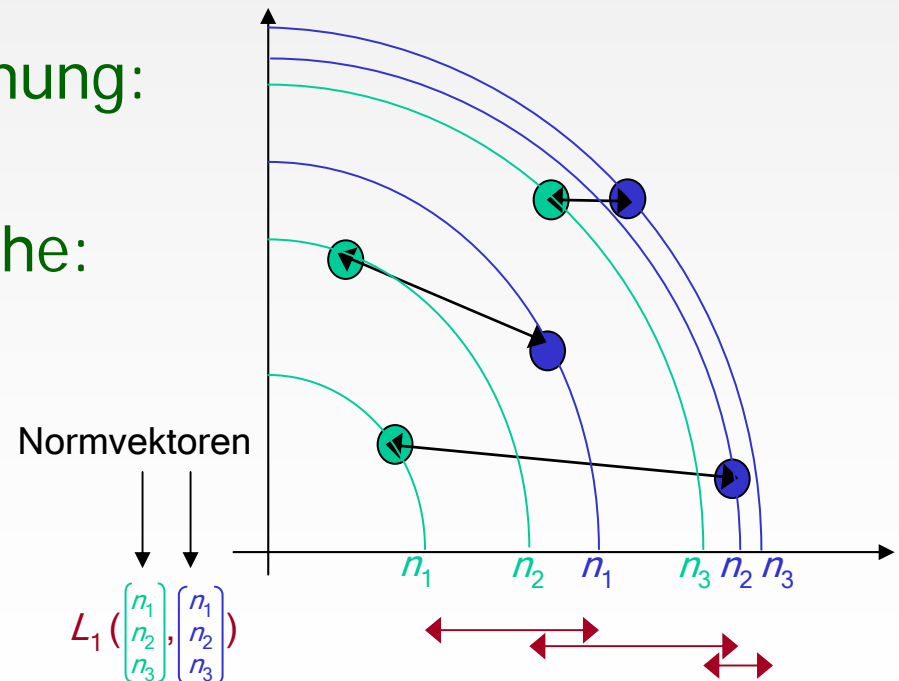
- Summieren der Abstände zu den nächsten Nachbarn in der anderen Menge
- 1:n-Zuordnungen sind zugelassen
- Kosten einer Distanzberechnung:  $O(k^2d)$
- Für partielle Ähnlichkeitssuche: Summieren der  $s$  kürzesten Abstände zwischen den nächsten Nachbarn



# Filter auf Vektormengen

## Norm-Vektor-Ansatz:

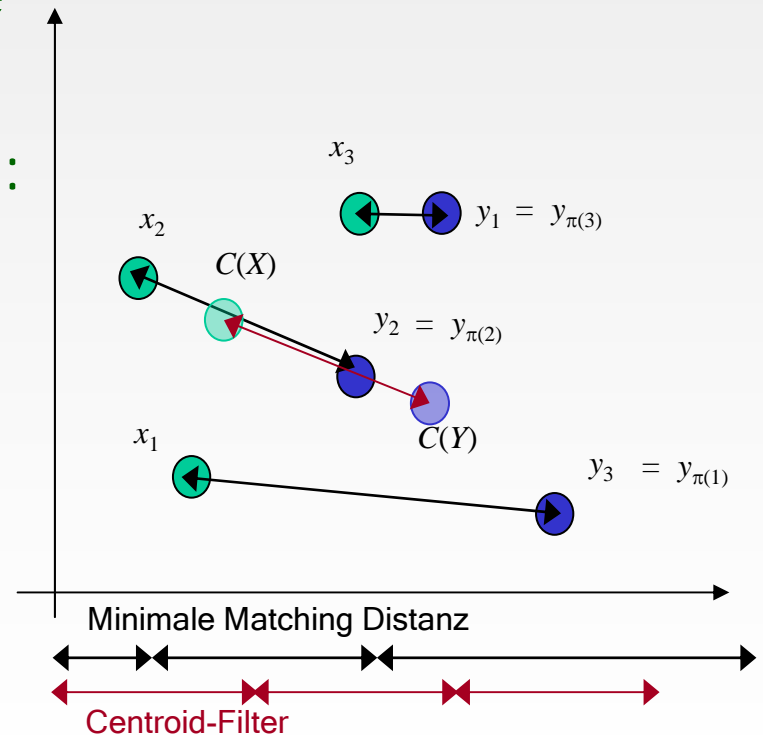
- Aggregation über die Dimensionen
- Zusammenfassen der sortierten Normwerte im Normvektor
- Kosten einer Distanzberechnung:  $O(k)$
- Für partielle Ähnlichkeitssuche: Closest-Pair-Ansatz auf den Normwerten



# Filter auf Vektormengen

## Centroid-Ansatz:

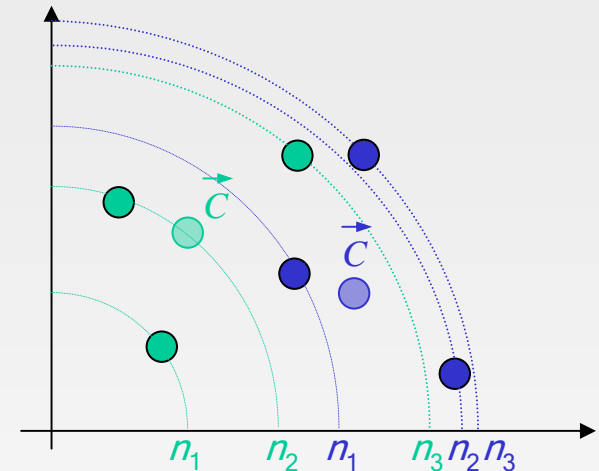
- Aggregation über die Vektoren
- Approximation der Vektormenge durch ihren Centroid
- Kosten einer Distanzberechnung:  $O(d)$
- Für partielle Ähnlichkeitssuche nicht anwendbar



# Filter auf Vektormengen

## Kombinierter Ansatz:

- Aggregation über Punkte und Dimensionen
- Kosten einer Distanzberechnung:  $O(d+k)$

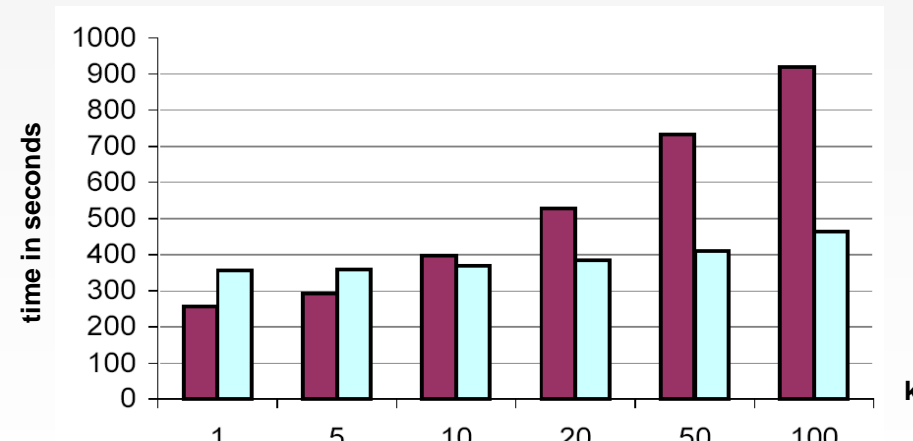
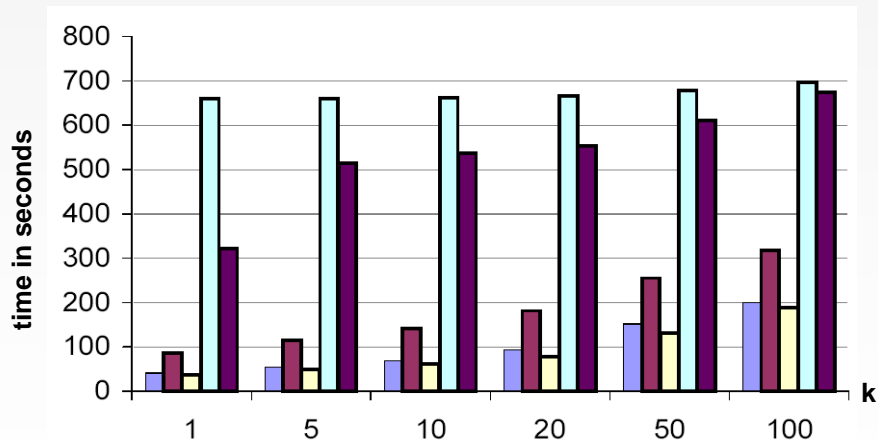
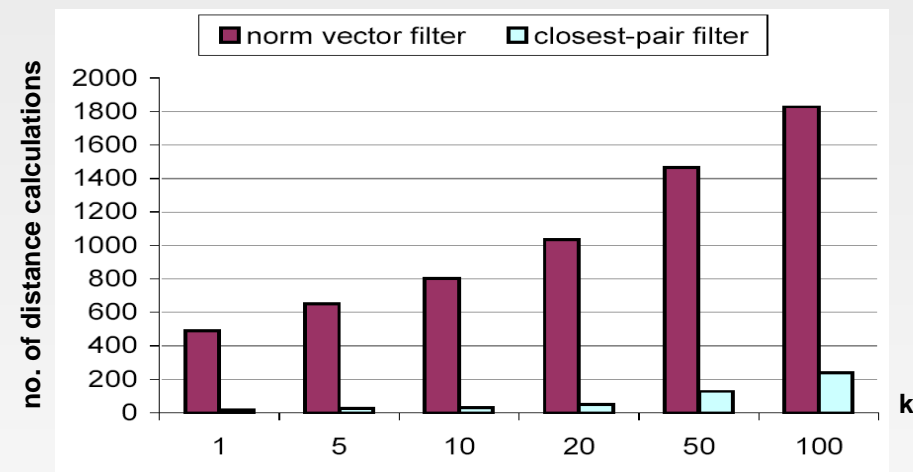
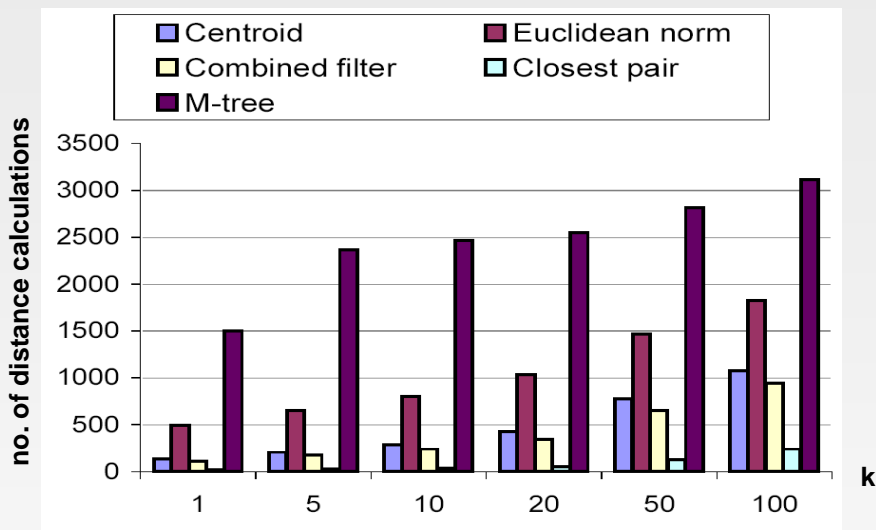


$$\max \left\{ L_1 \left( \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix}, \begin{bmatrix} n_1 \\ n_2 \\ n_3 \end{bmatrix} \right), L_p \left( \vec{C}, \vec{C} \right) \right\}$$

## Zusammenfassung:

	exact distance	closest pair	centroid	norm vector
complete similarity	$O(k^3 + k^2 d)$	$O(k^2 d)$	$O(d)$	$O(k)$
partial similarity	$O(\binom{k}{s} s k^2 + k^2 d)$	$O(k^2 d \log s)$	n/a	$O(k \log s)$

# Experimentelle Evaluation

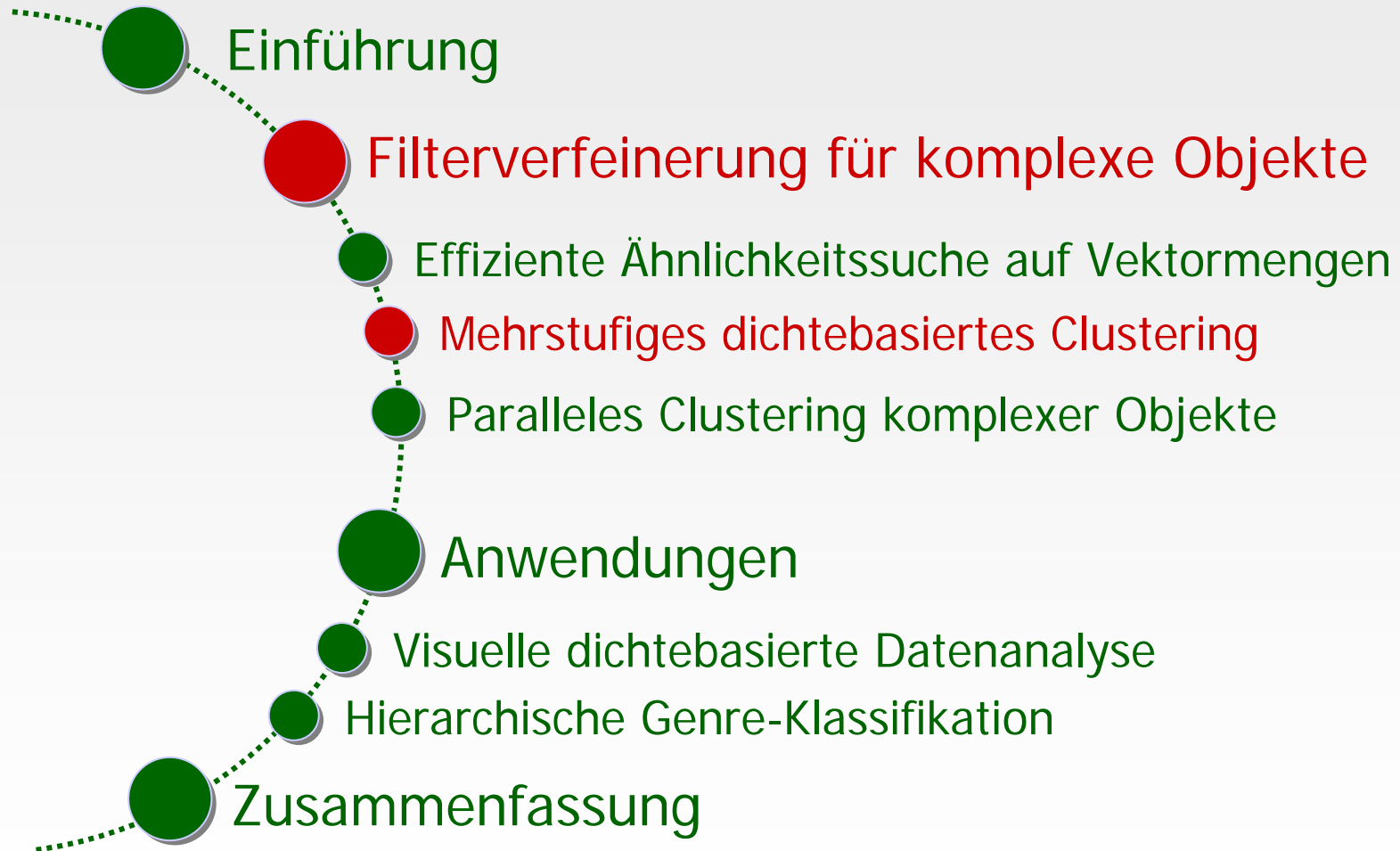


vollständige knn-Anfragen

partielle knn-Anfragen

# Überblick

---



# Clustering

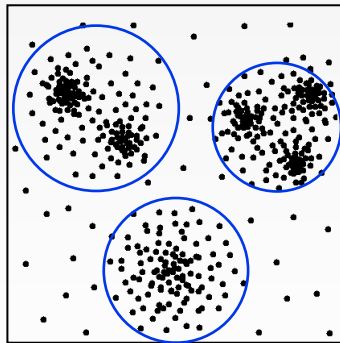
---

## Clustering

Gruppierung der Datenbank in Cluster, so dass

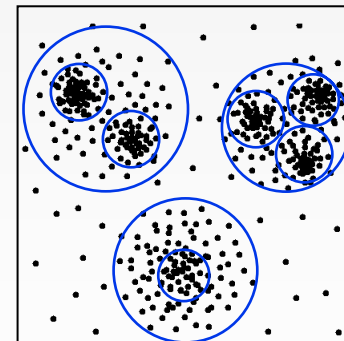
- ähnliche Objekte zusammengefasst werden
- unähnliche Objekte getrennt werden

### Flaches Clustering



z.B. dichtebasierter Algorithmus  
DBSCAN [KDD 96]

### Hierarchisches Clustering

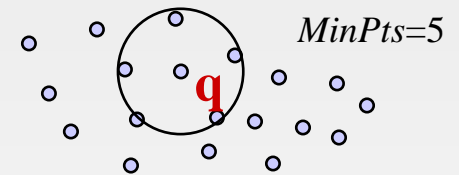


z.B. dichtebasierter Algorithmus  
OPTICS [SIGMOD 99]

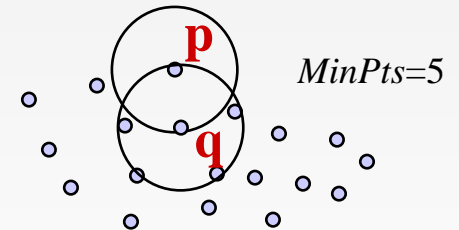
# Dichtebasiertes Clustering

- Parameter
  - Bereich  $\varepsilon$  and minimale Objektanzahl  $MinPts$

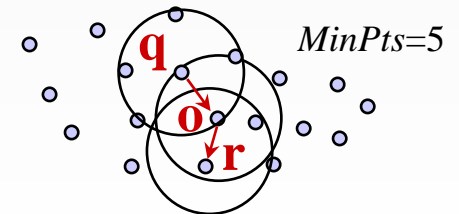
- Definition: Kernobjekt
  - $q$  ist Kernobjekt, falls  $| rangeQuery(q, \varepsilon) | \geq MinPts$



- Definition: direkt dichte-erreichbar
  - $p$  ist direkt dichte-erreichbar von  $q$ , falls  $q$  Kernobjekt und  $p \in rangeQuery(q, \varepsilon)$



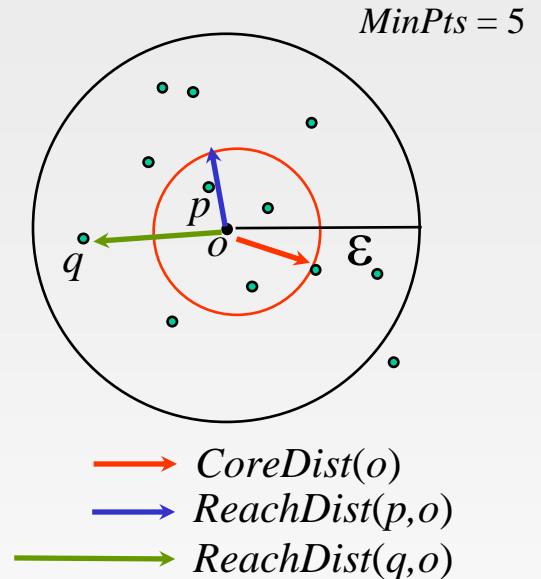
- Definition: dichte-erreichbar
  - transitive Hülle von "direkt dichte-erreichbar"



# Dichtebasiertes Clustering

- Grundidee von OPTICS:

Lineare Ordnung der Objekte,  
so dass Objekte eines Clusters  
in der Ordnung benachbart sind.



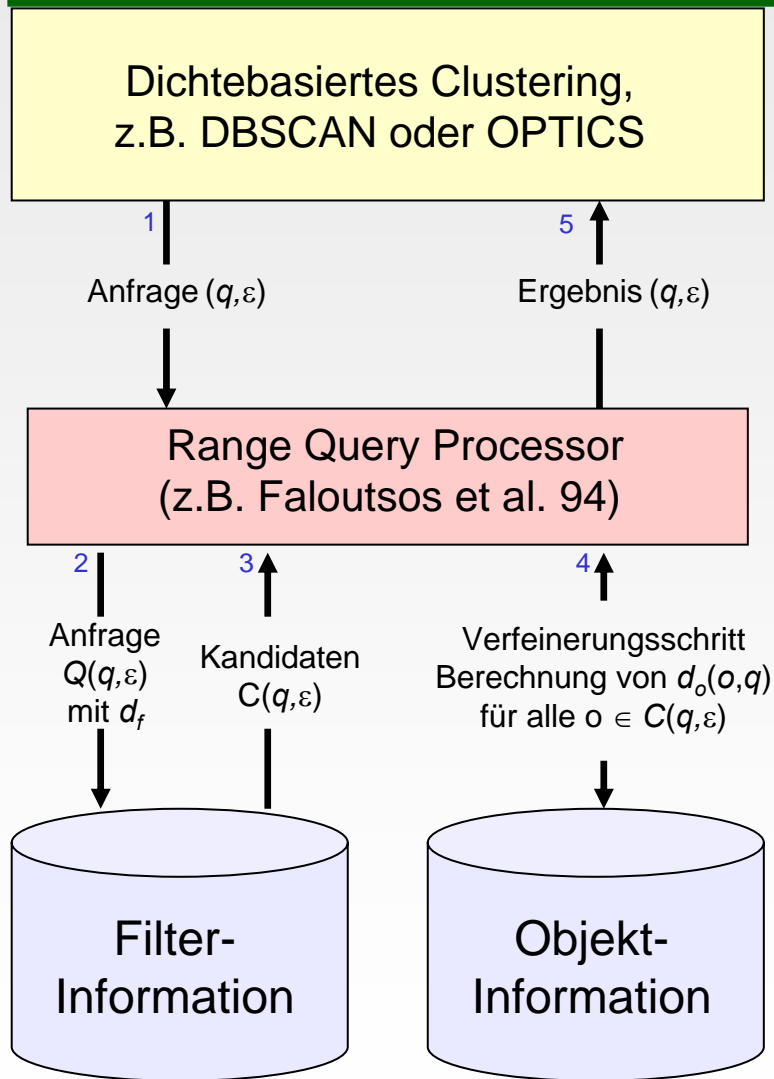
- Definition: Kerndistanz

$$CoreDist_{\varepsilon, MinPts}(o) = \begin{cases} \infty & \text{falls } |rangeQuery(o, \varepsilon)| < MinPts \\ MinPtsDist(o) & \text{sonst} \end{cases}$$

- Definition: Erreichbarkeitsdistanz

$$ReachDist_{\varepsilon, MinPts}(p, o) = \max(CoreDist_{\varepsilon, MinPts}(o), dist(p, o))$$

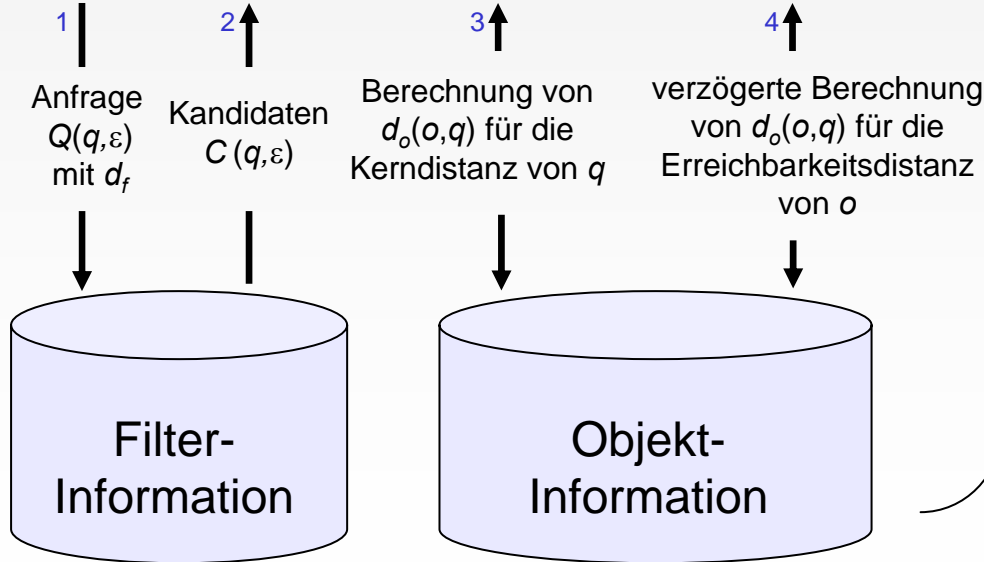
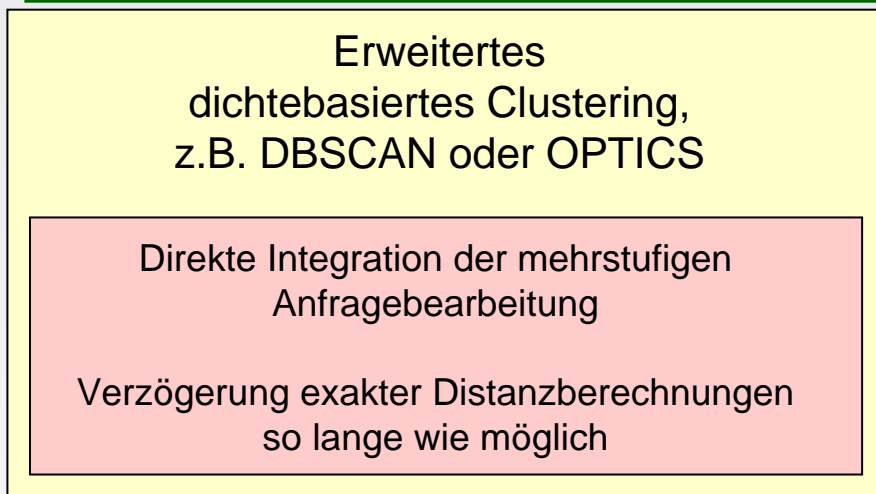
# Herkömmliches mehrstufiges Clustering



## Performance-Probleme

- Bereichsanfrage für jedes Datenbankobjekt  $q$  (1).
- Ausführung der Bereichsanfrage auf der Filter-Information (2,3).
- Eine teure exakte Distanzberechnung  $d_o(o, q)$  für jedes Objekt  $o$  aus der Kandidatenmenge  $C(q, \varepsilon)$  wird ausgeführt (4,5).
- Verfeinerungsschritt sehr teuer für wenig selektive Filter oder hohe  $\varepsilon$ -Werte.

# Integriertes mehrstufiges Clustering



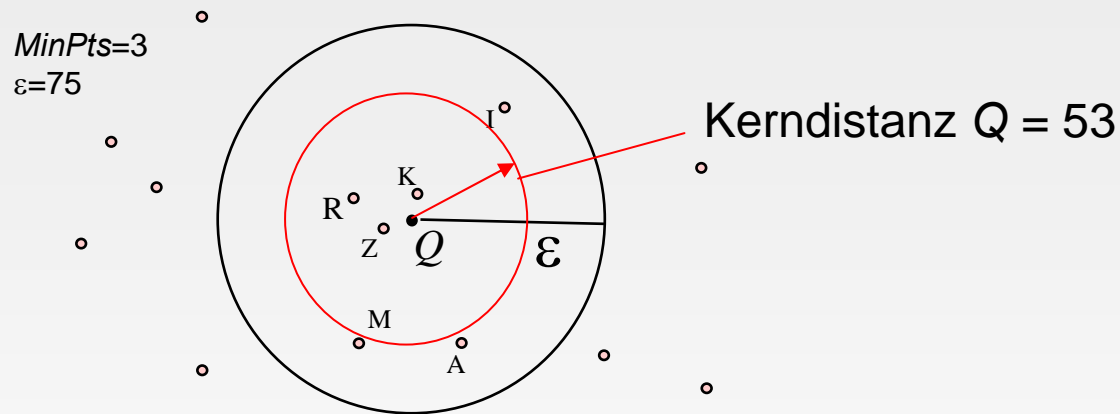
## Lösung

- Ausführung einer Bereichsanfrage auf der Filter-Information für jedes Datenbankobjekt  $q$  (1,2).
- Berechnung derjenigen exakten Distanzen  $d_o(o, q)$ , die zur Bestimmung der Kerndistanz von  $q$  nötig sind (3).
- Heuristik zur Bestimmung der Erreichbarkeitsdistanz, die exakte Distanzberechnungen einspart (4).

# Integriertes mehrstufiges Clustering

## Filter-Information

## Sortierte Distanzliste



$$d_o(K,Q)=43$$

$$d_o(Z,Q)=49$$

$$d_o(R,Q)=53$$

$$d_f(M,Q)=55$$

$$d_f(A,Q)=58$$

$$d_f(I,Q)=65$$

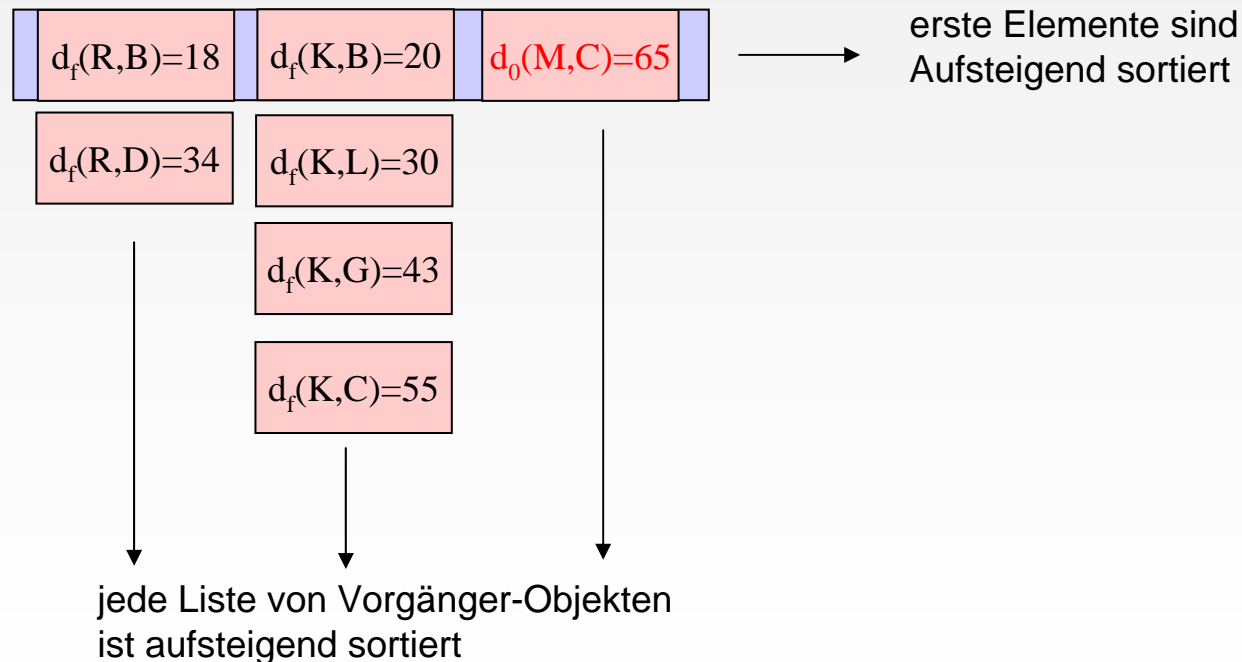
## Bestimmung der Kerndistanz

1. Filter-Bereichsanfrage für jedes Anfrageobjekt  $Q$
2. Sortierung der Kandidatenmenge aufsteigend nach der Filterdistanz
3. Exakte Distanzberechnungen bis  $MinPts$  nächste Nachbarn bestimmt wurden

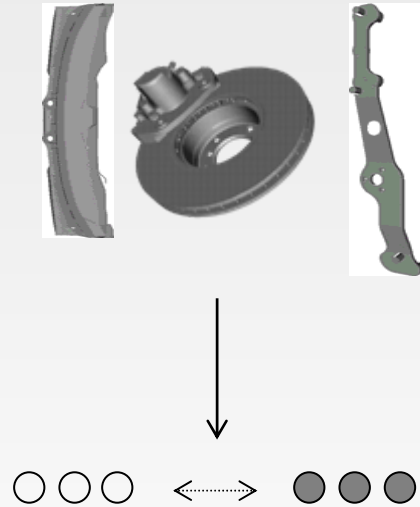
# Integriertes mehrstufiges Clustering

## Erweiterte Seedlist: Liste von Listen

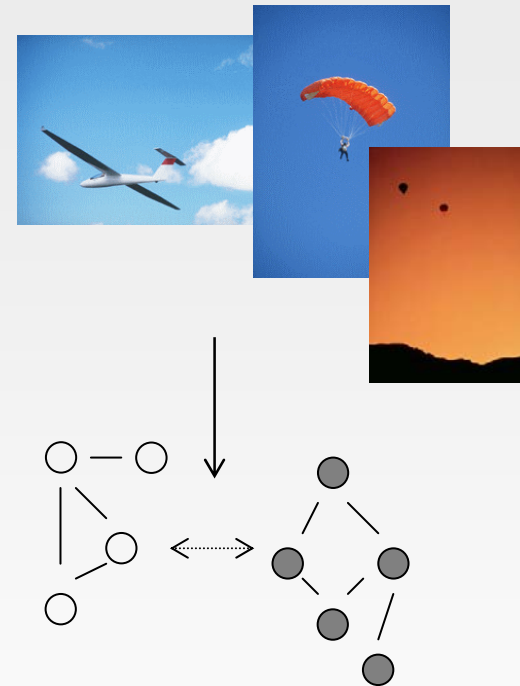
- Information über mögliche Vorgängerobjekte wird gespeichert
- Verzögerte Berechnung der exakten Distanzen



# Testdaten



- CAD-Objekte
- Hochdimensionale Feature-Vektoren, Mengen von Feature-Vektoren
- wenig selektiver Filter (Norm-Distanz, Zentroid-Distanz)

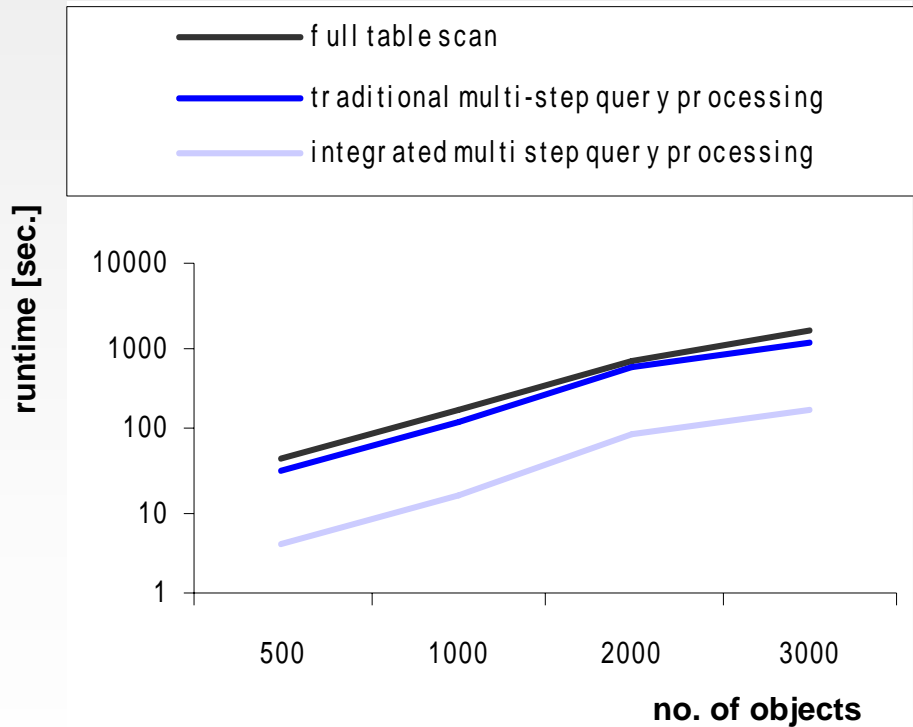


- Bilddaten
- Graphen
- teure Objekt-Distanz
- selektiver Filter

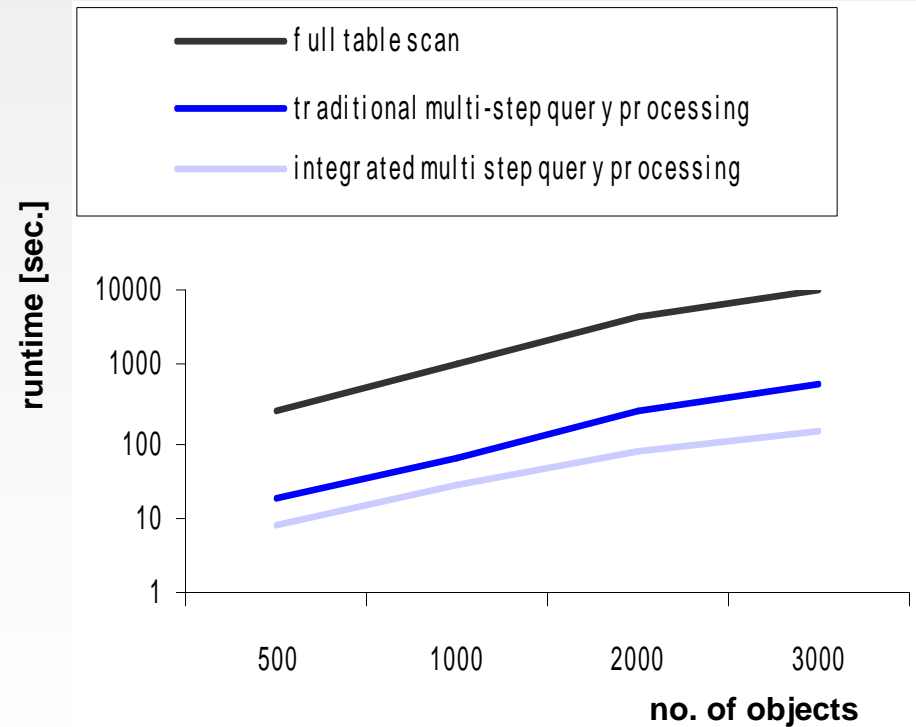
# Experimentelle Evaluation

## DBSCAN

### Feature-Vektoren



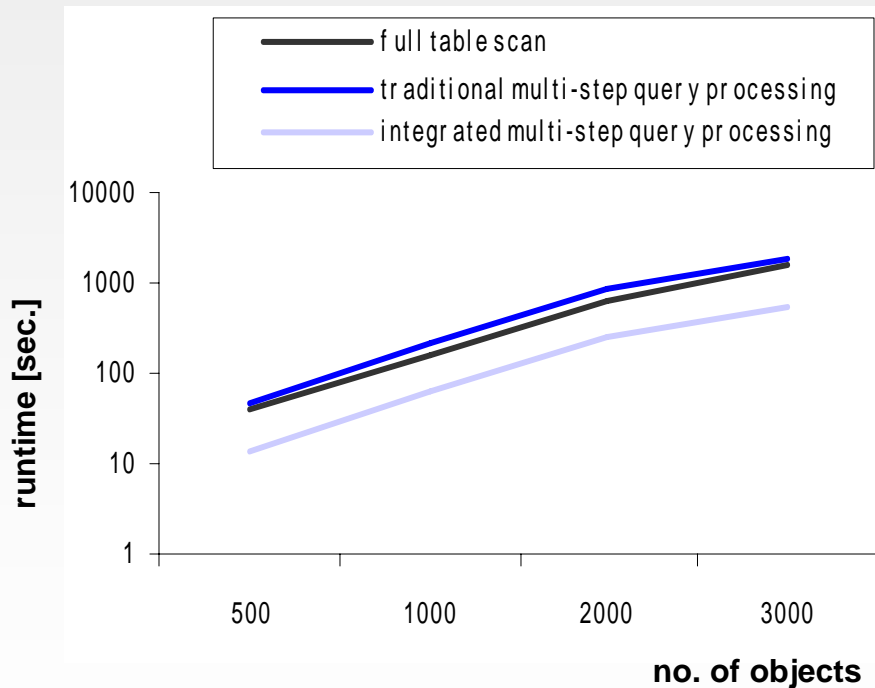
### Graphen



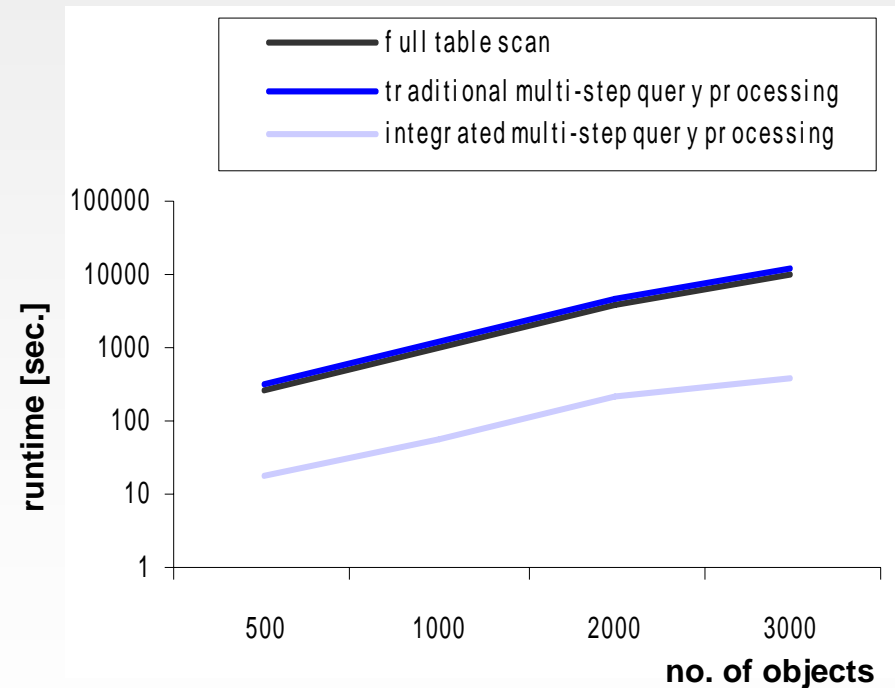
# Experimentelle Evaluation

## OPTICS

### Feature-Vektoren

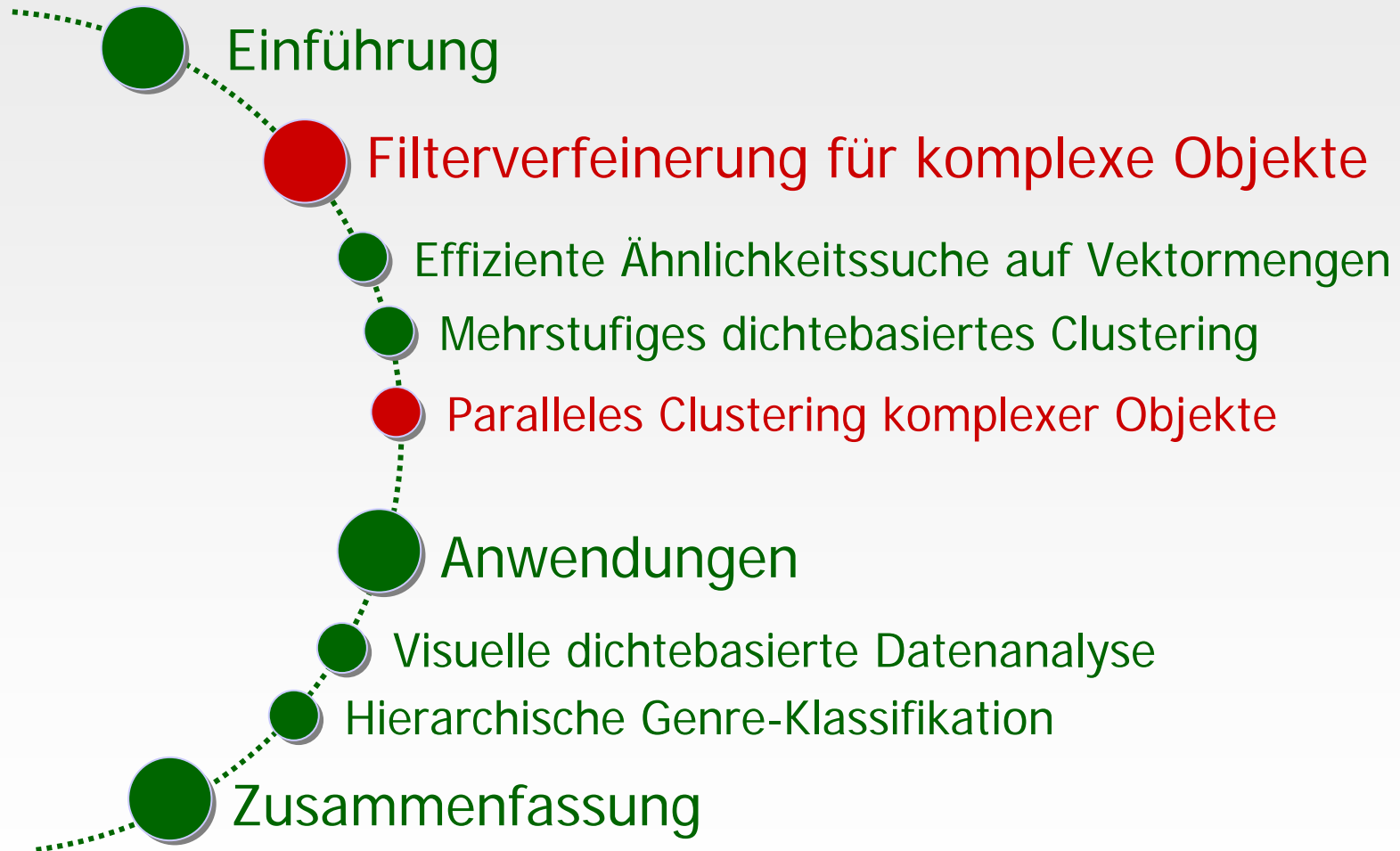


### Graphen



# Überblick

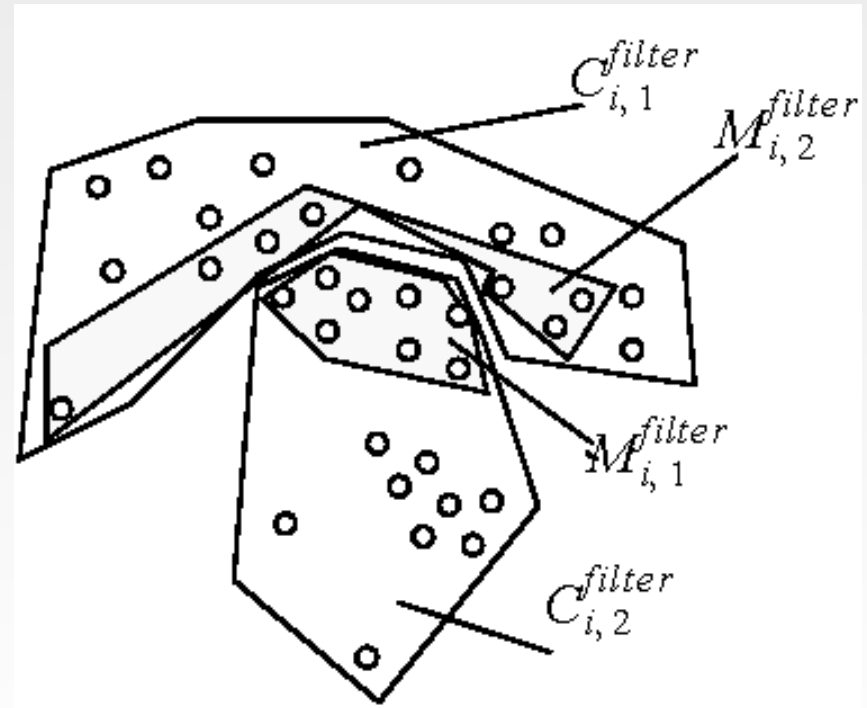
---



# Server-seitige Partitionierung

OPTICS-Lauf auf den Filterdistanzen ergibt:

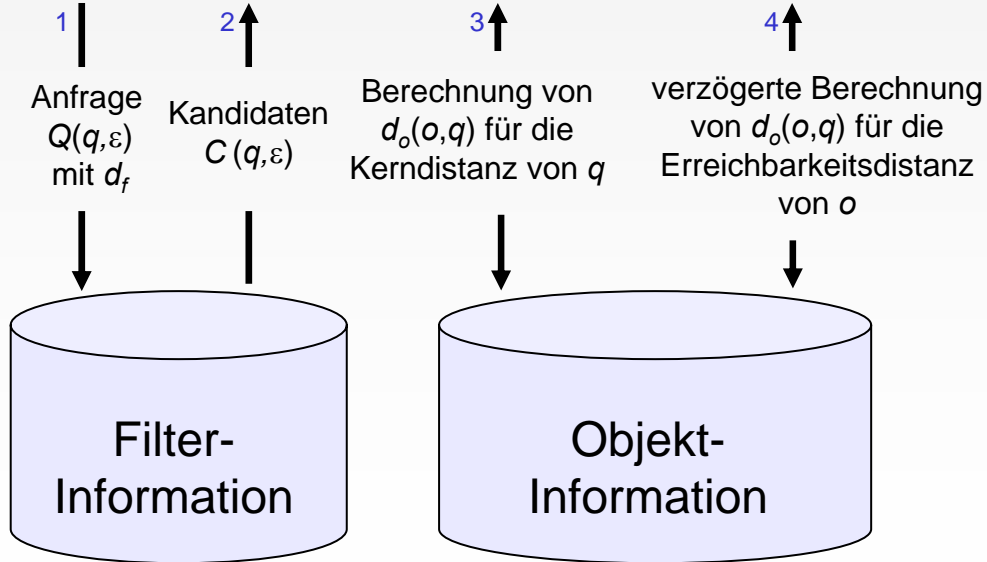
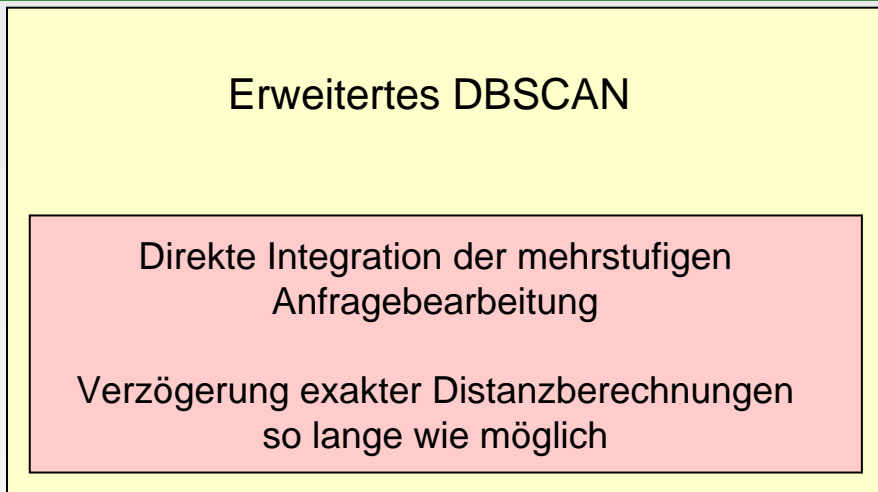
- Konservativ approximierte Cluster, d.h. Filter-Cluster sind Obermengen der exakten Cluster
- Progressiv approximierter Noise, d.h. Filter-Noise ist Teilmenge des exakten Noise



Filter-Merge-Punkte:

- Aufspaltung großer Filter-Cluster zur gleichmäßigen Lastverteilung
- Benötigt zur Bestimmung der Kerndistanz beim client-seitigen Clustering

# Client-seitiges Clustering



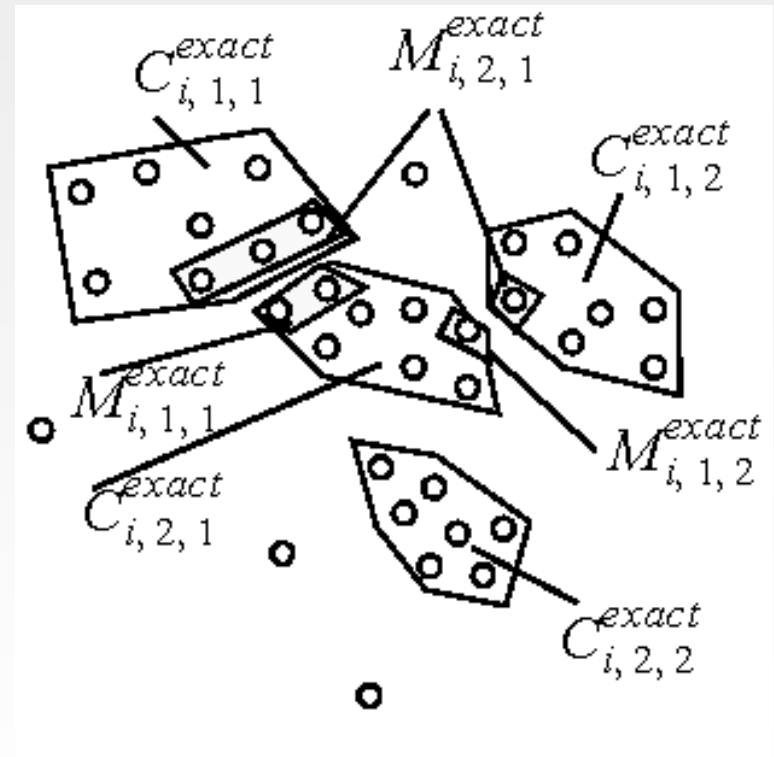
Erweitertes DBSCAN mit  
integrierter mehrstufiger  
Anfragebearbeitung

Zusätzlich Berücksichtigung  
der Filter-Merge-Punkte und  
Bestimmung exakter Merge-  
Punkte

# Server-seitige Verschmelzung

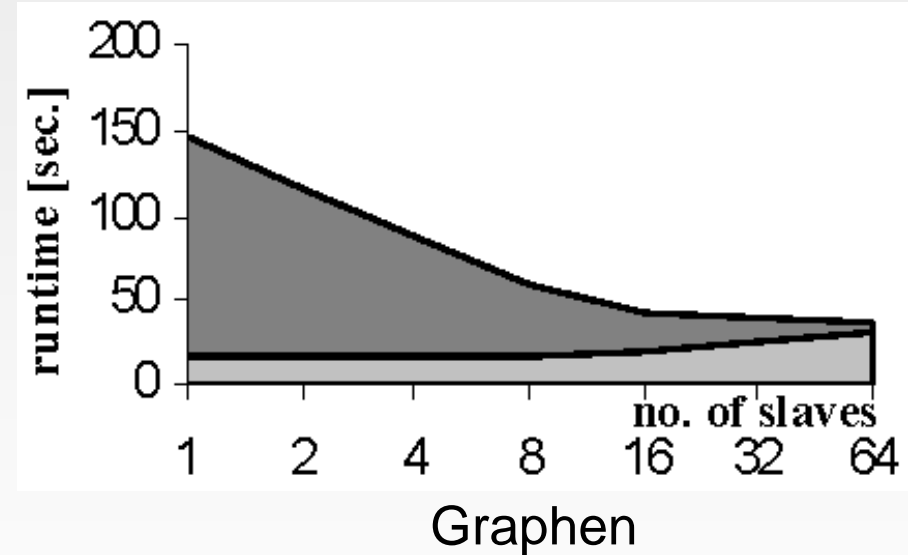
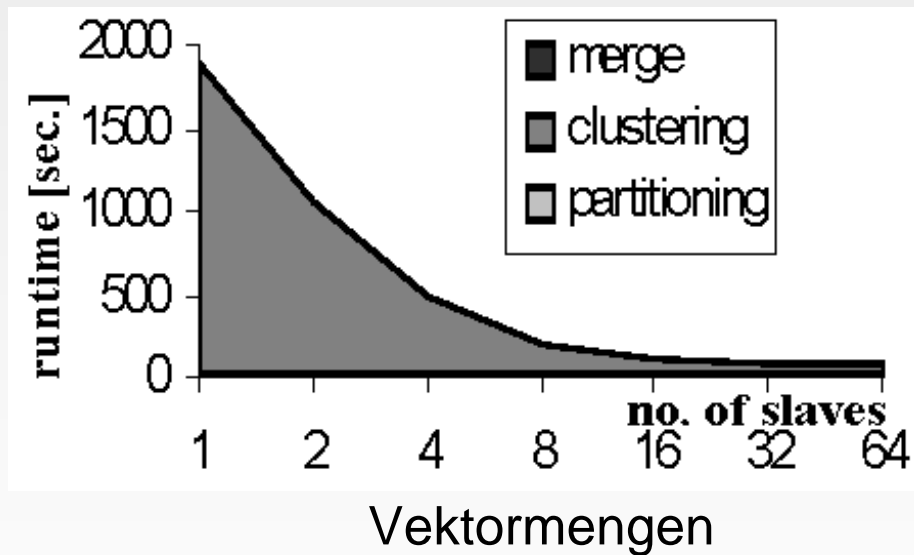
Verschmelzen der lokalen Clusterings:

- Exakte Merge-Punkte, d.h. die im exakten lokalen Cluster dichte-erreichbaren Filter-Merge-Punkte
- Cluster-Verbindungsgraph, d.h. Knoten für jeden exakten lokalen Cluster, Kante zwischen den Clustern, die einen exakten Merge-Punkt gemeinsam haben
- Datenbank-Verbindungsgraph, d.h. Vereinigung aller Cluster-Verbindungsgraphen
- Menge der Zusammenhangskomponenten entspricht einem globalen exakten DBSCAN-Clustering.



# Experimentelle Evaluation

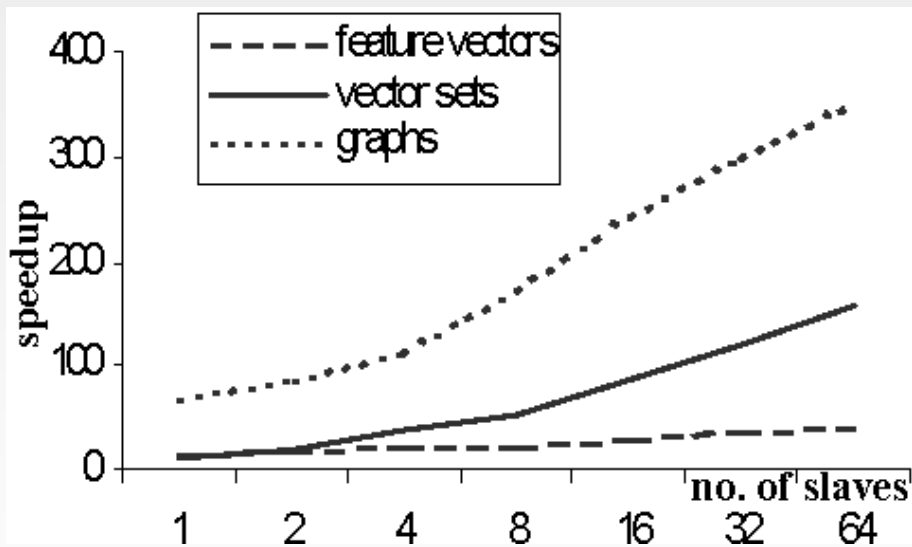
## Akkumulierte Laufzeiten bei unterschiedlicher Anzahl Slaves



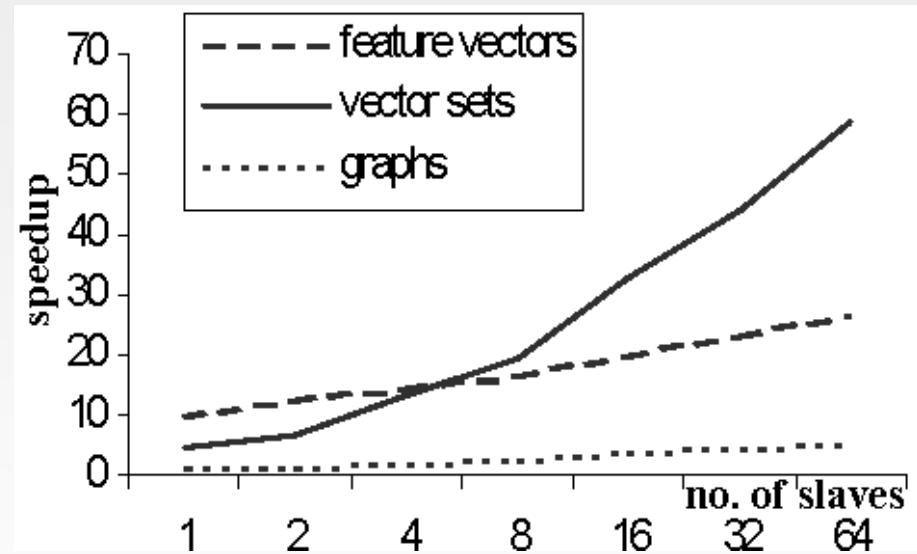
- Partitionierung enthält Übertragungskosten
- keine Übertragungskosten während des Clusterings

# Experimentelle Evaluation

## Beschleunigung bei unterschiedlicher Anzahl Slaves



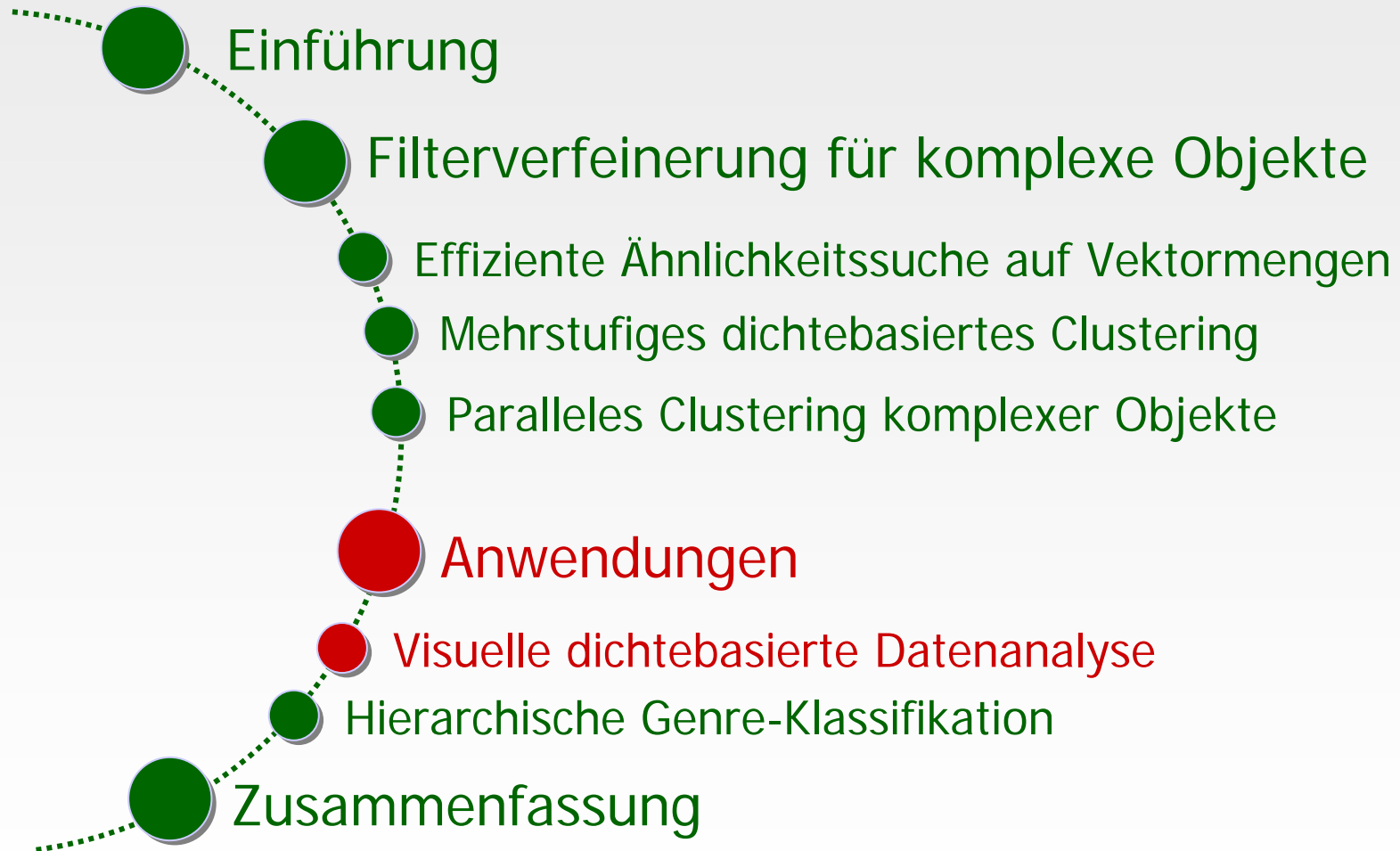
im Vergleich zu DBSCAN mit Full Table Scan



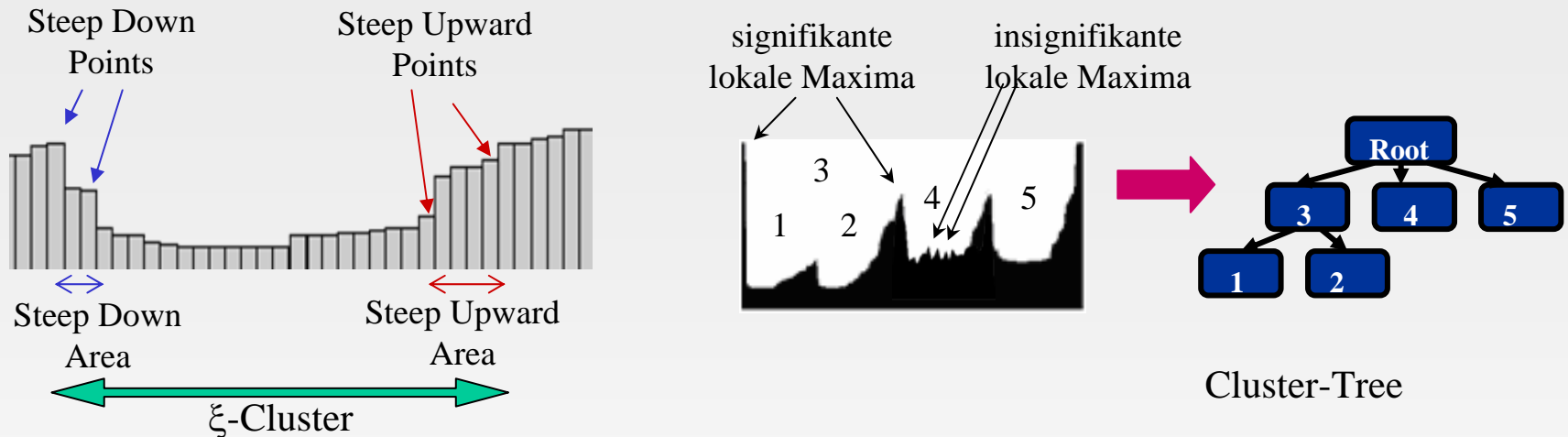
im Vergleich zu DBSCAN mit herkömmlicher mehrstufiger Anfragebearbeitung

# Überblick

---



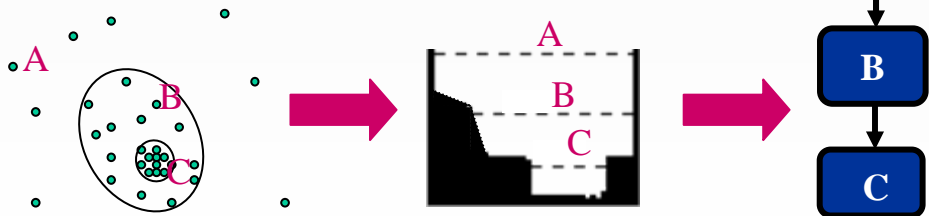
# Cluster-Erkennung



- $\xi$ -Clustering [Ankerst, Breunig, Kriegel, Sander: SIGMOD 99]  
Finde Cluster anhand steiler Bereiche
- Cluster-Tree [Sander, Qin, Lu, Niu, Kovarsky: PAKDD 03]  
Finde Cluster anhand signifikanter lokaler Maxima

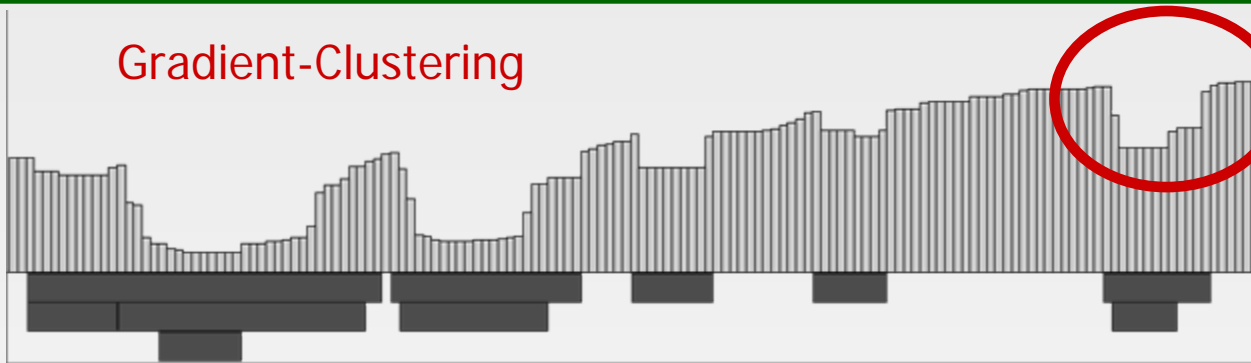
Problem: verschachtelte Cluster

Lösung: Gradient Clustering



# Cluster-Erkennung

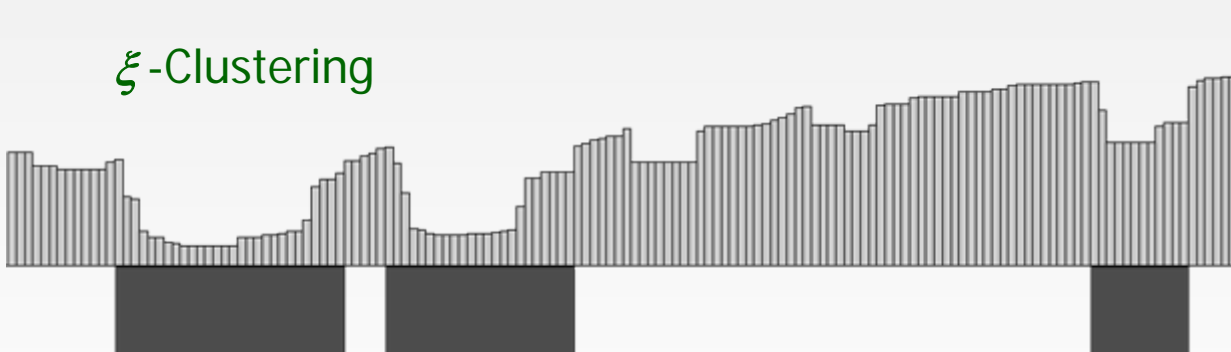
Gradient-Clustering



Erkennung  
verschachtelter  
Cluster

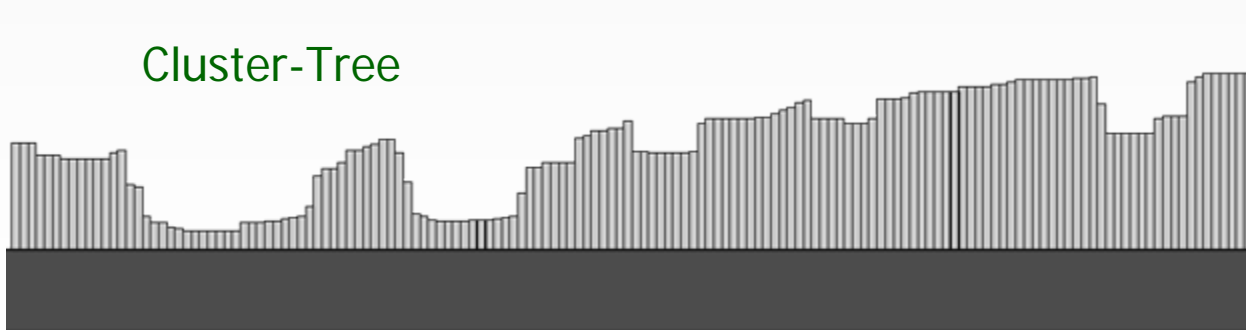
Viele Cluster und  
Subcluster erkannt

$\xi$ -Clustering



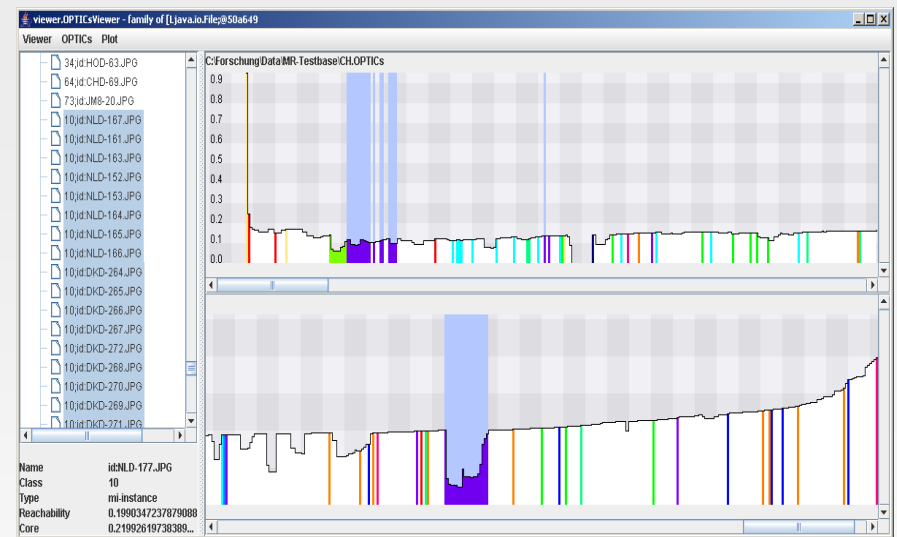
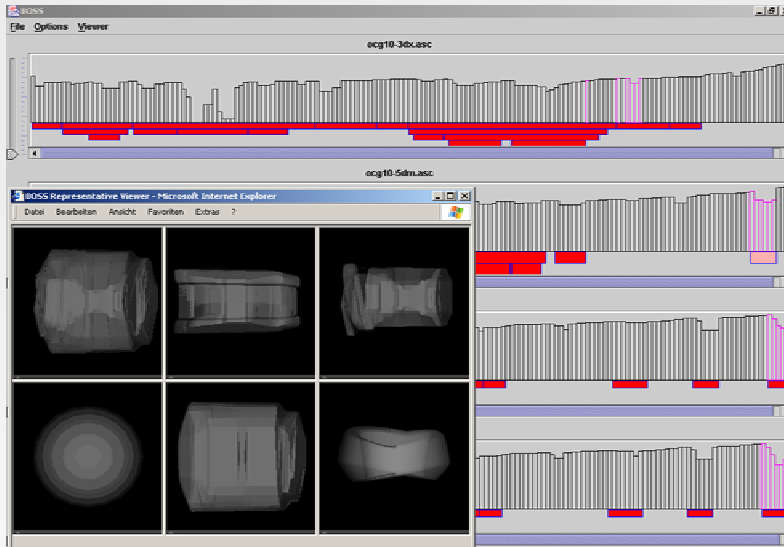
Einige Cluster  
erkannt

Cluster-Tree



Keine Cluster  
erkannt

# BOSS und VICO



## BOSS (Browsing OPTICS Plots for Similarity Search)

- Interaktive Analyse von OPTICS-Plots
- Cluster-Erkennung, Cluster-Repräsentanten

## VICO (Visualizing Connected Object Orderings)

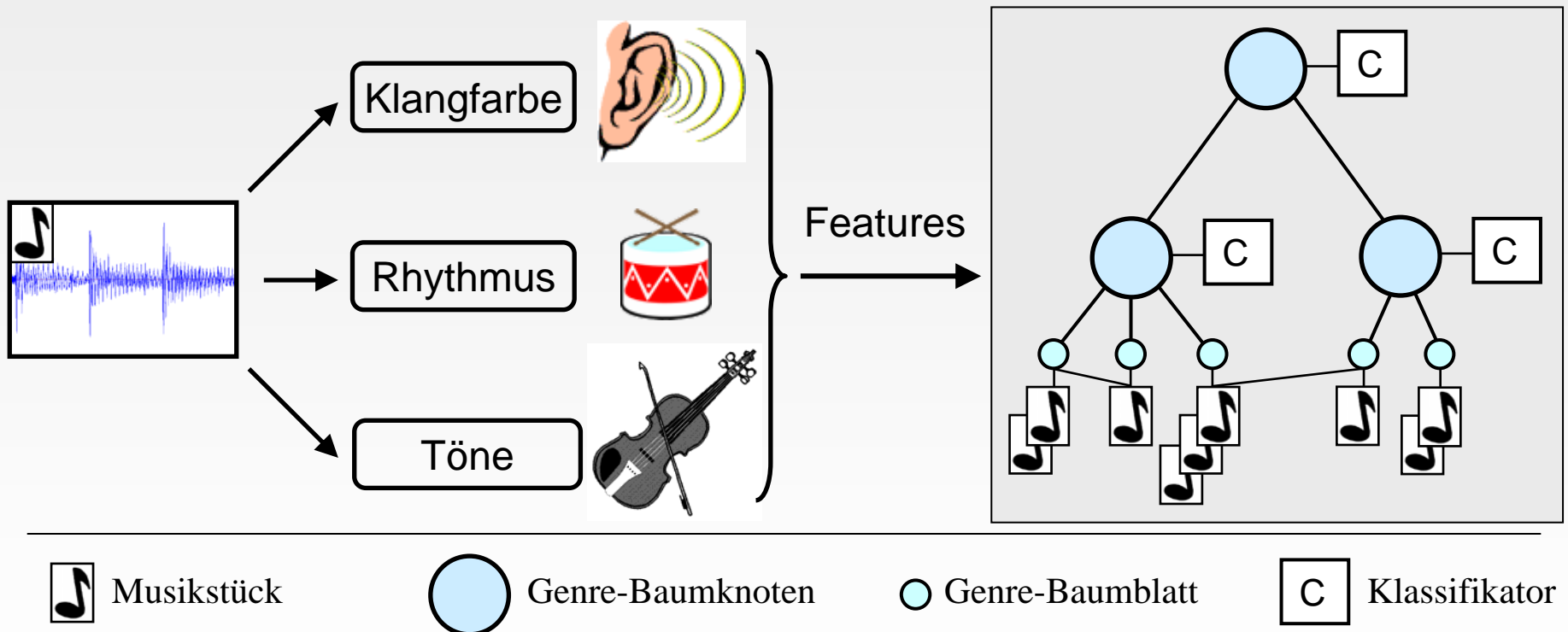
- Visuelle Analyse multirepräsentierter und multiinstanzierter Objekte

# Überblick

---

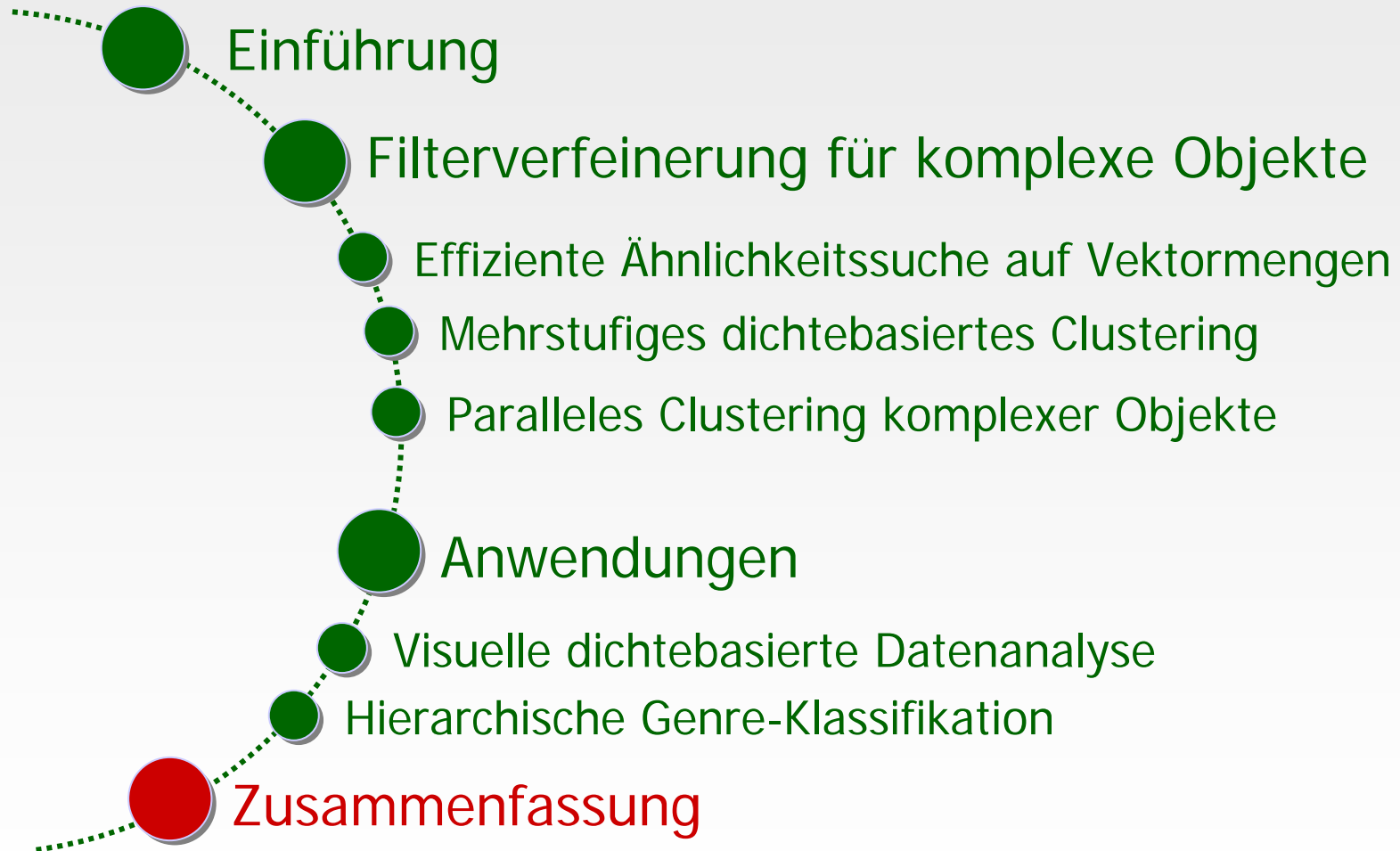


# Audioklassifikation



# Überblick

---



# Zusammenfassung

---

- Filter für Vektormengen [BTW 05]
  - Totale und partielle Ähnlichkeitsmaße für Vektormengen
  - Closest Pair-Filter, Norm-Filter, Zentroid-Filter
  
- Mehrstufiges dichtebasiertes Clustering [ICDM 04, PAKDD 06]
  - Integration mehrstufiger Anfragebearbeitung in DBSCAN und OPTICS
  - Parallelisierung von DBSCAN
  
- Anwendungen [SDM 04, EDBT 06, ICME 06]
  - Effektive Cluster-Erkennung mit Gradient Clustering
  - Prototypen BOSS, VICO und MUSCLE

# Ausblick

---

- Integration mehrstufiger Anfragebearbeitung in weitere Data Mining-Algorithmen, z.B. k-Medoid
- Parallele oder verteilte Version von OPTICS
- Qualitätsmaß für Cluster-Repräsentanten
- Genre-Klassifikation für weitere Medientypen, z.B. Video

---

Danke für die  
Aufmerksamkeit!