Effiziente Indexstrukturen für hochdimensionale Datenräume[1]

Christian Böhm

Ludwig-Maximilians-Universität München
Oettingenstr. 67, 80538 München, Tel. 089/2178-2228
boehm@informatik.uni-muenchen.de

Die Indexierung hochdimensionaler Datenräume ist eine neue Forschungsrichtung, die durch die Notwendigkeit, moderne Anwendungen mit mächtigen Suchwerkzeugen auszustatten, zunehmend an Bedeutung gewinnt. Mit Nicht-Standard-Datenbanken werden Anwendungen wie Multimedia, CAD, medizinische Bildverarbeitung, Aktienanalyse usw. unterstützt. Dabei ist die Ähnlichkeitssuche in großen Datenmengen eine wichtige Basisfunktion. Ähnlichkeitsanfragen werden häufig auf Nachbarschaftsanfragen in hochdimensionalen Feature-Räumen abgebildet, die durch multidimensionale Indexstrukturen unterstützt werden. Da Indexstrukturen häufig bei hohen Dimensionen versagen, wurden im Rahmen der Dissertation mehrere Optimierungstechniken vorgeschlagen. Sie basieren auf einem Kostenmodell für die indexbasierte Anfragebearbeitung, das ebenfalls in dieser Arbeit entwickelt wurde. Die Optimierung der Parameter Seitengröße, Indexdimension, Seitengeometrie usw. führt zu deutlichen Steigerungen der Performanz.

Indexing high-dimensional data spaces is a new domain of research which gains increasing importance by the needs of modern applications such as multimedia, CAD, medical imaging, stock price analysis etc. In these applications, similarity search is an important basic functionality. Similarity queries are often transformed into neighborhood queries in a high-dimensional feature space which are supported by multidimensional index structures. As these indexes often deteriorate with increasing dimension, in this dissertation various optimization techniques were proposed, founded on a cost model for index based query processing which was also developed in this thesis. The optimization of parameters such as the page size, the index dimension, the page geometry etc. leads to clear improvements of the performance.

---

1. Titel der englischsprachigen Dissertation [Böh 98]:
   Efficiently Indexing High-Dimensional Data Spaces

1 Motivation

Information ist in der heutigen Zeit der Generalschlüssel für wirtschaftlichen Erfolg. “Nur wer über das neueste Wissen verfügt und es für seine Entwicklungen nutzen kann, hat Chancen im globalen Wettbewerb”, so der Vizepräsident der Arbeitsgemeinschaft industrieller Forschungsvereinigungen (AiF), O. H. Schiele. Wichtig für die Anwendbarkeit von Information ist ihre Qualität und die schnelle Verfügbarkeit. In zunehmendem Maße entwickelt sich allerdings die effektive und effiziente Recherchierbarkeit der relevanten Informationen zum zentralen Problem.

Solange die Struktur der zu verwaltenden Information einfach ist, wie etwa bei eindimensionalen numerischen Werten oder bei Textdaten, kann dieses Problem als gelöst betrachtet werden. In den letzten Jahren haben sich jedoch eine Reihe von sogenannten Nicht-Standard-Applikationen von Datensystemen entwickelt, bei denen große Mengen komplex strukturierter Objekte gespeichert werden. In Bereichen wie CAD-Datenbanken, Multimedia, medizinischer Bildverarbeitung, molekularbiologischen Datenbanken oder der Zeitreihenanalyse recherchieren Benutzer häufig auf der Basis von Ähnlichkeit. Aufgrund vorgegebener Suchobjekte sollen Objekte mit ähnlichen Eigenschaften aus der Datenbank selektiert werden. Der Ähnlichkeitsbegriff ist für die einzelnen Applikationsbereiche spezifisch.

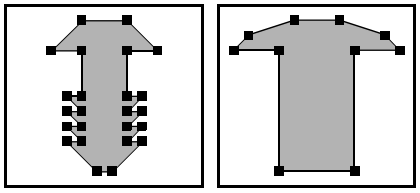

Abb. 1: CAD-Datenbanken.

Ein anschauliches Beispiel bilden die CAD-Datenbanken [BK 97]. In einem kürzlich abgeschlossenen Forschungsprojekt wurde das S^3-System (*similarity search system*) entwickelt. Ziel dieses Projektes war die Kostensenkung bei der Fertigung und Montage von Spritzgußteilen bei der Automobilproduktion durch eine Reduzierung der Teilevielfalt. Das S^3-System ermöglicht deshalb die geometriebasierte Recherche von ähnlichen Bauteilen.

Eine weit verbreitete Technik zur Ähnlichkeitssuche ist die sogenannte Feature-Transformation, bei der wichtige Eigenschaften eines Datenobjekts in einen Punkt eines mehrdimensionalen Vektorraums (Feature-Vektor) überführt werden. Features, die etwa in CAD-Datenbanken verwendet werden, sind Kurvatur-Koeffizienten sowie Volumens-Proportionen. Eine wichtige Eigenschaft der Feature-Transformation ist, daß der Ähnlichkeit der beteiligten Objekte eine geringe euklidische Distanz der zugeordneten Feature-Vektoren entspricht. So werden Ähnlichkeitsanfragen auf natürliche Weise in Nachbarschaftsanfragen im Feature-Raum übersetzt.

Um eine hohe Effizienz bei der Anfragebearbeitung zu erreichen, werden häufig multidimensionale Indexstrukturen zur Verwaltung der Feature-Vektoren eingesetzt. Leider versagen herkömmliche multidimensionale Indexstrukturen oft bei

einer hohen Dimension des Datenraums, da sie in erster Linie für niedrigdimensionale Räume konzipiert wurden. Verantwortlich für das Versagen der Indexstrukturen sind eine Reihe von Effekten, die üblicherweise mit dem Schlagwort '*Fluch der Dimension*' ('*Curse of Dimensionality*') belegt werden.

Das Hauptziel der vorliegenden Arbeit ist deshalb die Verbesserung der Performanz bei der indexbasierten Anfragebearbeitung in hochdimensionalen Datenräumen. Hierzu wurde ein Kostenmodell für die indexbasierte Anfragebearbeitung in hochdimensionalen Datenräumen entwickelt, das auf eine Reihe von Indexstrukturen und Techniken zur Anfragebearbeitung anwendbar ist und sowohl zur Evaluation dieser Techniken als auch zu deren Optimierung eingesetzt werden kann.

2 Die Anfragebearbeitung im Hochdimensionalen

Indexstrukturen für hochdimensionale Datenräume haben ihre Wurzeln in den sog. *multidimensionalen Indexstrukturen*, deren Entwicklung in den frühen 80er Jahren mit dem R-tree und dem Grid-File begann. Entwickelt vor dem Hintergrund geographischer Informationssysteme, optimieren diese Indexstrukturen die Suche in niedrigdimensionalen Datenräumen. Da diese Strukturen in mittel- und hochdimensionalen Datenräumen, wie sie bei der Feature-Transformation typischerweise entstehen, versagen, wurden später auch einige spezialisierte Indexstrukturen für diesen Zweck vorgestellt, etwa der TV-tree [LJF 94] und der X-tree [BKK 96].

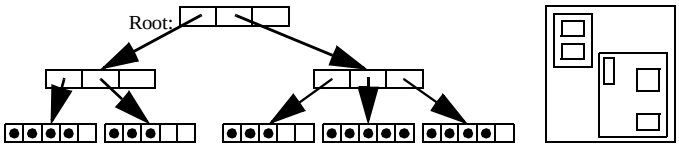

Abb. 2: Multidimensionale Indexstrukturen.

Allen diesen Strukturen ist gemeinsam, daß es sich um ausgeglichene Bäume handelt, deren Knoten durch Seiten des Hintergrundspeichers repräsentiert werden. Jedem Knoten ist außerdem eine Region des Datenraums zugeordnet, die im jeweils übergeordneten Knoten gespeichert ist. Die einem Knoten zugeordnete Region ist in der Region des Eltern-Knotens immer vollständig enthalten.

Unterschiede zwischen den Indexstrukturen ergeben sich aus der Gestalt der Seitenregionen, die beim TV-tree (hyper-) zylindrisch und beim X-tree rechteckig ist. Außerdem unterscheiden sich die Heuristiken, die beim Einfügen neuer Punkte in den Index und beim Überlauf von Knoten angewandt werden. In Ähnlichkeitssuchsystemen sind zwei Anfragetypen vorherrschend. Die Bereichsanfrage (*range query*) gibt einen Anfragepunkt q und einen Anfrageradius r vor und selektiert alle Objekte der Datenbank, deren Abstand von q kleiner oder gleich r ist. Im Gegensatz dazu selektiert die k-nächste-Nachbar-Anfrage (*k-nN, k-nearest neighbor query*) diejenigen k Objekte mit dem geringsten Abstand von q aus der Datenbank.

Bereichsanfragen werden meist durch eine einfache Tiefensuche im Baum bearbeitet, die genau alle von der Anfragekugel geschnittenen Seiten zugreift. Eine Tiefensuche ist auch zur Bearbeitung der *k*-nN-Anfrage möglich. An die Stelle des festen Radius *r* als Entscheidungskriterium für den Seitenzugriff tritt hier der Abstand zu den bisher gefundenen *k* nächsten Punkten (den *Kandidaten*). Die Tiefensuche ist jedoch keine optimale Lösung für das *k*-nN-Problem, da sich Beispiele finden lassen in denen mehr Seiten als erforderlich zugegriffen werden. Der sog. HS-Algorithmus, der diese Schwäche vermeidet [HS 95], sortiert die zu ladenden Seiten in einer Prioritätswarteschlange und lädt jeweils nur die erste Seite aus dieser Liste in den Arbeitsspeicher. Bei diesem Vorgehen läßt sich beweisen, daß genau diejenigen Seiten zugegriffen werden, deren Abstand nicht größer als der Abstand des *k*-nN ist. Hierdurch ist die Optimalität gewährleistet.

3 Ein Kostenmodell für die Anfragebearbeitung

Ziel des vorliegenden Kostenmodells [BBKK 97] ist die zuverlässige Vorhersage der Kosten der indexbasierten Anfragebearbeitung in hochdimensionalen Datenräumen. Die Anzahl der Seitenzugriffe soll sowohl für Bereichsanfragen als auch *k*-nN-Anfragen geschätzt werden. Aufgrund der Optimalität des HS-Algorithmus ist es möglich, *k*-nN-Anfragen auf äquivalente Bereichsanfragen zurückzuführen (vgl. Abschnitt 3.1). Im darauffolgenden Abschnitt 3.2 werden die Kosten für die Bearbeitung einer Bereichsanfrage unter den Annahmen einer unabhängigen Gleichverteilung bei der Datenmenge und der Anfrage geschätzt. In Abschnitt 3.3 werden einige Sondereffekte behandelt, die in hochdimensionalen Datenräumen auftreten und in Abschnitt 3.4 wird auf Korrelationseffekte eingegangen, die bei Realdaten häufig auftreten.

3.1 Reduktion von *k*-nN-Anfragen auf Bereichsanfragen

Aufgrund der Optimalität des HS-Algorithmus entspricht eine *k*-nN-Anfrage genau einer Bereichsanfrage, bei der die Distanz zum *k*-nN als Bereichsradius angegeben wurde. Deshalb entspricht die Reduktion der *k*-nN-Anfrage gerade einer Schätzung der entsprechenden Distanz. Frühere Ansätze gehen davon aus, daß sich der Erwartungswert dieses Radius aus der Hyperkugel mit dem Volumen $1/N$ ermitteln läßt, wobei N die Anzahl der in der Datenbank gespeicherten Punkte ist. Leider ist diese Rückrechnung nicht korrekt und liefert selbst unter idealisierten Randbedingungen schlechte Schätzwerte. Eine stochastisch korrekte Herangehensweise besteht darin, in einer Verteilungsfunktion jedem Radius r die Wahrscheinlichkeit zuzuordnen, mit der die *k*-nN-Distanz mindestens r ist. Für den Fall von $k = 1$ entspricht dies der

Wahrscheinlichkeit, daß kein Punkt in der Kugel mit Radius r enthalten ist, und deshalb:

$$P(r) = 1 - (1 - V(r))^N,$$

wobei $V(r)$ das Volumen der d-dimensionalen Hyperkugel mit Radius r ist:

$$V(r) = \frac{\pi^{d/2} \cdot r^d}{\Gamma(d/2 + 1)}$$

Aus $P(r)$ läßt sich durch Differentiation und daran anschließender Integration ein Erwartungswert bestimmen:

$$R = \int_0^\infty \left(r \cdot \frac{\partial P(r)}{\partial r} \right) dr$$

Auf ähnliche Weise läßt sich auch für Anfragen mit $k \neq 1$ sowie für Anfragen mit Maximumsmetrik vorgehen. Das Integral ist numerisch effizient auswertbar.

3.2 Kostenschätzung für Bereichsanfragen

Die individuelle Zugriffswahrscheinlichkeit einer gegebenen Seite mit einer rechteckigen Seitenregion unter einer Bereichsanfrage mit Radius r und beliebigem Mittelpunkt ergibt sich aus der sogenannten Minkowski-Summe der Seiten- und Anfrageregion. Bei diesem Konzept stelle man sich vor, daß an jeder Stelle des Datenraums, an denen sich Anfrage und Seitenregion schneiden, der Datenraum am Mittelpunkt der Anfrage markiert wird (vgl. Abb. 3). Da alle Positionen des Datenraums markiert wurden, an denen die Seite zugegriffen wird, entspricht die Zugriffswahrscheinlichkeit dem Verhältnis aus dem markierten Volumen und dem Volumen des Datenraums. Durch die Minkowski-Summe wird die Seitenregion an allen Ecken um den Bruchteil einer Hyperkugel vergrößert. Alle Bruchteile zusammen ergeben eine ganze Kugel. An allen Kanten wird ein Teil eines Hyperzylinders angehängt, der aus einer (d - 1) - dimensionalen Kugel mit Radius r und einer geraden Komponente der Länge der Kante besteht. Auch an den Oberflächensegmenten höherer Dimension befinden sich Hyperzylinder unterschiedlicher Dimensionalität. Zusammen ergibt sich für eine würfelförmige Seite mit Kantenlänge a die folgende Binomialformel für die Minkowski-Summe:

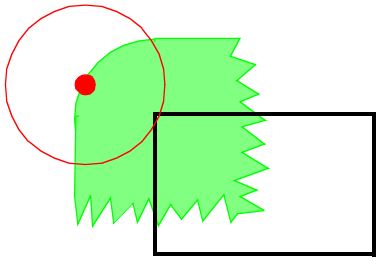

Abb. 3: Minkowski-Summe.

$$V_{\text{Mink}} = \sum_{0 \le i < d} \binom{d}{i} \cdot a^{d-i} \cdot V_i(r)$$

Für eine Schätzung der Anzahl der Seitenzugriffe ist weiterhin erforderlich, die Kantenlänge der Seitenregion zu schätzen.

3.3 Effekte in hochdimensionalen Datenräumen

In hochdimensionalen Datenräumen ist wegen $\lim_{d \to \infty} \sqrt[d]{1/N} \to 1$ für bel. $N > 0$ der Radius einer typischen Anfrage in derselben Größenordnung wie der Durchmesser des Datenraums. Aus diesem Grund liegen große Teile der Anfragekugel bzw. der Minkowski-Summe häufig außerhalb des Datenraums. Da sich dort keine Daten- oder Anfragepunkte befinden können, darf dieses herausragende Volumen bei der Wahrscheinlichkeitsbestimmung nicht mit berücksichtigt werden. Diesem Problem wird dadurch Rechnung getragen, daß anstelle des Kugelvolumens $V(r)$ ein korrigiertes Kugelvolumen $V_{\text{korrig}}(r)$ eingesetzt wird, bei dem der durchschnittliche Anteil außerhalb des Datenraums bereits subtrahiert ist. Das korrigierte Volumen läßt sich zur Compile-Zeit der Kostenschätzer-Programme mit einer relativ teuren Montecarlo-Methode numerisch bestimmen, und zur effizienten Anwendung tabellieren.

3.4 Effekte bei realen Datenmengen

Die obigen Kostenschätzungen beruhen auf der Einschränkung auf eine unabhängige Gleichverteilung der Daten- und Anfragepunkte. Hierbei erweisen sich glatte, unabhängige nicht-uniforme Datenverteilungen als unproblematisch, da die Minkowski-Summe umgekehrt proportional zur Dichte der Anfragepunkte wächst und sich diese beiden Effekte weitgehend aufheben. Problematischer sind Korrelationseffekte, die nichtlinear oder partiell sein können und die zu beträchtlichen Verzerrungen bei der Kostenschätzung führen. Der Grund für diese Verzerrungen ist die (falsche) Annahme, daß die Anzahl der Datenpunkte proportional zum einschließenden Volumen ist. Sind die Punkte korreliert, so befinden sie sich in einem niedrigerdimensionalen Teilraum (z.B. einer Gerade oder Ebene) und sind deshalb nur proportional zur niedrigerdimensionalen Projektion des Volumens. Die eigentliche Dimension des Datenraumes kann mit Hilfe der fraktalen Geometrie gemessen werden. Die fraktale Geometrie liefert auch einige Proportionalitätsgesetze die anstelle der Volumens-Proportionalität eingesetzt werden können.

Eine umfangreiche experimentelle Auswertung ergab eine hohe Genauigkeit der vorhergesagten durchschnittlichen Bearbeitungszeit unter zahlreichen natürlichen und künstlichen Datenmengen.

4 Optimierungstechniken für Indexstrukturen

Die Kostenschätzungen des im vorangegangenen Abschnitt vorgestellten Kostenmodells können dazu genutzt werden, um Parameter der Indexstrukturen zu optimieren. Im Rahmen der hier vorgestellten Arbeit wurden hierzu die logische Seitengröße der Indexstruktur (vgl. Abschnitt 4.1), die Indexdimension (vgl. Abschnitt 4.2) und ein Parameter der Seitengeometrie (vgl. Abschnitt 4.3), sowie die Datenverteilung bei einer verteilten Datenbank (vgl. Abschnitt 4.4) herangezogen.

4.1 Dynamische Seitengrößenoptimierung

Einer der wichtigsten Parameter bei Indexstrukturen allgemein und bei der hochdimensionalen Anfragebearbeitung im besonderen ist die logische Seitengröße des Indexes [BK 99a]. Die logische Datenseite ist die Einheit mit der Daten bei der Anfragebearbeitung und bei Update-Operation zwischen der Festplatte und Arbeitsspeicher transferiert werden. Üblicherweise ist die Seitengröße innerhalb eines gegebenen Indexes konstant um die Handhabung zu vereinfachen. Die logische Seitengröße ist immer ein Vielfaches der physikalischen Seitengröße, die durch das Plattenlaufwerk festgelegt ist.

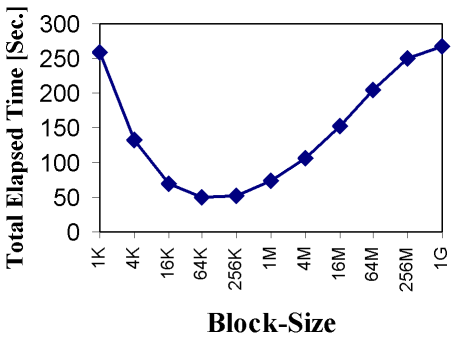

Abb. 4: Seitengrößenoptimierung.

Durch den technischen Aufbau des Plattenlaufwerkes ergeben sich zwei verschiedene Engpässe. Die Zugriffsrate (max. Anzahl von wahlfreien Zugriffen) wird in erster Linie durch die Positionierungszeit der Schreib-/Leseköpfe bestimmt, sowie durch die Umdrehungsgeschwindigkeit der Magnetplatten. Die Übertragungsrate (maximale Anzahl kontinuierlich lesbarer Bytes) ergibt sich aus der Umdrehungsgeschwindigkeit und der Kapazität des Disk-Controllers. Wird die logische Seitengröße zu klein gewählt, dann treten bei der Anfragebearbeitung viele wahlfreie Leseoperationen auf und die Zugriffsrate formt einen deutlichen Engpaß. Wird die Seitengröße zu hoch gewählt, dann werden viele unnötige Datensätze eingeladen und verarbeitet. In diesem Fall bildet die Übertragungsrate sowie u.U. die CPU-Leistung einen Engpaß. Die Lage des Optimums ist hierbei nicht nur von verschiedenen Hardware-Konstanten, sondern auch von der Dimension des Datenraums, der Datenverteilung (Grad der Korrelation) und der Anzahl der gespeicherten Datenpunkte abhängig. Wegen dieser Abhängigkeit ist auch nicht auszuschließen, daß sich die Lage des Optimums nach zahlreichen Einfüge-, Lösch- oder Änderungsoperationen verschiebt, und die Seitengröße des Indexes deshalb anzupassen ist. Um längere Stillstandzeiten zu vermeiden, wurde die Indexstruktur

so modifiziert, daß Anpassungen der Blockgröße an das Optimum sukzessive während des laufenden Betriebs (d.h. dynamisch) vorgenommen werden. Deshalb wurde die Forderung nach einer konstanten Seitengröße für den gesamten Index verworfen. Stattdessen wird für jede Seite eine individuelle Seitengröße zugelassen, die im Directory vermerkt ist. Abgesehen von Verschnitten, die sich aufgrund der physikalischen Seitengröße ergeben, sind die Seiten dieses Systems zu 100% gefüllt, was zu einer weiteren Leistungsverbesserung führt.

Die Seitengrößenoptimierung wird lokal an jeder einzelnen Seite vorgenommen. Zu diesem Zweck wird nach einer festgelegten Anzahl von Änderungsoperationen an einer Seite gemäß der in Abschnitt 3 eingeführten Kostenschätzung die Zugriffswahrscheinlichkeit der betrachteten Seite ermittelt. Die Seite wird probeweise geteilt bzw. mit einer geeigneten Partnerseite verschmolzen, und dann die Kostenschätzung nochmals durchgeführt. Ist die Kostenbilanz nach der Split- (Merge-) Operation günstiger, so wird sie beibehalten, ansonsten zurückgesetzt.

Die Problematik, Speicherlücken zu finden und zu beseitigen, die durch die Aufgabe einer konstanten Seitengröße entstehen, wird ebenfalls ausführlich diskutiert [BK 99a]. Die vorgeschlagene Lösung beruht auf der Technik der Garbage Collection. Eine Umfangreiche Evaluierung der Dynamischen Seitengrößenoptimierungstechnik ergibt ein Verbesserungspotential gegenüber Standard-Seitengrößen (4 KByte) von Faktoren zwischen 360% bis 460%. Auch die sequentielle Suche als indexfreie Zugriffsmethode wird um bis zu 660% durch den dynamisch optimierten Baum übertroffen.

4.2 Optimierung der Indexdimension

Diese Optimierungstechnik [BBK+ 99] hat ihre Motivation in einer Alternativtechnik zu multidimensionalen Indexstrukturen, den sog. invertierten Listen. Hierbei wird jedes Attribut in einem separaten eindimensionalen Index (B-Baum) verwaltet. Zur Bearbeitung von Bereichsanfragen wird die Anfrageregion ebenfalls durch Projektion zerlegt und an die einzelnen Indexe weitergeleitet. Die Ergebnislisten der einzelnen Anfragen werden verglichen. Das Gesamtergebnis besteht aus der Schnittmenge der Einzelergebnisse. Zur Schnittmengenbildung müssen die Ergebnismengen nach einem eindeutigen Objekt-Identifikator sortiert werden. Die eindimensionalen Indexe haben gegenüber den multidimensionalen Indexen den Vorteil, daß die Degenerationserscheinungen des “Curse of Dimensionality” nicht auftreten. Wegen den in Abschnitt 3.3 diskutierten Effekten in hochdimensionalen Datenräumen können die eindimensionalen Indexe jedoch nicht sehr selektiv sein, so daß die einzelnen Ergebnismengen sehr groß werden.

Eine Kompromißlösung zwischen den invertierten Listen und dem hochdimensionalen Index, der diese beiden Nachteile vermeidet, besteht darin, den hochdimensionalen Datenraum in mehrere Teilräume mit moderater Dimension zu zerlegen. Es stellt sich ähnlich zu Abschnitt 4.1 die Optimierungsaufgabe, eine geeignete Anzahl und Dimension der Teilindexe zu finden. Im Gegensatz zur Seitengrößenoptimierung kann die Dimensionszuweisung allerdings nur vor dem Indexaufbau erfolgen und ist dann statisch. Es wurden Verbesserungen von bis zu 230% gegenüber den multidimensionalen Indexen (R*-Baum) und bis zu 2000% gegenüber den invertierten Listen gemessen.

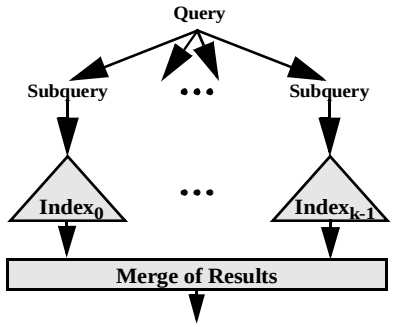

Abb. 5: Tree Striping.

4.3 Optimierung der Seitengeometrie und Pyramidentechnik

Hier wurde zunächst eine Optimierungstechnik für die Gestalt der Seitenregionen [BBK 98a] vorgestellt, die sich auf sog. *a-priori*-Wissen der gesamten Datenmenge bei der Index-Neukonstruktion [BK 99b] (der *Bottom-Up-Konstruktion*) stützt. In diesem Zusammenhang wurde auch eine neue Konstruktionsmethode für multidimensionale Indexe vorgestellt. Die Optimierung beruht auf der Beobachtung, daß im Hochdimensionalen bei niedriger Korrelation schmale Datenseiten Vorteile gegenüber annähernd quadratischen Seiten, die gewöhnlich bei der Konstruktion erzeugt werden, haben. Die Technik wurde mit der Pyramidentechnik [BBK 98b], die dieses Prinzip noch stärker ausnutzt, perfektioniert. Ein sehr deutlicher Verbesserungsfaktor bis zu 58800% wurde für die Geschwindigkeit der Indexkonstruktion erzielt. Auch die Anfragebearbeitung verbessert sich durch den Einsatz dieser Methode um bis zu 1680%.

4.4 Optimierte Datenverteilung

Diese Technik [BBB+ 97] dient zur hochdimensionalen Anfragebearbeitung in einer verteilten Datenbank. Hier ist die Hauptaufgabe die Verteilung der Daten über die verschiedenen Knoten eines Rechnernetzes zur Optimierung der Anfragebearbeitung (das sog. *Declustering*). In Experimenten wurde ein nahezu linearer Speed-Up und ein konstanter Scale-Up erzielt. Konkurrierende Verfahren (z.B. Hilbert-Declustering) wurden um bis zu 500% übertroffen.

5 Zusammenfassung

Die vorgestellten Techniken, insbesondere die zur Kostenmodellierung, haben sich als wichtige Beiträge für das neue Forschungsgebiet der hochdimensionalen Index-

strukturen erwiesen. Die Anwendbarkeit dieser Techniken geht zudem weit über die in der Dissertation vorgestellten Gebiete hinaus. Die Ergebnisse wurden auch von namhaften internationalen Forschergruppen weiterentwickelt und angewandt. Weber, Schek und Blott [WSB 98] haben in einer Performanz-Studie verschiedene Zugriffsmethoden miteinander verglichen und das Kostenmodell dazu verwendet, die Grenzen des indexbasierten Ansatzes aufzuzeigen. Ciaccia, Patella und Zezula [CPZ 98] haben die Ergebnisse auf das Gebiet der Indexstrukturen für metrische Räume übertragen. Papadopoulos und Manolopoulos [PM 98] haben die Ergebnisse für eine neue Verteilungstechnik in Disk Arrays benutzt. Außerdem haben Riedel, Gibson und Faloutsos [RGF 98], sowie Agrawal, Gehrke et al. [Agr+ 98] das Kostenmodell im Bereich des Data Mining eingesetzt.

Zahlreiche Anwendungsmöglichkeiten werden im Rahmen der zukünftigen Arbeit erforscht werden. Insbesondere stellt sich die Frage nach der Kombinierbarkeit der verschiedenen Optimierungstechniken und nach der Integration in kommerzielle Datenbanksysteme [BBKM 99]. Neue Anwendungen für Indexstrukturen wie Biometrie oder Data Warehousing stellen zusätzliche attraktive Herausforderungen dar.

Literatur

[Agr+ 98] Agrawal, Gehrke, Gunopulos, Raghavan: *Automatic Subspace Clustering of High Dimensional Data for Data Mining Applications.* ACM SIGMOD Int. Conf. 1998

[BBB+ 97] Berchtold, Böhm, Braunmüller, Keim, Kriegel: *Fast Parallel Similarity Search in Multimedia Databases*. ACM SIGMOD Int. Conf. 1997

[BBK 98a] Berchtold, Böhm, Kriegel: *Improving the Query Performance of High-Dimensional Index Structures by Bulk-Load Operations*. Int. Conf. on Extending Database Technology, EDBT 1998

[BBK 98b] Berchtold, Böhm, Kriegel: *The Pyramid-Technique: Towards Breaking the Curse of Dimensionality.* ACM SIGMOD Int. Conf. 1998

[BBKK 97] Berchtold, Böhm, Keim, Kriegel: *A Cost Model For Nearest Neighbor Search in High-Dimensional Data Space.* ACM Int. Symp. on Principles of Database Systems, PODS, 1997.

[BBKM 99] Berchtold, Böhm, Kriegel, Michel: *Implementation of Multidimensional Index Structures for Knowledge Discovery in Relational Databases*, Int. Conf. on Data Warehousing and Knowledge Discovery, DaWaK, 1999.

[BBK+ 99] Berchtold, Böhm, Keim, Kriegel, Xu: *Optimal Multidimensional Query Processing Using Tree Striping*, subm. for pub. 1999.

[BK 97] Berchtold , Kriegel: *S3: Similarity Search in CAD Database Systems*. ACM SIGMOD Int. Conf. 1997.

[BK 99a] Böhm, Kriegel: D*ynamically Optimizing High-Dimensional Index Structures.* Subm. for pub. 1999.

[BK 99b] Böhm, Kriegel: Efficient Bulk Loading of Large *High-Dimensional Indexes.* Int. Conf. Data Warehousing and Knowledge Discovery, DaWaK, 1999.

[BKK 96] Berchtold, Keim, Kriegel: *The X-tree: An Index Structure for High-Dimensional Data*. Int. Conf. on Very Large Data Bases, VLDB 1996.

[Böh 98] Böhm: *Efficiently Indexing High-Dimensional Data Space*s, Diss. Ludwig Maximilians Universität München, Utz-Verlag München, 1998.

[CPZ 98] Ciaccia, Patella, Zezula: *A Cost Model for Similarity Queries in Metric Spaces.* ACM Int. Sympos. on Principles of Database Systems, PODS 1998.

[HS 95] Hjaltason, Samet: *Ranking in Spatial Databases*. Int. Symp. on Large Spatial Databases, SSD 1995.

[LJF 94] Lin, Jagadish, Faloutsos: *The TV-Tree: An Index Structure for High-Dimensional Data.* VLDB Journal 3(4), 1994.

[PM 98] Papadopoulos, Manolopoulos: *Similarity Query Processing Using Disk Arrays*. ACM SIGMOD Int. Conf. 1998

[RGF 98] Riedel, Gibson, Faloutsos: *Active Storage for Large-Scale Data Mining and Multimedia.* Int. Conf. on Very Large Data Bases, VLDB 1998.

[WSB 98] Weber, Schek, Blott: *A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces*. Int. Conf. on Very Large Data Bases, VLDB 1998.

Lebenslauf

Christian Böhm wurde 1968 in Rosenheim geboren, wo er 1988 am Ignaz-Günther-Gymnasium das Abitur erlangte. Nach seinem Studium der Informatik an der Technischen Universität München, das er 1994 mit Auszeichnung beendete, arbeitete er am Bayerischen Forschungszentrum für wissensbasierte Systeme (FORWISS), wo er für die Einrichtung eines bundesweiten Online-Informationsdienstes für die kunst- und literaturwissenschaftliche Forschung verantwortlich zeichnete. Seit 1996 ist er an der Ludwig-Maximilians-Universität als wissenschaftlicher Angestellter bzw. Assistent unter Prof. Dr. Hans-Peter Kriegel tätig. Nebenberuflich arbeitet er als freier Mitarbeiter des Consulting-Unternehmens STB GmbH, Augsburg (http://www.stb-gmbh.de), insbesondere an Datenanalyseverfahren und datenbankbasierten Intranetanwendungen. Gemeinsam mit vier Mitautoren wurde er 1997 mit dem *ACM SIGMOD best paper award* ausgezeichnet [BBB+ 97].