

E. Achtert, C. Böhm, S. Brecheisen, H.-P. Kriegel, P. Kunath,  
A. Pryakhin, M. Renz, M. Schubert

## Komplexe Objektbeschreibungen zur Suche in Multimedia-Datenbanken

### 1 Einleitung

Die inhaltsbasierte Suche nach Multimediale Daten, wie Fotos, Musikstücken oder Filmen, hat durch die jüngsten Entwicklungen in der Unterhaltungselektronik, der Datenübermittlung und der Computertechnologie an allgemeinem Interesse gewonnen. Während früher die Verwaltung großer Mengen von Multimediale Daten eher professionellen Anwendern vorbehalten war, existieren heute in vielen Haushalten mehrere Hundert Gigabyte an Fotos, digitalen Videos und Musikstücken. Die Zahl der potentiellen Anwender von Multimediale Datenbanken ist heute also deutlich größer als noch vor wenigen Jahren. Allerdings ergibt sich dadurch das Problem, die gesammelten Daten so zu verwalten, dass sie in dem riesigen Datenberg durch intuitiv bedienbare Systeme effizient gefunden werden können. Dabei sind auch die Anforderungen an die Suchsysteme drastisch gestiegen. Während man im professionellen Umfeld noch mit einem umfassend geschulten Anwender rechnen kann, müssen die Systeme für die Anwendung im Privatbereich wesentlich intuitiver zu bedienen sein, um sich hier durchsetzen zu können.

Der Bereich Multimedia wurde daher um eine Vielzahl neuer Forschungsgebiete erweitert, die weit über die Ähnlichkeitssuche in hochdimensionalen Feature-Vektoren hinausgehen. In modernen Multimediale Systemen werden die verwalteten Datenobjekte häufig durch mehrere Repräsentationen beschrieben, die sich wiederum aus einer Menge von Feature-Vektoren zusammensetzen können. Die dabei verwendeten Feature-Vektoren sind dabei deutlich niedrig-dimensionaler als bei der Modellierung durch einen einzelnen Feature-Vektor. Systeme zur inhaltsbasierten Bildsuche wie BlobWorld [CBGM02] stellen Bilder beispielsweise durch Farb-, Form und Texturvektoren dar. Die Ähnlichkeit zwischen zwei Bildern kann dann als Kombination dieser Kriterien betrachtet werden und ist dadurch meist deutlich

besser erfasst als in Systemen, die sich auf nur einen Typ von Eigenschaften festlegen.

Ein weiterer wichtiger Aspekt in der aktuellen Forschung ist die Bestimmung von Eigenschaften höherer Ordnung, sog. Highlevel-Features. Sie beschreiben den Inhalt von Multimediale Daten auf semantischer Ebene, z.B. mittels Schlagworten. Diese Objektbeschreibungen können für die inhaltsbasierte Suche in Bildern, Videos und Audiodateien verwendet werden. Mit Hilfe von Highlevel-Features können Anfragen wie zum Beispiel »suche Videos, in denen ein BMW zu sehen ist«, »suche Bilder, auf denen lachende Personen abgebildet sind« oder »suche Musikstücke aus dem Bereich Volksmusik« effizient und effektiv beantwortet werden.

Diese Entwicklung stellt eine extreme Herausforderung dar, sie erfordert neue Techniken zur Anfragebearbeitung und zur Verwaltung von Multimediale Objekten. In diesem Artikel wird insbesondere auf die Verwaltung von mengenwertigen, multirepräsentierten und unscharfen Objekten eingegangen. Solche Objektdarstellungen helfen, den Problemen der Vieldeutigkeit und der subjektiven Interpretation von Medieninhalten entgegenzuwirken und somit die semantische Lücke beim Multimediale Retrieval zu verkleinern. Im folgenden wird ein Überblick über mögliche Lösungsansätze für derartige Herausforderungen bei der Suche in Multimediale Datenbanken vorgestellt.

### 2 Multiinstanz-Objekte

Aufgrund der Komplexität heutiger Multimediale Daten ist es für viele Applikationen nützlich, Objekte durch eine Menge von Feature-Vektoren zu beschreiben. Ein Bild kann zum Beispiel mehrere Formen beinhalten. Die bisherigen Verfahren, die jedes Bild als einzelnen Vektor dargestellt haben, sind dazu ungeeignet. Es ist vielmehr sinnvoll, jede Form auf

einen einzelnen Formdeskriptor abzubilden und ein Bild durch die Menge der darin enthaltenen Formdeskriptoren zu beschreiben. Der Vergleich von zwei Bildern impliziert demnach den Vergleich zweier Mengen von Formdeskriptoren.

Weitere Anwendungsgebiete für multiinstanz bzw. mengenwertige Objektbeschreibungen sind unter anderem Videodaten, Musikdaten oder auch 3D-Daten wie CAD-Bauteile. Ein Film kann über die Menge der darin enthaltenen Kameraeinstellungen und Szenen beschrieben werden. Musikdaten können über die Menge der verschiedenen auftretenden Stimmen oder Instrumente beschrieben werden. CAD-Objekte können in eine Menge räumlicher Primitive zerlegt werden. Die Anzahl der Primitive, die ein Objekt bestmöglich beschreiben, kann von Objekt zu Objekt variieren. Deshalb bewirkt der Objektvergleich auf Basis von mengenwertigen Darstellungen häufig eine intuitivere Modellierung von Ähnlichkeit als der Vergleich von Darstellungen mit einem Vektor. Im Folgenden beschreiben wir daher Datenbanklösungen für den Vergleich und die effiziente Suche in mengenwertigen Daten.

**Mengenwertige Distanzfunktionen.** In [KBK+03] werden 3D-Objekte durch Mengen von Überdeckungssequenzen approximativ beschrieben, um sowohl effiziente als auch effektive Ähnlichkeitssuche in einer Datenbank von 3D-Objekten zu ermöglichen. Das zentrale Problem stellt dabei die Herleitung einer geeigneten Distanzfunktion auf Vektormengen dar. Zum einen soll die Distanzfunktion den gewünschten Ähnlichkeitsbegriff widerspiegeln, zum anderen sollen die Distanzberechnungen zwischen den 3D-Objekten mit vertretbarem Aufwand möglich sein. In der Literatur wurden bereits zahlreiche Distanzfunktionen für Vektormengen vorgeschlagen. Die Hausdorff-Distanz ist eine bekannte und einfach zu berechnende Metrik. Sie ist jedoch in diesem Anwendungsbereich als Ähnlichkeitsmaß ineffektiv, weil sie nicht alle Instanzen im Objekt ausreichend berücksichtigt. Somit bleibt die Gesamtstruktur der Vektormengen im Ähnlichkeitsmodell unberücksichtigt und die Information, die durch

Kombination verschiedener Distanzen einzelner Vektoren gewonnen werden kann, bleibt ungenutzt. Weitere Distanzfunktionen für Multiinstanz-Daten sind die Summe minimaler Distanzen, die Surjektionsdistanz sowie die Link-Distanz [EM97]. Diese Distanzfunktionen eignen sich zwar zur Ähnlichkeitsmodellierung, sind aber keine Metriken, weil sie die Dreiecksungleichung nicht erfüllen. Dieser Umstand macht ihre Anwendung zur Ähnlichkeitssuche in umfangreichen Multimedia-Datenbanken unattraktiv, da es sehr aufwendig ist, Ähnlichkeitsanfragen auf der Basis eines nicht-metrischen Distanzmaßes effizient zu bearbeiten. In [EM97] wird zwar gezeigt, wie die genannten Distanzmaße zu Metriken erweitert werden können, jedoch führt dieses Verfahren zu einer exponentiellen Berechnungskomplexität. Um eine effiziente Anfragebearbeitung zu ermöglichen, wurde daher in [KBK+03] eine metrische Distanzfunktion für Vektormengen vorgeschlagen, deren Berechnung auf dem Graphproblem der vollständigen Paarung mit minimalem Gewicht basiert. Um die Distanz zwischen je zwei durch Vektormengen repräsentierten Objekten zu bestimmen, dienen die beiden Vektormengen als Knotenmengen eines vollständigen bipartiten Graphen. Als Kantenkosten werden die euklidischen Distanzen zwischen den einzelnen Vektoren ermittelt. Ein effizienter Optimierungsalgorithmus bestimmt schließlich eine Kantenmenge, die jedem Element der Vektormenge des einen Objekts genau ein Element der Vektormenge des anderen Objekts zuordnet und die Summe der Kantenkosten minimiert. Das auf diese Weise bestimmte Distanzmaß ist metrisch. Daher können die Vektormengen zur Anfragebeschleunigung in einer metrischen Indexstruktur wie dem M-Baum [CPZ97] verwaltet werden. Eine andere Möglichkeit der Anfragebeschleunigung ist die Anwendung mehrstufiger Anfragebearbeitung mit Filterdistanzfunktionen, die weniger aufwendig zu berechnen sind und die exakte Distanzfunktion so gut wie möglich nach unten abschätzen. Eine solche Filterdistanz, die den Abstand der Mittelwertvektoren der Vektormengen verwendet, wird in [KBK+03] vorgestellt. Zwei weitere

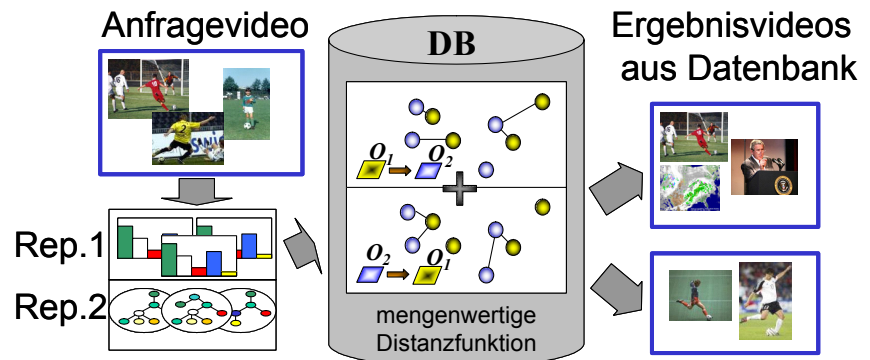


Abb. 1: Multi-repräsentierte Suche auf Videodaten

Filterdistanzen, die die euklidischen Normen der einzelnen Vektoren bzw. eine weniger strenge Zuordnung der Vektoren bei der Bestimmung der kostenminimalen Kantenmenge verwenden, werden in [BKP05] beschrieben und experimentell untersucht.

### 3 Multi-Repräsentierte Objekte

Zusätzlich zur inhaltsbasierten Suche nach ähnlichen Objekten in Multimedia-Datenbanken werden Objekte oftmals in Kategorien eingeteilt, um sie besser verwalten zu können. Für das Auffinden von ähnlichen Objekten genügt es, Objekte der gleichen oder ähnlicher Kategorien zu betrachten. Dazu müssen in einem Vorverarbeitungsschritt Kategorien ermittelt werden, die eine gewisse Anzahl ähnlicher Objekte enthalten, oder die vorhandenen Multimedia-Objekte müssen vordefinierten Kategorien zugeordnet werden. Diese vordefinierten Kategorien können dabei die persönlichen Vorlieben eines Benutzers widerspiegeln oder von Experten definiert werden. Zum rechnergestützten Auffinden von Gruppen ähnlicher Objekte bieten sich Clustering-Algorithmen an, während zur Zuordnung von Objekten zu vordefinierten Kategorien Klassifikationsalgorithmen Anwendung finden.

Wie bereits in der Einführung erwähnt, lassen sich Multimedia-Objekte durch multiple Repräsentationen darstellen. Zum Beispiel kann man Filme bzgl. ihres Tons oder ihrer Bilder beschreiben. Darüberhinaus kann ein Objektmerkmal durch verschiedene Feature-Repräsentationen beschrieben werden, z.B. kann ein Bild in einem Video durch Texturhisto-

gramme und Farbhistogramme beschrieben werden. Da Repräsentationen Beschreibungen unterschiedlicher Qualität für jedes einzelne Objekt liefern können, wird die Effektivität von Such-, Klassifikations- und Clustering-Algorithmen durch die Berücksichtigung von mehreren Repräsentationen signifikant verbessert. Dieser Abschnitt bietet einen Überblick über Such-, Klassifikations- und Clustering-Verfahren, die multi-repräsentierte Inhalte betrachten.

#### 3.1 Inhaltsbasierte Suche

Es existieren bereits einige Suchverfahren, die den multi-repräsentierten Charakter von Multimedia-Daten berücksichtigen. Die Autoren von [BKS+04] schlagen eine entropiebasierte Methode vor, um die Effektivität der Ähnlichkeitssuche auf 3D-Objekten unter Berücksichtigung von mehreren Repräsentationen zu erhöhen. Dieser Ansatz benutzt vordefiniertes Wissen in Form von Objekten mit einer vorgegebenen Kategoriezugehörigkeit. Eine interaktive Methode zum Fusionieren von mehreren Repräsentationen von Videodaten wird in [SJL+03] vorgestellt. Hier besteht sowohl die Möglichkeit die einzelnen Repräsentationen manuell zu kombinieren, als auch interaktiv in die Suche einzugreifen.

In [KKKP06] wird ein effizienter Ansatz präsentiert, der die Effektivität der Ähnlichkeitssuche durch Betrachtung von multiplen Repräsentationen auf Videodaten signifikant verbessert (vgl. Abbildung 1), wobei die einzelnen Repräsentationen als mengenwertige Objekte sind (vgl. Abschnitt 2). Filmdaten bestehen zunächst aus mehreren Tausend Einzelbildern, die in Feature-Vektoren, z.B.

Farbhistogramme, transformiert werden. Um eine schnelle Anfragebearbeitung zu gewährleisten, werden Multimedia-Objekte mit Hilfe von so genannten Summarisierungstechniken in eine kompakte Form gebracht. Beispielsweise kann die Anzahl der betrachteten Feature-Vektoren durch Clustering-Verfahren wie  $k$ -Means drastisch reduziert werden. Pro Repräsentation erhält man als Ergebnis eine Menge von Repräsentanten, die jeweils eine Gruppe ähnlicher Feature-Vektoren beschreiben. Um die Gesamtdistanz zwischen multi-repräsentierten Objekten zu berechnen, werden die Distanzen innerhalb der einzelnen Repräsentationen durch eine gewichtete Linearkombination miteinander verbunden. Als Distanzen bieten sich beispielsweise die in Abschnitt 2 vorgestellten Distanzfunktionen an. Zur Bestimmung der Gewichte innerhalb der Linearkombination stellen die Autoren vier Gewichtungsfunktionen vor.

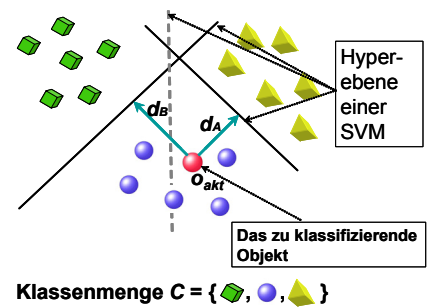
Die erste Gewichtungsfunktion nutzt die Tatsache aus, dass jeder Repräsentant nach der Summarisierung eine Menge von originalen Feature-Vektoren vertritt. Das Gewicht basiert auf dem relativen Anteil (bzgl. der Gesamtdatenmenge) der Feature-Vektoren, die durch einen Repräsentanten beschrieben werden. Die zweite Gewichtungsfunktion baut auf spezifischen Qualitätskriterien einer zugrunde liegenden Summarisierungstechnik auf, z.B. für  $k$ -Means-basierte Summarisierungstechniken kommt die Summe von quadrierten Distanzen zwischen einem Objekt und dem nächstliegenden Repräsentanten zum Einsatz. Die dritte Gewichtungsfunktion betrachtet die lokale Nachbarschaft eines Repräsentanten, um die Auswirkungen von Störeinflüssen, wie Rauschen zu vermeiden. Das Gewicht ist durch den relativen Anteil (bzgl. der Datenbankgröße) der originalen Vektoren in der lokalen Nachbarschaft eines Repräsentanten gegeben. Die vierte Gewichtungsfunktion betrachtet die Güte, mit der ein gegebener Repräsentant alle Feature-Vektoren eines Multimedia-Objektes darstellt. Diese Gewichtungsfunktion nimmt an, dass der Abstand zwischen einem Repräsentanten und jedem beliebigen Feature-Vektor eines Objektes eine gaußverteilte Zufallsvariable dar-

stellt. Das Gewicht ist durch die invertierte Entropie dieser Zufallsvariable definiert. Entropiewerte nahe eins bedeuten eine zufällige Verteilung der Abstände, eine niedrige Entropie heißt, dass die Vektoren eine geringe Distanz zum Repräsentanten aufweisen. Somit ist ein Repräsentant mit niedriger Entropie eine gute Darstellung seiner Feature-Vektoren.

### 3.2 Klassifikation

In [CVK04] wird ein Verfahren zur Klassifikation von Musikstücken vorgestellt, das auf  $k$ -nächster Nachbar Klassifikation und neuronalen Netzen basiert. Dabei werden Musikstücke in die Kategorien »Rock« und »Klassik« eingeteilt. [XMS+03] beschreibt ein Verfahren zur Genre-Klassifikation mit *Support Vector Machines* (SVMs). Diese lernen die optimalen Klassengrenzen zwischen den verschiedenen Musikgenres anhand von Trainingsdaten. Das in [Zha03] vorgeschlagene Verfahren ordnet Musikstücke den Kategorien einer Klassenhierarchie zu. Die hierarchische Organisation der Klassen erlaubt dem Benutzer einen einfachen Zugang zu den verwalteten Musikstücken. Allgemeinere Klassen werden dabei in speziellere Unterklassen unterteilt, was dem Benutzer eine top-down Navigation ermöglicht. Ein Nachteil der vorgeschlagenen Methode ist eine fest vorgegebene Klassenhierarchie, die nur eine eingeschränkte Unterstützung für benutzerdefinierte Kategorisierung zulässt.

In [BKPP06] wird eine flexiblere Methode zur hierarchischen Klassifikation vorgeschlagen, die mehrere Repräsentationen verwendet, um die Genauigkeit der Klassenzuordnung zu erhöhen. Dabei wird für jede Vaterklasse in der benutzerdefinierten Hierarchie ein Klassifikator trainiert, der die Musikstücke dieser Klasse der jeweils wahrscheinlichsten Unterklasse zuordnet. Ein Musikstück ist durch eine Menge von Feature-Vektoren in jeder Repräsentation gegeben, z.B. wird die Klangfarbe durch 30 bis 200 Feature-Vektoren pro Sekunde beschrieben. Um eine effiziente Anfragebearbeitung zu gewährleisten, wird die große Anzahl von Instanzen in jeder Repräsen-



$d_A, d_B$ : Abstände zwischen  $o_{akt}$  und Klassengrenze

Abb. 2: Bestimmung der Gewichte für objektgerichtete Gewichtung

tation im ersten Schritt deutlich reduziert. Zur Berechnung dieser kompakten Form wird die Vektorenmenge  $s$  gebildet, indem  $k$  Vektoren aus der Vereinigungsmenge aller Objektbeschreibungen ausgewählt werden. Die Vektoren in  $s$  können zufällig selektiert werden oder durch Repräsentanten berechnet werden, die durch ein partitionierendes Clustering-Verfahren, wie  $k$ -Means, berechnet worden sind.

Zur kompakten Beschreibung eines Musikstücks wird ein Vektor  $v$  berechnet, dessen Dimensionalität der Anzahl der Vektoren in  $s$  entspricht. Für die Berechnung der  $i$ -ten Koordinate bestimmt man die Abstände des  $i$ -ten Vektors in  $s$  zu den Instanzen der Objektbeschreibung. Die kleinste dieser Distanzen wird dann als Vektorkoordinate verwendet. Der resultierende Vektor  $v$  kann immer noch eine hohe Anzahl von Dimensionen aufweisen, außerdem sind nicht alle Dimensionen für alle Teilklassifikationsprobleme relevant. Daher wird in jedem inneren Knoten der Klassenhierarchie ein zusätzlicher Reduktionsschritt durchgeführt. Dieser Schritt kalkuliert zuerst eine Diskretisierung der Wertebereiche für jede Dimension in  $v$ . Anschließend werden  $n$  Dimensionen anhand ihres Informationsgewinns (Information Gain) ausgewählt.

An jedem inneren Knoten wird ein Klassifikator eingesetzt, der auf SVMs aufbaut. Um multiple Repräsentationen zu handhaben, wird eine *objektgerichtete Gewichtung* [KKPS04a] verwendet, deren grundlegende Idee in Abbildung 2 dargestellt ist. Zur Abschätzung der Klassifikationsgüte wird die Beobachtung ausgenutzt, dass ein gegebenes Objekt

genau dann durch die gegebenen Repräsentation gut beschrieben wird, wenn es in dieser Repräsentation eine ausreichend große Distanz zur Klassengrenze aufweist. Wie in Abbildung 2 dargestellt, wird das Gewicht des zu klassifizierenden Objektes  $o_{akt}$  durch die kleinste Distanz  $d_A$  zu einer der Hyperebenen bestimmt.

### 3.3 Clustering

Ein Spektralclustering-Verfahren für multi-repräsentierte Objekte wird in [DeS05] vorgestellt. Die Zuordnung von Objekten zu Clustern ist so realisiert, dass der Widerspruch zwischen den Clustermodellen in den einzelnen Repräsentationen minimiert wird. Die Autoren von [BS04] erweitern Expectation Maximization Clustering und agglomeratives Clustering auf multi-repräsentierte Objekte. Allerdings sind alle bisher betrachteten Clustering-Methoden für eine große Anzahl von Repräsentationen mit verschiedenen Semantiken ungeeignet.

In [KKPS04b] wird daher ein dichte-basiertes Clustering-Verfahren für multi-repräsentierte Objekte beschrieben, das zwei Methoden zur Handhabung von multiplen Repräsentationen verwendet: Vereinigung und Überschneidung. Beide Methoden stellen zwei grundlegende Semantiken für die Kombination multipler Repräsentationen dar. Da die einzelnen Repräsentationen häufig kein perfektes Modell der intuitiven Ähnlichkeit definieren, entspricht ein kleiner Wert der Ähnlichkeitsfunktion nicht zwangsläufig einer echten Ähnlichkeit. Daher werden zwei unterschiedliche Typen von Repräsentationen unterschieden. Kleine Distanzen in Repräsentationen des ersten Typs implizieren relativ sicher Ähnlichkeit, aber als unähnlich eingestufte Objekte können tatsächlich ähnlich sein. Textannotationen sind ein gutes Beispiel für diesen ersten Typ. Objekte, die in ähnlichen Worten beschrieben werden, sind höchstwahrscheinlich auch tatsächlich ähnlich. Andererseits müssen tatsächlich ähnliche Objekte nicht unbedingt mit ähnlichen Worten beschrieben werden. In Repräsentationen des zweiten Typs wird Unähnlichkeit relativ sicher erkannt, aber als ähnlich eingestufte Objekte können

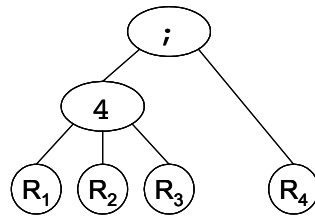


Abb. 3: Kombinationsbaum

tatsächlich unähnlich sein. Ein Beispiel für den zweiten Typ ist der Featureerraum der Farbhistogramme. Unterschiedliche Farbhistogramme zweier Bildern implizieren relativ sicher Unähnlichkeit. Andererseits sind Bilder, die sehr ähnliche Farbhistogramme haben, nicht zwingenderweise ähnlich. Ein klassisches Beispiel hierfür ist der Vergleich zwischen den Farbhistogrammen von Billardtisch und Blumenwiese. Repräsentationen des ersten Typs werden so kombiniert, dass eine Ähnlichkeit in einer der Repräsentationen ausreicht, um Ähnlichkeit zu implizieren (Vereinigung). Für die Kombination von Repräsentationen des zweiten Typs wird dagegen verlangt, dass alle Repräsentationen die Ähnlichkeit zwischen zwei Objekten feststellen (Überschneidung).

Dieses Verfahren wird in [AKPS06] auf die Verwendung von Kombinationen beider Semantiken und das Auffinden von hierarchischen Clustern erweitert. Um multiple Repräsentationen zu vereinigen, wird ein so genannter Kombinationsbaum auf folgende Weise konstruiert: (1) Blattknoten enthalten einzelne Repräsentationen. (2) Die inneren Knoten können entweder einen Vereinigungsoperator oder einen Überschneidungsoperator enthalten. (3) Sohnknoten eines inneren Knotens mit einem Vereinigungsoperator sind Repräsentationen des ersten Typs. (4) Sohnknoten eines inneren Knotens mit einem Überschneidungsoperator sind Repräsentationen des zweiten Typs. Abbildung 3 zeigt ein Beispiel für einen Kombinationsbaum. Basierend auf Kombinationsbäumen wird in [AKPS06] eine Distanz auf multiplen Repräsentationen definiert. Darüberhinaus wird der hierarchische dichte-basierte Clustering-Algorithmus OPTICS [ABKS99] auf multiple Repräsentationen erweitert, was das automatische Erzeugen von Cluster-Hierarchien ermöglicht.

## 4 Reverse Nearest Neighbor Anfragebearbeitung

Die steigende Komplexität der Daten erfordert nicht nur neuartige Objektrepräsentationen, sondern impliziert auch neue Anforderungen im Bereich der Anfragebearbeitung. Ein Beispiel hierfür wären Anbieter von Handy-Klingeltönen, Musikstücken oder Spielen, deren Angebotspalette sich fortwährend erweitert. Zur Gewährleistung von effizienten und effektiven Vertriebsaktivitäten ist es sinnvoll, nur diejenigen Kunden über ein neues Produkt zu informieren, die aufgrund ihrer bisher erworbenen Produkte wahrscheinlich das größte Interesse an diesem neuen Produkt haben werden. Hierzu benötigt man alle Kunden aus der Datenbank, für die das neue Produkt in der  $k$ -Nächste Nachbarn ( $k$ NN) Menge ihrer bereits erworbenen Produkte enthalten ist. Dieser Anfragetyp wird als sogenannte Reverse  $k$ -Nearest Neighbor ( $Rk$ NN) Anfrage bezeichnet. Dabei werden diejenigen Objekte in einer Datenbank gesucht, in deren  $k$ NN-Mengen ein gegebenes Anfrageobjekt  $q$  enthalten ist. Ziel einer  $Rk$ NN-Anfrage ist es, den Einfluss eines Anfrageobjekts auf die Gesamtdatenmenge zu untersuchen.

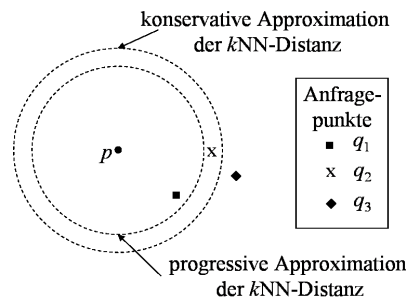
Der naive Ansatz zur Beantwortung einer  $Rk$ NN-Anfrage ermittelt die  $k$  nächsten Nachbarn aller  $n$  Datenbankobjekte und besitzt somit einen Zeitaufwand von  $O(n^2)$ . In den letzten Jahren wurde eine Vielzahl effizienter Methoden zur  $Rk$ NN-Suche veröffentlicht. Ein Großteil dieser Algorithmen, wie z.B. [KM00, YL01], basiert auf vorberechneten  $k$ NN-Distanzen und ist daher nur für einen im Vorfeld fest vorgegebenen Parameter  $k$  konzipiert. In [TPL04] wurde ein allgemeinerer Ansatz vorgestellt, der  $Rk$ NN-Anfragen für beliebige Parameter  $k$  effizient bearbeitet. Dieser Ansatz basiert allerdings auf Voronoi-Zellen, die nur in euklidischen Vektorräumen existieren. Wie bereits erwähnt ist es aber häufig vorteilhaft, auch andere Distanzmetriken zu verwenden. Zum Beispiel lassen sich die im vorherigen Abschnitt vorgestellten mengenwertigen Objektdarstellungen nicht direkt mit der euklidischen Distanz vergleichen. Die Nachteile dieser Algorithmen eliminiert der in

[ABK+06] vorgestellte Ansatz, der sowohl auf generellen metrischen Datenbanken angewendet werden kann und somit auch für Multiinstanz-Objekte geeignet ist, als auch dynamische Parameter  $k$  zulässt und im folgenden näher erläutert wird.

**Filter- und Verfeinerungs-Architektur.** Der konzeptionelle Ansatz von [ABK+06] besteht in einer Filter- und Verfeinerungs-Architektur für die  $RkNN$ -Anfragebearbeitung. Mittels approximierter  $kNN$ -Distanzen wird in einem Filterschritt zwischen drei Kategorien von Datenbankobjekten unterschieden: Objekte, die mit Sicherheit zur Ergebnismenge gehören, Objekte, die sicher von der Ergebnismenge ausgeschlossen werden können und sogenannte Kandidaten, für die der Filter keine eindeutige Aussage treffen kann. Für diese Kandidaten muss die Zugehörigkeit zum Ergebnis in einem Verfeinerungsschritt noch überprüft werden.

Die Approximation der  $kNN$ -Distanzen erfolgt über zwei Funktionen  $c$  und  $p$ . Für jedes Objekt  $o$  werden zwei Funktionen  $c_o$  bzw.  $p_o$  gespeichert, welche die  $kNN$ -Distanzen von  $o$  konservativ bzw. progressiv approximieren. Die konservative Approximation  $c_o(k)$  ist stets kleiner oder gleich der tatsächlichen  $kNN$ -Distanz des Objekts  $o$ , während die progressive Approximation  $p_o(k)$  stets größer oder gleich der tatsächlichen  $kNN$ -Distanz von  $o$  ist.

Diese beiden Approximationen erlauben die effiziente Identifikation von Objekten, die mit Sicherheit von der Ergebnismenge einer  $RkNN$ -Anfrage ausgeschlossen werden können bzw. die mit Sicherheit in der Ergebnismenge enthalten sind. Objekte deren konservativ approximierter  $kNN$ -Distanz größer ist als ihre Distanz zum Anfrageobjekt, können per Definition kein  $RkNN$  des Anfrageobjekts sein und sind damit nicht Teil des Ergebnisses. Andererseits gilt für Objekte, deren progressiv approximierter  $kNN$ -Distanz kleiner ist als ihre Distanz zum Anfrageobjekt, dass sie in der  $RkNN$ -Menge des Anfrageobjekts und somit in der Ergebnismenge enthalten sind. Lediglich für die verbleibenden Objekte, deren Distanz zum Anfrageobjekt zwi-



**Abb. 4: Konservative und progressive Approximation**

schen konservativer und progressiver Approximation liegt, muss in einem Verfeinerungsschritt eine  $kNN$ -Anfrage gestellt werden, um ihre Ergebniszugehörigkeit zu überprüfen.

Abbildung 4 veranschaulicht das Konzept der konservativen und progressiven Approximation. Ist  $q_1$  das Anfrageobjekt, so ist das Objekt  $p$  mit Sicherheit ein Ergebnis der  $RkNN$ -Anfrage. Für den Fall, dass  $q_3$  das Anfrageobjekt ist, ist das Objekt  $p$  mit Sicherheit kein Ergebnis der  $RkNN$ -Anfrage. Nur im Falle, dass die  $RkNN$ -Anfrage mit  $q_2$  als Anfrageobjekt gestellt wird, muss das Objekt  $p$  noch einen Verfeinerungsschritt durchlaufen.

**Berechnung der Approximationen.** Die Approximation sowohl der konservativen als auch der progressiven  $kNN$ -Distanzen erfolgt mit Hilfe der Theorie über Selbstähnlichkeit [Sch91]. Hierbei wird angenommen, dass die  $kNN$ -Distanzen dem Potenzgesetz folgen und annähernd eine Gerade im logarithmischen Raum bilden. Für jede Approximation der  $kNN$ -Distanzen wird daher eine Funktion herangezogen, die einen linearen Verlauf im logarithmischen Raum besitzt, was einer Parabel im nicht-logarithmischen Raum entspricht. Für die Approximation der  $kNN$ -Distanzen eines Objekts müssen daher lediglich zwei Parameter gespeichert werden, die Steigung  $m$  und der Achsenabschnitt  $t$  der Approximationsgeraden. In einem Vorverarbeitungsschritt werden die  $kNN$ -Distanzen der Objekte für  $k \leq k_{max}$  berechnet und danach die konservativen und die progressiven Approximationsgeraden mit Hilfe von Verfahren der Optimierungstheorie berechnet.

**Aggregation der Approximationen.** Zur effizienten  $RkNN$ -Anfragebearbei-

tung auf metrischen Datenbanken kann z.B. ein M-Baum [CPZ97] entsprechend erweitert werden. Für jeden Knoten werden dabei die Maxima aller konservativen Approximationen und die Minima aller progressiven Approximationen der Sohnknoten aggregiert. Diese aggregierten Approximationen werden dabei wiederum durch Funktionen repräsentiert. Während der Anfragebearbeitung können beginnend bei der Wurzel diejenigen Knoten von der Suche ausgeschlossen werden, deren approximierte  $kNN$ -Distanzen größer sind als die Distanz zwischen Anfrageobjekt und dem jeweiligen Knoten. Dieser Ansatz ist auf jede baumartige Indexstruktur, wie z.B. den  $R^*$ -Baum [BKSS90] oder den X-Baum [BKK96] übertragbar.

## 5 Unschärfe Objekte

In vielen Multimedia-Anwendungsgebieten wie Objektverfolgung oder Objektidentifikation in Videos lassen sich Objekte oftmals nicht exakt beschreiben und sollten daher unscharf dargestellt werden. Ein weiterer Vorteil unscharfer Objektdarstellungen liegt in der kompakten Repräsentation, z.B. lässt sich eine grosse Mengen von Pixeln mittels einer einzigen Gausskurve beschreiben. Zur Darstellung von unscharfen Objekten kann man entweder Konfidenzintervalle für die Feature-Werte angeben, oder die Auftrittswahrscheinlichkeiten der Feature-Werte mittels Dichtefunktionen beschreiben.

Die Verwaltung von unscharfen Objekten und die Unterstützung wahrscheinlichkeitsbasierter Anfragen in Datenbanken mit unscharfen Objekten stellt ein aktives Forschungsgebiet im Multimediabereich dar, einen umfassenden Überblick findet man in [AHV95] und [Mot95]. In den letzten Jahren wurden viele Arbeiten im Bereich der Verwaltung und Anfragebearbeitung von unsicheren Daten veröffentlicht, z.B. in Sensordatenbanken [CKP03] und speziell in Umgebungen mit bewegten Objekten [CKP04, WSCY99]. Ähnlich zu dem Ansatz in [KKPR06] modellieren die Verfahren in [CKP03, CKP04, WSCY99] die Unsicherheit durch Wahrscheinlichkeitsdichtefunktionen (PDFs) über den Featurewerten. Die meisten dieser Ansätze

ze beruhen auf einer Diskretisierung der Wahrscheinlichkeits-Dichtefunktion, die dann in einer Indexstruktur für räumliche Objekte approximiert abgespeichert wird.

### 5.1 Der Gauss-Baum

Sowohl durch die Diskretisierung als auch durch die Approximation der Wahrscheinlichkeits-Dichtefunktion entsteht ein Genauigkeitsverlust. Der folgende Ansatz vermeidet dieses Problem, indem er nicht die Wahrscheinlichkeits-Dichtefunktion selbst, sondern vielmehr deren Parameter indexiert. In [BPS06a, BPS06b] werden für diesen Zweck multivariate Normalverteilungen (d.h. Gauss-Kurven) verwendet, die durch einen Mittelwert- ( $\mu_i$ ) und Varianz-Vektor ( $\sigma_i$ ) definiert sind. Der Varianzwert  $\sigma_i$  gibt hierbei über die Exaktheit bzw. Schärfe des Merkmalswertes  $\mu_i$  Auskunft. Die um die oben angesprochenen Unsicherheits- bzw. Varianz-Werte ergänzten Feature-Vektoren, die somit multivariate Normalverteilungen repräsentieren, nennt man *probabilistische Feature-Vektoren* (PFV). Eine Datenbank besteht aus einer großen Menge solcher PFVs. Als Anfrageobjekte dienen in [BPS06a] sowohl herkömmliche als auch probabilistische Feature-Vektoren. Zur Ermittlung derjenigen Objekte aus der Datenbank, die mit hoher Wahrscheinlichkeit mit dem Anfrageobjekt übereinstimmen, existieren zwei Arten von sogenannten Identifikationsanfragen. Bei einer sog. *Probabilistic Range Query* wird hierfür vom Benutzer eine Grenzwahrscheinlichkeit angegeben. Es werden also diejenigen Objekte ermittelt, die mit dem Anfrageobjekt mit einer Wahrscheinlichkeit von mindestens  $x\%$  übereinstimmen. Bei einer *k-Maximum-Likelihood Query* werden dagegen die  $k$  Objekte ausgegeben, welche die höchste Trefferwahrscheinlichkeit aufweisen. Bei allen diesen Anfragetypen legen wir einen Bayes-Ansatz zugrunde.

Das Verfahren in [BPS06b] kann zusätzlich auch Bereichsanfragen beantworten, die in Anwendungen wie z.B. bei bewegten Objekten mit unbekanntem Bewegungsmuster relevant sind. Eine Anfrage definiert hier für jedes Merkmal ein Intervall mit unterer und oberer Grenze

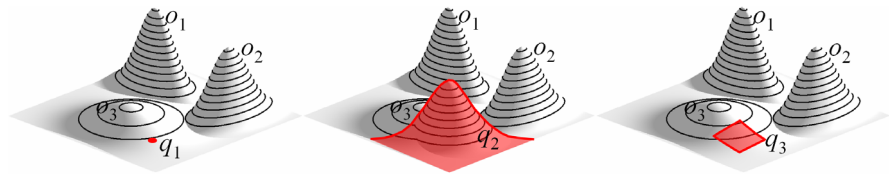


Abb. 5: Punkt-, PVF-, und Bereichsanfragen auf unscharfen Objekten

und ermittelt diejenigen Objekte, die sich mit hinreichender bzw. höchster Wahrscheinlichkeit in dem definierten Bereich befinden. Diese drei Anfragetypen sind in Abbildung 5 gemeinsam mit drei Datenobjekten ( $o_1, o_2, o_3$ ) in einem zweidimensionalen Merkmalsraum visualisiert.

Zur effizienten Anfragebearbeitung bei den oben genannten Anfragetypen wird eine hierarchische Indexstruktur eingesetzt. Typischerweise diskretisieren andere Verfahren zunächst die Wahrscheinlichkeitsdichtefunktionen der gespeicherten Objekte, um sie dann approximiert in einer geeigneten räumlichen Index-Struktur abzuspeichern. Das Gauss-Baum-Verfahren verfolgt dagegen den Ansatz, die Parameter der Verteilungsfunktion (im Fall von multivariaten Normalverteilungen also den Mittelpunkt und die Varianzen) in einer Indexstruktur zu verwalten. Die Dimensionalität des zu indexierenden Raumes verdoppelt sich also gegenüber der ursprünglichen Zahl  $d$  von Merkmalen.

Der Gauss-Baum ist analog zu räumlichen Indexstrukturen wie dem R-Baum als ausgeglichener Suchbaum definiert. Jeder Seite bzw. jedem Knoten des Baumes wird eine rechteckige Region MUR (minimal umgebendes Rechteck) des Parameterraumes zugeordnet. Ein MUR ist durch eine untere und obere Grenze für jeden Parameter definiert, wobei die unteren Grenzen mit  $\underline{\mu}_i$  bzw.  $\underline{\sigma}_i$  und die oberen Grenzen mit  $\bar{\mu}_i$  bzw.  $\bar{\sigma}_i$  bezeichnet werden.

Es ist jedoch nicht möglich, direkt in dem zugeordneten neuen Merkmalsraum

mit doppelter Dimensionalität eine Ähnlichkeitssuche (etwa mit euklidischer Distanz) durchzuführen. Die Anfragealgorithmen müssen geeignet angepasst werden um der unterschiedlichen Rolle der Verteilungsparameter  $\mu_i$  und  $\sigma_i$  ( $1 \leq i \leq d$ ) gerecht zu werden.

Für die Anfragebearbeitung ist von besonderer Bedeutung zu ermitteln, welche maximale Wahrscheinlichkeitsdichte eine beliebige, in einem solchen Knoten gespeicherte Gauss-Kurve an einem beliebigen Punkt  $x$  des Merkmalsraums aufweisen kann. Diese Grenzdichte  $\bar{N}_{MUR}(x)$  bildet die wesentliche Grundlage, um einen Knoten bzw. den gesamten darunter liegenden Teilbaum von der weiteren Anfragebearbeitung auszuschließen. Da in dem Teilbaum nur solche Gauss-Kurven  $N_{\mu_i, \sigma_i}(x)$  gespeichert sind, deren Parameter durch die Ungleichungen  $\underline{\mu}_i \leq \mu_i \leq \bar{\mu}_i$ ,  $\underline{\sigma}_i \leq \sigma_i \leq \bar{\sigma}_i$  eingeschränkt sind, ergibt sich:

$$N_{MUR}(x) = \max\{N_{\mu_i, \sigma_i}(x)\}, \forall (1 \leq i \leq d)$$

$$\underline{\mu}_i \leq \mu_i \leq \bar{\mu}_i$$

$$\underline{\sigma}_i \leq \sigma_i \leq \bar{\sigma}_i$$

Dies lässt sich durch eine Unterscheidung zwischen sieben verschiedenen Fällen sehr effizient analytisch ermitteln [BPS06a]. Abbildung 6 zeigt ein Beispiel eines eindimensionalen Merkmalsraums mit einem MUR = (3.0, 4.0, 0.6, 0.9) und einigen PFVs. Die Funktion  $\bar{N}_{MUR}(x)$  ist das Maximum aller Wahrscheinlichkeitsdichtefunktionen, die theoretisch in einem MUR gespeichert werden können und stellt insbesondere auch eine obere Schranke für die Dichten aller tatsächlich gespeicherten PVFs dar (A-E in Abbil-

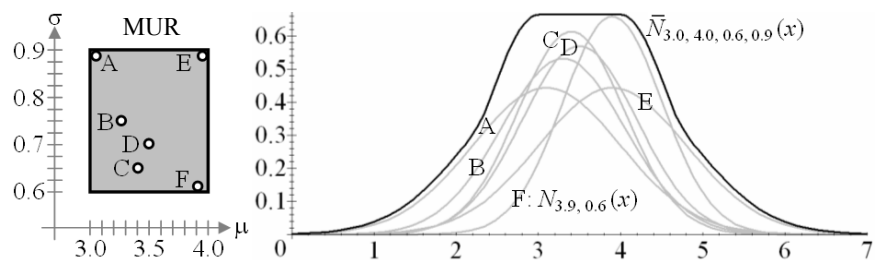


Abb. 6: Die Hüllfunktion  $\bar{N}_{MUR}(x)$

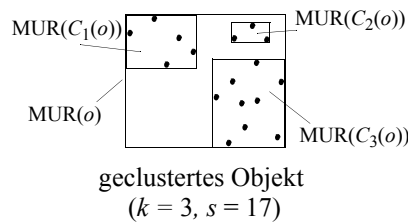
dung 6). Aus  $\bar{N}_{MUR}(x)$  lässt sich über den Satz von Bayes unmittelbar eine Wahrscheinlichkeit schätzen, wenn das Anfrageobjekt ein herkömmlicher Feature-Vektor  $x$  (d.h. ein Punkt im  $d$ -dimensionalen Merkmalsraum) ist. Für probabilistische Feature-Vektoren als Anfrageobjekte wird in [BPS06a] ebenfalls eine effiziente analytische Lösung hergeleitet, die sich aus der Addition der Varianzen von Anfrage- und Datenobjekten ergibt. Darüberhinaus behandelt [BPS06b] eine effiziente Lösung für Intervalle als Anfrageobjekte.

### 5.2 Unscharfe Join-Anfragen

Ein Join bildet Paare von Tupeln aus zwei Relationen  $R$  und  $S$ , die das Join-Prädikat erfüllen. Beim Ähnlichkeits-Join basiert das Join-Prädikat auf der Ähnlichkeit zwischen den Objekten, die in den Relationen gespeichert sind. Eine gängige Methode ist, die Objekte mit Feature-Vektoren zu beschreiben und die Ähnlichkeit zwischen zwei Objekten durch eine Distanzfunktion  $d: O \times O \rightarrow IR_0^+$  zu bewerten, z.B. die euklidische Distanz. Eine der wichtigsten Ähnlichkeits-Join Operationen ist der Distance-Range Join  $R \bowtie S$ . Er berechnet die Menge aller Paare, deren Ähnlichkeits-Distanz einen gegebenen Parameter  $\varepsilon$  nicht überschreitet.

Die meisten Arbeiten zur effizienten Bearbeitung von Join-Anfragen beziehen sich auf den Schnitt von ausgedehnten räumlichen Objekten [HJR97, LR96, PD96]. Eine häufig verwendete Technik ist der R-Baum basierte Spatial Join (RSJ) [BKS93]. Dieser Algorithmus durchläuft parallel R-Baum ähnliche Indexstrukturen, die auf den Relationen  $R$  und  $S$  angelegt wurden. Für jedes zu betrachtende Paar von Directoryseiten  $(P_R, P_S)$  bildet der Algorithmus alle Paare von Sohnseiten von  $(P_R, P_S)$ , die das Join-Prädikat erfüllen. Für die resultierenden Paare wird der Algorithmus rekursiv aufgerufen d.h. die korrespondierenden Indexstrukturen werden mittels Tiefendurchlauf abgearbeitet.

**Ähnlichkeits-Join unscharfer Objekte.** Um einen Join von unsicheren Objekten mit Hilfe von traditionellen Join-



**Abb. 7: Sample-basierte Approximation von unscharfen Objekten**

verfahren durchzuführen, muss die Ähnlichkeit zweier Objekte durch genau einen Wert beschrieben werden. Das bedeutet, dass die komplette wahrscheinlichkeitsbehaftete Distanzinformation in einen numerischen Wert aggregiert wird, was offensichtlich einen Informationsverlust nach sich zieht [KKPR06].

Der wahrscheinlichkeitsbasierte Ähnlichkeits-Join basiert auf der direkten Integration einer wahrscheinlichkeitsbehafteten Distanzfunktion, anstatt mit aggregierten Werten zu arbeiten. Jedem Objekt-paar wird dabei ein Wert zugewiesen, der die Wahrscheinlichkeit widerspiegelt, dass es zur Ergebnismenge gehört. Objekt-paare mit einer positiven Wahrscheinlichkeit werden zusammen mit dem jeweiligen Wahrscheinlichkeitswert als Ergebnis ausgegeben.

Obwohl es für manche Objektdarstellungen möglich ist, die wahrscheinlichkeitsbasierte Ähnlichkeit zweier unscharfer Objekte mittels deren PDFs zu berechnen, ist es oft günstiger stattdessen Monte-Carlo Sampling einzusetzen. Dies trifft vor allem dann zu, wenn die unscharfen Objekte bereits durch diskrete Wahrscheinlichkeitsdichtefunktionen beschrieben sind. In einem solchen Fall hat man die Samplepunkte unmittelbar zur Verfügung, anderfalls muss man diese Punkte durch Abtasten der Dichtefunktion gewinnen. Im folgenden wird davon ausgegangen, dass jedes Objekt  $o$  durch eine Menge von  $s$  Samplepunkten beschrieben wird, d.h.  $s$  viele Repräsentationen verkörpern  $o$ .

Um den Berechnungsaufwand des Joins zu reduzieren ist es günstig, mit Gruppen von Samples zu arbeiten. Zwei Sample-Punkte  $o_i$  und  $o_j$ , die zum selben Objekt  $o$  gehören, werden dabei zu einem Cluster zusammengefasst, wenn sie nahe beieinander liegen. Ein derartiges Clustering auf Objektbasis kann beispielsweise

mit dem partitionierenden Clusteralgorithmus  $k$ -Means erzeugt werden. Jedes Cluster wird dann durch ein MUR approximiert (siehe Abbildung 7). Dadurch wird ein Objekt nicht länger durch  $s$  Samplepunkte, sondern durch  $k$  Clusterapproximationen beschrieben, die zusammen alle  $s$  Samples des Objekts enthalten. Die geclusterten Repräsentationen werden dann in einem R\*-Baum abgelegt. Diese Art der Speicherung von unscharfen Objekten ermöglicht es, den Distance-Range Join mittels eines parallelen R-Baum Durchlaufs ähnlich zum RSJ-Algorithmus [BKS93] effizient auszuführen. Im Gegensatz zum RSJ-Algorithmus wird auf der Blattebene für jedes Objekt-paar aus den approximierten Sample-Punkten ein Wahrscheinlichkeitswert berechnet und dem jeweiligen Objekt-paar zugeordnet.

**Filterschritt für Effiziente Anfragebearbeitung.** Der wahrscheinlichkeitsbasierte Join-Algorithmus gibt alle Objekt-paare als Resultat zurück, deren Join-Wahrscheinlichkeit positiv ist. Oftmals ist es jedoch wünschenswert, die Resultatmenge in einem für den Benutzer überschaubaren Umfang zu halten. Eine Möglichkeit dafür ist, die Resultate nach und nach auszugeben, absteigend geordnet nach dem Wahrscheinlichkeitswert. Das gibt dem Benutzer die Möglichkeit, die Anfrage frühzeitig zu beenden, sobald die derzeitige Antwortmenge ausreicht oder sobald ein gewisser Wahrscheinlichkeitswert unterschritten ist.

Die durch MURs gegebenen Sample-Approximationen erlauben es, die jeweilige Join-Wahrscheinlichkeit zweier Objekte mittels einer oberen Schranke effizient abzuschätzen, ohne auf die exakten Sample-Werte zugreifen zu müssen. Diese konservative Abschätzung ermöglicht in einem Filterschritt, diejenigen Objekt-paare aus der Ergebnismenge auszufiltern, deren abgeschätzte Join-Wahrscheinlichkeit den benutzerdefinierten Wahrscheinlichkeitsgrenzwert unterschreitet. Somit kann die exakte Berechnung der Join-Wahrscheinlichkeit für viele Objekt-paare vermieden werden, so dass die Anfrage wesentlich effizienter durchgeführt werden kann.

## 5 Literatur

- [ABKS99] Ankerst, M.; Breunig, M. M.; Kriegel, H.-P.; Sander, J.: OPTICS: Ordering Points To Identify the Clustering Structure. SIGMOD, 1999.
- [ABK+06] Achtert, E.; Böhm, C.; Kröger, P.; Kunath, P.; Renz, M.; Pryakhin, A.: Efficient reverse k-nearest neighbor search in arbitrary metric spaces. SIGMOD, 2006.
- [AHV95] Abiteboul, S.; Hull, R.; Vianu, V.: Foundations of Databases. Addison Wesley, 1995.
- [AKPS06] Achtert, E.; Kriegel, H.-P.; Pryakhin, P.; Schubert, M.: Clustering Multi-represented Objects Using Combination Trees. PAKDD, 2006.
- [BKK96] Berchold, S.; Keim, D. A.; Kriegel, H.-P.: The X-Tree: An index structure for high-dimensional data. VLDB, 1996.
- [BKPP06] Brecheisen, S.; Kriegel, H.-P.; Kunath, P.; Pryakhin, A.: Hierarchical Genre Classification for Large Music Collections. ICME, 2006.
- [BKP05] Brecheisen, S.; Kriegel, H.-P.; Pfeifle, M.: Efficient Similarity Search on Vector Sets. BTW, 2005.
- [BKS93] Brinkhoff, T.; Kriegel, H.-P.; Seeger, B.: Efficient Processing of Spatial Joins Using R-trees. SIGMOD, 1993.
- [BKSS90] Beckmann, N.; Kriegel, H.-P.; Schneider, R.; Seeger, B.: The R\*-Tree: An efficient and robust access method for points and rectangles. SIGMOD, 1990.
- [BKS+04] Bustos, B.; Keim, D. A.; Saupe, D.; Schreck, T.; Vranic, D. V.: Using entropy impurity for improved 3D object similarity search. ICME, 2004.
- [BPS06a] Böhm, C.; Pryakhin, A.; Schubert, M.: The Gauss-Tree: Efficient Object Identification of Probabilistic Feature Vectors. ICDE, 2006.
- [BPS06b] Böhm, C.; Pryakhin, A.; Schubert, M.: Probabilistic Ranking Queries on Gaussians. SSDBM, 2006.
- [BS04] Bickel, S.; Scheffer, T.: Multi-View Clustering. ICDM, 2004.
- [CBGM02] Carson, C.; Belongie, S.; Greenspan, H.; Malik, J.: Blobworld: Image segmentation using Expectation-Maximization and its application to image querying. Transactions on PAAMI, 24(8), 2002.
- [CKP03] Cheng, R.; Kalashnikov, D. V.; Prabhakar, S.: Evaluating probabilistic queries over imprecise data. SIGMOD, 2003.
- [CKP04] Cheng, R.; Kalashnikov, D. V.; Prabhakar, S.: Querying imprecise data in moving object environments. TKDE, 2004.
- [CPZ97] Ciaccia, P.; Patella, M.; Zezula, P.: M-Tree: An efficient access method for similarity search in metric spaces. VLDB, 1997.
- [CVK04] Costa, C. H. L.; Valle, J. D.; Koerich, A. L.: Automatic Classification of Audio Data. TSMC, 2004.
- [DeS05] de Sa, V. R.: Spectral clustering with two views. ICML Workshop on Learning With Multiple Views, 2005.
- [EM97] Eiter, T.; Mannila H.: Distance Measures for Point Sets and Their Computation. Acta Informatica, 34(2), 1997.

- [HJR97] Huang, Y.-W.; Jing, N.; Rundensteiner, E.A.: Spatial Joins Using R-trees: Breadth-First Traversal with Global Optimizations. VLDB, 1997.
- [KBK+03] Kriegel, H.-P.; Brecheisen, S.; Kröger, P.; Pfeifle, M.; Schubert, M.: Using Sets of Feature Vectors for Similarity Search on Voxalized CAD Objects. SIGMOD, 2003.
- [KKKP06] Kriegel, H.-P.; Kröger, P.; Kunath, P.; Pryakhin, A.: Effective Similarity Search in Multimedia Databases using Multiple Representations. MMM, 2006.
- [KKPR06] Kriegel, H.-P.; Kunath, P.; Pfeifle, M.; Renz, M.: Probabilistic Similarity Join on Uncertain Data. DASFAA, 2006.
- [KKPS04a] Kriegel, H.-P.; Kröger, P.; Pryakhin, P.; Schubert, M.: Using Support Vector Machines for Classifying Large Sets of Multi-Represented Objects. SDM, 2004.
- [KKPS04b] Kriegel, H.-P.; Kailing, K.; Pryakhin, P.; Schubert, M.: Clustering Multi-represented Objects with Noise. PAKDD, 2004.
- [KM00] Korn, F.; Muthukrishnan, S.: Influenced sets based on reverse nearest neighbor queries. SIGMOD, 2000.
- [LR96] Lo, M.-L.; Ravishanker, C. V.: Spatial Hash Joins. SIGMOD, 1996.
- [Mot95] Motro, A.: Management of Uncertainty in Database Systems. In Modern Database Systems, Won Kim (Ed.), Addison Wesley, 1995.
- [PD96] Patel, J. M.; DeWitt, D. J.: Partition Based Spatial-Merge Join. SIGMOD, 1996.
- [Sch91] Schroeder, M.: Fractals, Chaos, Power Laws: Minutes from an infinite paradise. W.H. Freeman and company, 1991.
- [SJL+03] Smith, J. R.; Jaimes, A.; Lin, C.-Y.; Naphade, M. R.; Natsev, A.; Tseng, B. L.: Interactive search fusion methods for video database retrieval. ICIP, 2003.
- [TPL04] Tao, Y.; Papadias, D.; Lian, X.: Reverse kNN search in arbitrary dimensionality. VLDB, 2004.
- [WSCY99] Wolfson, O.; Sistla, A. P.; Chamberlain, S.; Yehsa, Y.: Updating and Querying Databases that Track Mobile Units. Distributed and Parallel Databases, 7(3), 1999.
- [XMS+03] Xu, C.; Maddage, N. C.; Shao, X.; Cao, F.; Tian, Q.: Musical genre classification using support vector machines. ICASP, 2003.
- [YL01] Yang, C.; Lin, K.-I.: An index structure for efficient reverse nearest neighbor queries. ICDE, 2001.
- [Zha03] Zhang, T.: Semi-automatic approach for music classification. SPIE CIMMS, 2003.



Hans-Peter Kriegel ist seit 1991 Inhaber des Lehrstuhls für Datenbanksysteme an der Ludwig-Maximilians-Universität (LMU) München und seit 2003 Direktor des Departments „Institut für Informatik“. Er hat über 200 Veröffentlichungen auf begutachteten Tagungen sowie in Fachzeitschriften. Diplom (1973) und Promotion (1976) in Informatik erfolgten an der Universität Karlsruhe. Seine For-

schungsschwerpunkte liegen in den Bereichen Datenbanksysteme für komplexe Objekte (Molekularbiologie, Medizin, Multimedia, CAD usw.), insbesondere Anfragebearbeitung, Ähnlichkeitssuche, hochdimensionale Indexstrukturen sowie im Bereich Knowledge Discovery in Databases und Data Mining.



Christian Böhm ist Professor für Praktische Informatik an der Ludwig-Maximilians-Universität (LMU) München und erforscht Indexstrukturen für die Ähnlichkeitssuche sowie Data-Mining-Algorithmen. Er ist Autor von ca. 50 wissenschaftlichen Publikationen. Elke Achtert arbeitet als wissenschaftliche Mitarbeiterin in der Gruppe von Prof. Christian Böhm im Bereich Knowledge Discovery in Databases und Ähnlichkeitssuche in Multimedia Datenbanken.



Stefan Brecheisen arbeitet als wissenschaftlicher Mitarbeiter in der Gruppe von Prof. Hans-Peter Kriegel im Bereich Ähnlichkeitssuche von komplexen Objekten.



Peter Kunath arbeitet als wissenschaftlicher Mitarbeiter in der Gruppe von Prof. Hans-Peter Kriegel im Bereich Effiziente Verwaltung von räumlichen, mobilen und multimedialen Objekten in Datenbanken.



Matthias Renz arbeitet als wissenschaftlicher Mitarbeiter in der Gruppe von Prof. Hans-Peter Kriegel im Bereich Effiziente Verwaltung von räumlichen, zeitlichen und mobilen Objekten.



Alexey Pryakhin arbeitet als wissenschaftlicher Mitarbeiter in der Gruppe von Prof. Hans-Peter Kriegel im Bereich Mining von komplexen Daten und multimedialen Daten.



Matthias Schubert hat in der Gruppe von Prof. Hans-Peter Kriegel promoviert und arbeitet dort als wissenschaftlicher Assistent an Multimedia Datenbanken, Management unscharfer Daten, Data Mining, und Web Content Mining.

