Eine Speicher- und Zugriffsarchitektur zur effizienten Anfragebearbeitung in Geo-Datenbanksystemen

Thomas Brinkhoff, Holger Horn, Hans-Peter Kriegel, Ralf Schneider

Institut für Informatik, Universität München, Leopoldstr. 11 B, 8000 München 40

Zusammenfassung

Im Bereich geographischer Datenbanksysteme liegen aufgrund der Komplexität der Objekte und Anfragen sowie der extrem großen Datenvolumina besondere Anforderungen an die Speicher- und Zugriffsarchitektur in bezug auf eine effiziente Anfragebearbeitung vor. In den letzten Jahren wurden eine Reihe von Konzepten, wie räumliche Indexstrukturen, Approximationen, Objektzerlegung und Mehrphasen-Anfragebearbeitung als einzelne Bausteine vorgeschlagen und analysiert, um die Leistungsfähigkeit von Geo-Datenbanksystemen zu steigern. In dieser Arbeit beschreiben wir eine globale Speicher- und Zugriffsarchitektur, die sich modular aus obigen Bestandteilen zusammensetzt. Um einen mengenorientierten Zugriff großräumiger Bereichsanfragen effizient zu unterstützen, haben wir unsere Geo-Architektur um das neue Konzept einer Szenen-Organisation erweitert. Ein experimenteller Leistungsvergleich zeigt, daß sich durch diese Szenen-Organisation massive Leistungssteigerungen, insbesondere für große Bereichsanfragen erzielen lassen.

1 Einführung und Motivation

Die rechnergestützte Verwaltung, Darstellung und Auswertung raumbezogener Information hat in den letzten Jahren immer größer werdende Bedeutung erlangt. So kommen geographische Informationssysteme in steigendem Maße in öffentlicher Verwaltung, Wissenschaft und Wirtschaft zum Einsatz. In vielen Anwendungsfeldern wie Raumplanung, Katasterwesen, Umweltüberwachung sind sie die Basis von entscheidungsunterstützenden Systemen [Bar 89].

Kern geographischer Informationssysteme (GIS) sind *geographische Datenbanken*. Im Gegensatz zu betriebswirtschaftlich-administrativen Anwendungen, die auf Standard-Datenbanken basieren, sind solche Datenbanksysteme für geographische Anwendungen ungeeignet [Wid 91]. Die unzureichende Ausdrucksstärke beispielsweise relationaler Systeme führt zu unnatürlicher Datenmodellierung und unzureichender Effizienz.

In den letzten Jahren haben daher verschiedene Forschungsgruppen eine Vielzahl von Konzepten und Techniken entwickelt, um einzelnen Anforderungen geographischer Datenbanksysteme gerecht zu werden. Beispiele dafür sind räumliche Datenmodelle, effiziente Geo-Indexstrukturen usw.

In den folgenden Abschnitten schlagen wir eine Speicher- und Zugriffsarchitektur für Geo-Datenbanksysteme *(Geo-Architektur)* vor, die aufzeigt, wie eine Reihe verschiedener Konzepte in eine Gesamtarchitektur integriert werden können. Dabei soll jedoch weder ein vollständig neues Datenbanksystem noch ein neuentwickelter Datenbankkern präsentiert werden. Vorschläge für solche Systeme sind DASDBS [SW 86], EXODUS [CDRS 86], GRAL [Gue 89] oder POSTGRES [SR 86]. Statt dessen kombinieren wir verschiedene Konzepte und Techniken zu einem effizienten Mechanismus für die räumliche Anfragebearbeitung, der in o. g. Systeme integriert werden kann. Dabei kommen neben bekannten Techniken auch neue Konzepte zum Einsatz, beispielsweise die *Szenen-Organisation*, die bei großräumigen Bereichsanfragen zu massiven Leistungssteigerungen führt.

In Abschnitt 2 stellen wir zunächst die Objekte und Operationen vor, die in geographischen Anwendungen auftreten. Daraus leiten wir wichtige Basisanfragen ab, die durch unsere Architektur effizient bearbeitet werden sollen. Im dritten Abschnitt teilen wir die Bearbeitung der Anfragen in Phasen ein. Algorithmen und Verfahren zur effizienten Unterstützung der einzelnen Phasen werden im vierten Abschnitt vorgestellt. Diese werden Bestandteil unserer Speicher- und Zugriffsarchitektur. Abschnitt 5 evaluiert das Leistungsverhalten der Architektur und geht dabei insbesondere auf das hier neu vorgestellte Konzept der Szenen-Organisation ein. Den Abschluß bilden Zusammenfassung und Ausblick.

2 Objekte und Operationen in einem Geo-Datenbanksystem

In dieser Arbeit schlagen wir eine konzeptionelle Architektur zur Objektspeicherung und Anfragebearbeitung in geographischen Datenbanksystemen vor. Für die Entwicklung einer solchen Architektur sind der Typ der zu verwaltenden Objekte sowie die Art der gewünschten Operationen von entscheidender Bedeutung. Dies gilt um so mehr, je höher die Anforderungen in bezug auf Laufzeit- und Speicherplatzeffizienz sind. In den folgenden Abschnitten geben wir daher zunächst eine Spezifikation der betrachteten Objekte und Operationen.

2.1 Objekte

Die in einer geographischen Datenbank gespeicherten Objekte *(Geo-Objekte)* modellieren einen bestimmten Ausschnitt der Erdoberfläche in bezug auf eine oder mehrere Eigenschaften (Themen). Dementsprechend sind die Objekte durch eine *geometrische Komponente* und eine *thematische Komponente* charakterisiert. Während die geometrische Komponente Ort und Ausdehnung des Realitätsbereiches beschreibt, der durch das Objekt repräsentiert wird, enthält die thematische Komponente eine qualitative oder quantitative thematische Charakterisierung.

Geometrische Komponente

Die geometrische Komponente der Objekte besteht jeweils aus einem der topologischen Grundelemente der Ebene, dem *Punkt*, der *Linie* oder der *Fläche*.

Punkte lassen sich einfach unter Angabe ihrer Koordinaten bezogen auf ein gegebenes Koordinatensystem beschreiben. Um linienartige Gebilde zu modellieren, finden sowohl Polygonzüge als auch freiformbare Kurven (Bézier/B-Spline) Verwendung. Der Darstellung flächiger Objekte kommt in Geo-Datenbanksystemen sehr große Bedeutung zu; auf sie wollen wir uns im folgenden konzentrieren. Zur Modellierung von Flächen existieren zwei grundsätzlich verschiedene Ansätze: das *Rastermodell* und das *Vektormodell*, für die in der Literatur jeweils eine Fülle spezifischer Vor- und Nachteile aufgeführt werden. Das Vektormodell weist eine sehr gute Skalierbarkeit auf und hat i. allg. einen geringeren Speicherplatzbedarf als das Rastermodell. Weiterhin bleiben die Objekte beim Vektoransatz individuell zugreifbar und sind damit einfacher indizierbar. Aus diesen Gründen wird in den letzten Jahren das Vektormodell innerhalb geographischer Datenbanksysteme favorisiert.

Für unsere Architektur sind *einfache Polygone mit Löchern (EPL)* die Objektklasse zur Flächenmodellierung. Ein EPL besteht aus einem einfachen, d.h. nicht selbstüberlappenden, Polygon, aus dem einfache Polygone als Löcher herausgeschnitten sein können (vgl. Bild 1). Die Klasse der EPL ist für geographische Anwendungen gut geeignet. Sie erlaubt eine Flächenbeschreibung in beliebiger Genauigkeit und berücksichtigt explizit die in geographischen Gegebenheiten häufig auftretenden “Lochbereiche” wie Seen oder Enklaven.

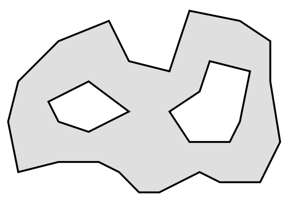

Bild 1: Einfaches Polygon mit Löchern

Thematische Komponente

Die thematische Komponente eines Objektes charakterisiert den durch seine Geo-Komponente modellierten Realitätsausschnitt in bezug auf ein oder mehrere Themen qualitativ bzw. quantitativ, z.B. in bezug auf Flächennutzung oder Niederschlagsmenge. Dazu finden i. allg. einfache Zahlen oder Zeichenketten, die in ihrer Kombination einen Attributvektor (Tupel) bilden, Verwendung.

Objektmodell

Die zu entwickelnde Speicher- und Zugriffsarchitektur für geographische Datenbanksysteme soll in der Lage sein, Mengen von Objekten bestehend aus geometrischer (EPL) und thematischer (Vektor einfacher Datentypen) Komponente zu verwalten. Bild 2 gibt ein Beispiel.

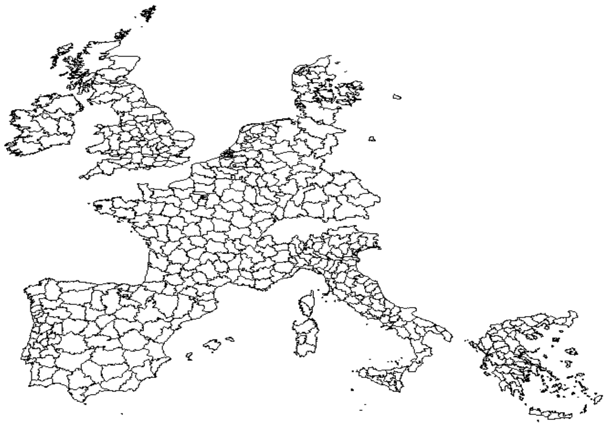

Bild 2: Karte europäischer Verwaltungsbezirke modelliert als Menge von EPL

Beide Komponenten erfordern eine völlig unterschiedliche Handhabung durch die Geo-Architektur. Für die Verwaltung von Vektoren einfacher Datentypen (Tupel) existiert in relationalen Datenbanksystemen eine Vielzahl erprobter und effizienter Datenstrukturen und Algorithmen. Demgegenüber erfordert die Organisation der Geo-Komponente neuartige Strukturen, die in der Lage sind, die beliebig komplexe geometrische Beschreibung so zu verwalten, daß Anfragen, die sich auf Lage und Geometrie der Objekte beziehen, möglichst effizient bearbeitet werden können.

Neben diesen grundsätzlichen Eigenschaften der Geo-Objekte sind für den Entwurf der Architektur zwei weitere Aspekte von entscheidender Bedeutung. Eine möglichst genaue Charakterisierung der Objekte aus realen Anwendungen und eine Spezifikation der auf den Objektmengen auszuführenden Anfragen und Operationen.

2.2 Charakteristika der Objekte

Eine allgemeingültige exakte Beschreibung der Objektmengen, wie sie in Geo-Datenbanksystemen auftreten, läßt sich nicht geben. Zu unterschiedlich sind die Anforderungen aus dem breiten Spektrum geographischer Anwendungen. Eine grobe grundsätzliche Charakterisierung in bezug auf eine Reihe von Eigenschaften ist aber möglich.

Komplexität und Variabilität der Daten

- *Anzahl der Objekte und Datenvolumen*

  Die Anzahl der Objekte in einer geographischen Datenbank hängt sehr stark vom Anwendungsbereich ab, für den sie entworfen wurde sowie von den Eigenschaften des modellierten Ausschnittes der Erdoberfläche. Nach [Fra 91] und [Cra 90] treten in realen Anwendungen Objektmengen mit einer Größe bis zu 10^9 Datensätzen auf. Beim Datenvolumen ist mit bis zu 1 TerraByte zu rechnen. Die zu entwickelnde Geo-Architektur muß in der Lage sein, derartige Datenvolumina (auf dem Sekundärspeicher) zu verwalten.

- *Variabilität in Objekten und Mengen*

  Daten in realen geographischen Anwendungen zeichnen sich durch eine sehr große Variabilität der Objekte und Objektmengen aus [Fra 91]. Dies bezieht sich vorrangig auf:

  - *Objektausdehnung*
    Sie variiert nach [Fra 91] in einer Spanne von $1 : 10^6$, wobei die größten Objekte den gesamten Datenraum umfassen können.

  - *Objektform*

- *Speicherplatzbedarf*
  In der World Data Bank II [GC 87] beispielsweise variiert der Speicherplatzbedarf von Polygonen innerhalb einer Spanne von 0,5 KB bis über 1,1 GB.
- *Verteilung der Objekte in der Ebene*
  Die Anzahl der Objekte pro Flächeneinheit (Dichte) variiert in realen Anwendungen nach [Fra 91] in einer Spanne von $1 : 10^4$.

Insbesondere ist zu beachten, daß es prinzipiell weder für die Objektausdehnung, die Komplexität der Objektstruktur, den Speicherplatzbedarf noch für die Dichte der Objekte Obergrenzen gibt. Bei Auswahl bzw. Entwicklung von Datenstrukturen und Algorithmen für eine Geo-Architektur ist dies zu beachten.

Dauerhafte Speicherung des Datenbestandes in schwach dynamischer Umgebung

Die Erfassung des Datenbestandes eines geographischen Datenbanksystems ist ungleich aufwendiger als bei Standardsystemen im betriebswirtschaftlich-administrativen Bereich. Als Datenbasis liegen oft Tausende von Papierkarten vor, die aufwendig digitalisiert und nahtlos in eine Gesamtdatenbank integriert werden müssen. Dieses weitgehend manuelle Vorgehen ist eine Quelle von Ungenauigkeiten und Inkonsistenzen, die durch aufwendige Konsistenzprüfungen auf der Datenbank entdeckt und beseitigt werden müssen. Eine weitere Quelle geometrischer Daten sind Satellitenaufnahmen von der Erdoberfläche, die aufbereitet und in das Vektorformat konvertiert werden. Insgesamt macht die Datengewinnung und Konsistenzerhaltung ca. 80% der Betriebskosten eines geographischen Datenbanksystems aus [Arn 90].

Ist der Datenbestand einmal erfaßt, so wird er persistent gespeichert und langfristig genutzt. Dabei bleiben die Daten i. allg. jedoch nicht statisch, sondern unterliegen noch Veränderungen durch die nachträgliche Korrektur von Fehlern oder Inkonsistenzen und durch Veränderungen im modellierten Weltausschnitt. Insgesamt ist der Datenbestand als schwach dynamisch zu charakterisieren.

Die aufgeführten Merkmale der Geo-Objekte und die im folgenden beschriebenen Anfragen und Operationen bilden gemeinsam eine Art Anforderungsdefinition für die zu entwickelnde Speicher- und Zugriffsarchitektur.

2.3 Anfragen und Operationen

Aufgrund der sehr unterschiedlichen Anwendungssituationen, in denen geographische Datenbanksysteme eingesetzt werden, ist es nicht möglich einen standardisierten Satz von Anfragen und Operationen anzugeben, der allen Anforderungen genügt [SV 89]. Statt dessen unterscheiden wir 4 Basis-Operationsklassen und benennen jeweils typische Vertreter, die durch die Zugriffsarchitektur unterstützt werden sollen.

1) Modifikationen

Analog zu Standard-Datenbanksystemen stehen in einem Geo-Datenbanksystem Operationen zur Verfügung, die Datensätze zur Datenbank hinzufügen, sie löschen oder modifizieren.

2) Selektionen

Grundsätzlich lassen sich zwei Arten von Selektionen unterscheiden; solche, die sich auf die räumliche, und solche, die sich auf die thematische Komponente der Objekte beziehen.

a) Räumliche Selektionen:

Bei den räumlichen Selektionen werden Objekte anhand ihrer räumlichen Lage und Ausdehnung aus einer Menge von Objekten selektiert.

- *Ortsbezogene Selektion: Punktanfragen*
  Für einen gegebenen Anfrageort (Punkt) P und eine gegebene Objektmenge M liefert eine Punktanfrage *(Point Query)* alle Objekte aus M, die sich am Ort P befinden, die P also geometrisch enthalten (siehe Bild 3(a)).
- *Bereichsbezogene Selektion: räumliche Bereichsanfragen*
  Für eine gegebene polygonale Anfrageregion R und eine gegebene Objektmenge M liefert die *Region Query* alle Objekte aus M, die die Anfrageregion R schneiden. Ein Spezialfall der Region Query ist

die *Window Query*, bei der die Anfrageregion durch ein achsenparalleles Rechteck gebildet wird (vgl. Bild 3(b)).

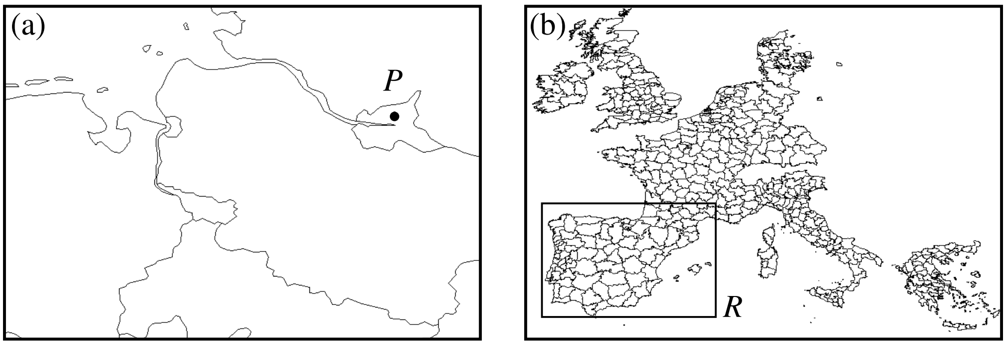

Bild 3: Beispiele für ortsbezogene und bereichsbezogene Selektion

b) Thematische Selektionen:

Bei den thematischen Selektionen werden Objekte anhand bestimmter Eigenschaften ihrer thematischen Komponente aus einer Menge von Objekten selektiert. In diesem Abschnitt konzentrieren wir uns zunächst auf die geometrische Komponente der Objekte. Die Unterstützung thematischer Selektionen wird in Abschnitt 4.5 betrachtet.

3) Kombinationen

- *Spatial Join*
  Die räumliche Join-Operation liefert für zwei Objektmengen A und B alle Paare (a, b), $a \in A$, $b \in B$, deren geometrische Komponenten sich schneiden. Für jedes Objekt $a \in A$ sind also gerade die Objekte in B zu finden, die sich mit a überlappen. Für die effiziente Ausführung des Spatial Joins ist ein geometrisch selektiver Zugriff auf die Objekte notwendig [BKS 93c].
- *Map Overlay*
  Der Map Overlay ist eine der wichtigsten Operationen in einem Geo-Informationssystem [Bur 86]. Er kombiniert zwei (oder mehr) Mengen von Geo-Objekten. Diese Verknüpfung wird über eine Overlay-Funktion gesteuert, die bestimmt, welche Schnittobjekte in welcher Form zur Ergebnismenge gehören. Der Map Overlay basiert somit auf dem Spatial Join bzw. dessen Varianten. Zusätzlich zum Spatial Join müssen die tatsächlichen Schnittobjekte zwischen zwei Ausgangsobjekten berechnet oder benachbarte Objekte, die die gleiche Thematik besitzen, miteinander verschmolzen werden [KBS 91].

4) Auswertungen

Den Selektionen bzw. Kombinationen bestehender Objektmengen sind innerhalb von Anwendungsprogrammen häufig weitere Verarbeitungsschritte nachgeschaltet. Die dazu notwendigen Operationen und Verfahren sind sehr spezifisch für die jeweilige Anwendung und deshalb selbst nicht Teil einer allgemeinen Speicher- und Zugriffsarchitektur. Grundsätzlich unterscheiden lassen sich aber:

- *automatische Analyse*
  In die Klasse der Analysefunktionen fallen Berechnungen, die auf der geometrischen und oder der thematischen Komponente der Objekte durchgeführt werden. Typische Vertreter sind z.B. Durchschnittsberechnungen von Fläche oder Umfang von Objekten, Berechnung von Abständen und Wegelängen zwischen Objekten, Ermittlung von Minimum oder Maximum bestimmter thematischer Attribute usw.
- *Visualisierung*
  In vielen Fällen ist die automatische Analyse eines Datenbestandes nicht möglich, sondern es sind manuelle Zwischenschritte durch den Benutzer zur vollständigen Analyse erforderlich. Dafür ist eine Visualisierung der Daten auf graphischen Ausgabemedien (i.d.R. Bildschirm) erforderlich.

Die Aufstellung macht deutlich, daß den räumlichen Selektionen innerhalb der Anfragen und Operationen eine besondere Bedeutung zukommt. Sie sind nicht nur eine eigenständige Anfrageklasse, sondern fungieren auch als wichtigste Basisoperationen der Klassen 2 - 4. Ihre effiziente Realisierung ist daher notwendige Voraussetzung für ein gutes Leistungsverhalten des gesamten geographischen Datenbanksystems.

3 Ein Phasenmodell für die geometrische Anfragebearbeitung

Nachdem wir sowohl Objekte als auch Anfragen und Operationen klassifiziert und ihre Charakteristika beschrieben haben, soll jetzt eine Architektur zur Speicherung der Geo-Objekte und zur Bearbeitung der Anfragen aufgebaut werden. Vorrangige Aufgabe dieser Architektur ist die *effiziente* Bearbeitung der geometrischen Anfragen und Operationen. Daher betrachten wir diese Anfragen näher, unterscheiden verschiedene Phasen in ihrer Bearbeitung und geben Algorithmen und Verfahren zur Unterstützung der einzelnen Phasen an.

Wie im vorangegangenen Kapitel beschrieben, sind die räumlichen Selektionen die wichtigsten Basisoperationen bei der Bearbeitung geometrischer Anfragen. Ihre Durchführung läßt sich in verschiedene Schritte einteilen.

Schritt 1: Einschränkung des Datenraumes

Bei räumlichen Selektionen ist es möglich, die Suche nach Objekten auf einen geometrischen Bereich einzuschränken, in dem überhaupt nur Kandidaten liegen können, die die Anfrage erfüllen.

Soll die Einschränkung des Datenraumes möglichst effizient erfolgen, ist der Einsatz von Zugriffspfaden erforderlich, die die Objekte entsprechend ihrer räumlichen Lage und Ausdehnung im Datenraum indizieren und schnell zugreifbar machen. Alle Objekte, die eine Anfrage erfüllen, liegen wegen der geometrischen Selektivität stets auch räumlich benachbart zueinander. Um einen effizienten Objektzugriff zu unterstützen, sollten räumlich nahe beieinanderliegende Objekte daher durch den Zugriffspfad auch möglichst physisch nahe beieinander gespeichert werden, d.h. *geclustert* werden.

Wegen der bereits beschriebenen Komplexität der Geo-Objekte ist es nicht möglich, über die vollständige Ausdehnungsinformation der Objekte zu indizieren. Ein Zugriffspfad kann daher auch nicht das exakte Ergebnis einer Anfrage liefern, sondern nur einen möglichst großen Teil der Objekte sofort von der Ergebnismenge ausschließen. Es verbleibt eine Menge von Kandidaten, die in der Ergebnismenge liegen können und an Schritt 2 der Anfragebearbeitung weitergeleitet werden. Diese Art der zweistufigen Anfragebearbeitung ist aus [Ore 89] unter den Begriffen *Filter- und Verfeinerungsschritt* bekannt.

Schritt 2: Genauere Untersuchung der Objekte

In Schritt 2 der Anfragebearbeitung werden die Kandidaten daraufhin untersucht, ob sie die Anfrage tatsächlich erfüllen. Dazu ist die Auswertung eines geometrischen Prädikates wie z.B. das Enthaltensein eines Punktes im Objekt, das Überschneiden mit einem Rechteck o.ä. zu überprüfen. Ganz analog zu Schritt 1 der Anfragebearbeitung können dabei verschiedene Teilschritte unterschieden werden. Zunächst wird die Untersuchung des Objektes auf die für den Test notwendigen lokalen Teile beschränkt. Bild 4 zeigt dazu ein Beispiel: Um zu entscheiden, ob das Anfragefenster den Voltasee schneidet, ist nur dessen nordwestliche Spitze näher zu untersuchen.

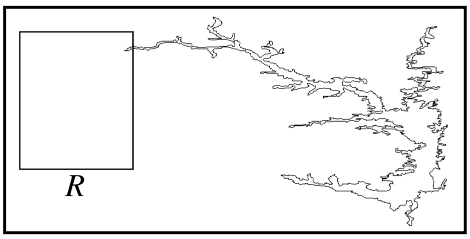

Bild 4: Test eines Anfragebereichs gegen den Voltasee

Wegen der Komplexität der Objekte einerseits und der geometrischen Selektivität der Anfragen andererseits, ist es sinnvoll, die komplexen Objekte lokal zu strukturieren und die entstehenden Strukturelemente bezüglich ihrer Lage und Ausdehnung räumlich zu organisieren. Auf den für die jeweilige Anfrage tatsächlich relevanten Objektteilen wird dann der geometrische Test durchgeführt. Dazu kommen Verfahren aus dem Gebiet der algorithmischen Geometrie zum Einsatz, die schließlich entscheiden, ob ein Objekt die Anfrage tatsächlich erfüllt oder nicht.

Schritt 3: Ausgabe des Objektes zur Weiterverarbeitung

Ist ein Objekt als Teil des Ergebnisses identifiziert, so muß es möglichst schnell als Gesamtobjekt verfügbar gemacht werden, z.B. um weiterverarbeitet zu werden. Dies macht eine physisch zusammenhängende Speicherung aller zum Objekt gehörenden Teile erforderlich, die durch eine spezielle Speicherungsorganisation realisiert werden muß [Wei 89].

4 Eine Architektur für die Anfragebearbeitung in Geo-Datenbanksystemen

Nach der abstrakten Beschreibung des Phasenmodells in der geometrischen Anfragebearbeitung geben wir im folgenden algorithmische Techniken zur Unterstützung der einzelnen Phasen an. Sie kommen später als Bausteine zur effizienten Durchführung der einzelnen Schritte in unserer Gesamtarchitektur zum Einsatz.

4.1 Räumliche Indexstrukturen

Indexstrukturen als elementarer Bestandteil der internen Ebene eines Datenbanksystems dienen dazu, eine dynamische Menge von Objekten so auf einem seitenorientierten Sekundärspeicher zu organisieren, daß eine Menge von Anfragen und Operationen auf dem Datenbestand effizient, d.h. unter möglichst geringem Bedarf an Sekundärspeicherzugriffen und CPU-Zeit bearbeitet werden kann.

In Standard-Datenbanksystemen kommen für solche Zwecke vornehmlich B-Baum-Varianten zum Einsatz. Mit ihnen lassen sich linear geordnete Schlüssel von N Objekten so organisieren, daß für eine Modifikation oder exakte Suche nur auf log(N) Seiten zugegriffen werden muß.

Für geographische Datenbanksysteme sind B-Bäume und andere eindimensionale Indexstrukturen jedoch nicht geeignet. Hier sind vielmehr Strukturen gesucht, die in der Lage sind, flächige geometrische Objekte (einfache Polygone mit Löchern) bezüglich ihrer Lage und Ausdehnung in der Ebene zu indizieren. Beliebige einfache Polygone mit Löchern sind jedoch zu komplex, um ihre gesamte Lage und Ausdehnungsinformation, d.h. die gesamte Objektbeschreibung, bei der Indizierung berücksichtigen zu können. Statt dessen betrachten wir zunächst die räumliche Indizierung einfacherer Flächenobjekte. Übersichten über räumliche Indexstrukturen sind z.B. in [Sam 90] und [Wid 91] enthalten.

Die einfachste Klasse ausgedehnter Geo-Objekte sind *achsenparallele Rechtecke*. Für diese Objektklasse existiert bereits eine Reihe von Indexstrukturen mit dem *R-Baum* [Gut 84] als einem bekannten Vertreter. Der R-Baum faßt soviele räumlich nahe beieinanderliegende Objekte (Rechtecke) zusammen, wie auf eine (Daten-) Seite passen, speichert sie dort gemeinsam ab und umschreibt sie mit einem *minimal umgebenden achsenparallelen Rechteck (MUR)*. Eine Menge derartiger Rechtecke wird ihrerseits auf einer (Directory-) Seite gespeichert und wiederum mit einem MUR umschrieben. So wird die gesamte Objektmenge stufenweise räumlich zusammengefaßt und es entsteht ein baumartiges Directory aus Rechtecken (Bild 5).

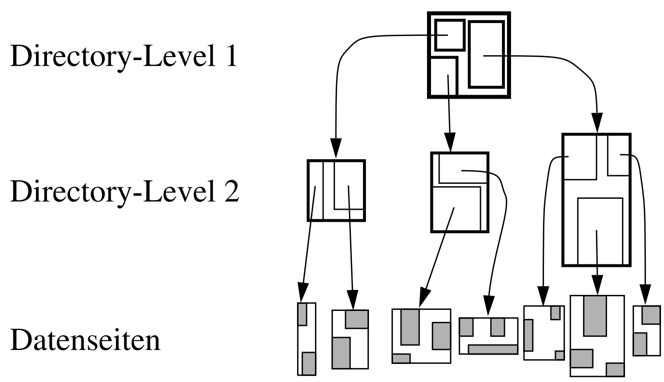

Bild 5: Schematische Darstellung eines R*-Baumes

Eine besonders effiziente Version des R-Baumes ist der *R*-Baum* [BKSS 90], der eine ausgefeilte Strategie zur Partitionierung von Seitenregionen verwendet und bei Bedarf auch Umstrukturierungen im Baum vornimmt.

Diese Art räumlicher Organisation der Rechtecke erlaubt eine effiziente Bearbeitung von Punkt- und kleinräumigen Bereichsanfragen [BKSS 90], allerdings eben nur auf Mengen achsenparalleler Rechtecke oder anderer sehr einfacher Objekte, die in eine Datenseite passen. In realen Anwendungen ist es jedoch notwendig, sowohl komplexere Objekte speichern als auch größere Bereichsanfragen effizient bearbeiten zu können. Für komplexere Objektklassen und großräumigere Anfragen existieren bislang nur sehr wenige Vorschläge für Indexstrukturen. Die im späteren Teil dieses Kapitels vorgestellte Zugriffsarchitektur wird in der Lage sein, Mengen einfacher Polygone mit Löchern zu speichern und auch großräumige Anfragen effizient zu beantworten.

4.2 Approximationen

Zur Ergebnismenge einer geometrischen Anfrage gehören all die Objekte einer Objektmenge, die ein geometrisches Prädikat erfüllen, also z.B. einen Punkt enthalten, von einem gegebenen Objekt geschnitten werden o.ä.. Wie im vorangegangenen Abschnitt beschrieben, dienen räumliche Indexstrukturen dazu, möglichst viele Objekte von vornherein von der Ergebnismenge auszuschließen. Es verbleiben Kandidaten, die genauer geometrisch getestet werden müssen. Dies ist bei sehr komplexen Objekten (Polygonen mit sehr vielen Eckpunkten) außerordentlich zeitaufwendig [KHS 91]. So liegt es nahe, dem eigentlichen Test auf dem Objekt einen Vortest voranzustellen, um für möglichst viele Kandidaten schon dort zu entscheiden, daß sie die Anfrage sicher erfüllen oder sicher nicht erfüllen.

Für einen derartigen Vortest eignet sich das Konzept der *Objektapproximation*. In [Kri 91] ist eine detaillierte Klassifikation verschiedener Approximationen gegeben. Eine Approximation hat eine im Vergleich zum Objekt einfache Beschreibung und nähert das Objekt möglichst gut an - zwei offensichtlich konkurrierende Kriterien. Um Objektapproximationen für einen geometrischen Vortest effizient nutzen zu können, muß das Objekt vollständig in der Approximation enthalten sein *(konservative Approximation)* [Sch 92]. Beispiele für konservative Approximationen sind achsenparallele Rechtecke, konvexe n-Ecke, Ellipsen usw. (Bild 6).

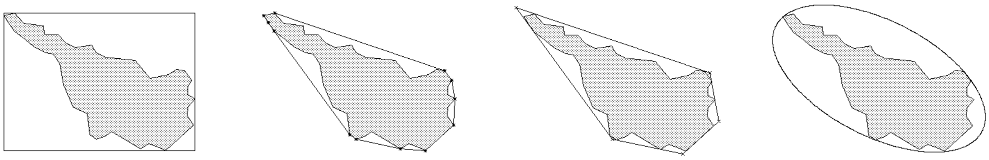

Bild 6: Verschiedene konservative Approximationen

Für eine Punktanfrage beispielsweise wird dann innerhalb der Anfragebearbeitung für alle Kandidatenobjekte zunächst getestet, ob ihre Approximation den Anfragepunkt enthält. Aufgrund der Einfachheit der Approximationsobjekte ist dieser Test sehr schnell durchführbar. Fällt der Test negativ aus, d.h. enthält die Approximation den Punkt nicht, so kann ihn auch das Objekt nicht enthalten und der Kandidat wird verworfen. Ein teurer Punkt-In-Polygon-Test kann so eingespart werden. Nur im Falle eines positiven Vortests ist der Test auf dem Objekt selber noch durchzuführen. Von seinem Ergebnis hängt endgültig ab, ob das Objekt zur Ergebnismenge gehört oder nicht. Wegen ihrer einfachen Beschreibbarkeit eignen sich Approximationsobjekte gut für die Speicherung in räumlichen Indexstrukturen (vgl. Abschnitt 4.2).

In [BKS 93a] werden Approximationsverfahren für geometrische Objekte ausführlich beschrieben und ihre Eignung anhand von Datenbanken aus realen Anwendungen evaluiert. Das minimal umgebende Fünfeck hat sich dabei als bester Kompromiß zwischen Approximationsgüte und Speicherplatzbedarf erwiesen. Weiterhin wird gezeigt, daß sich der R*-Baum auch gut zur Organisation anderer Approximationen als minimal umgebender Rechtecke eignet.

4.3 Objektzerlegungen

Während die Objektapproximation das Ziel hat, möglichst häufig geometrische Tests auf den Polygonen ganz zu verhindern, dienen *Objektzerlegungen* dazu, diese Tests zu vereinfachen, d.h. schneller ausführbar zu machen.

Betrachten wir wieder den Test, ob ein Polygon einen gegebenen Anfragepunkt enthält. Zur Durchführung dieses Tests ist ein Algorithmus mit in der Zahl der Polygonpunkte linearer Laufzeit erforderlich [PS 88]. Bei detailreichen Polygonen mit vielen tausend (reellwertigen) Polygonpunkten führen solche Tests zu erheblichen Laufzeiten. Andererseits fällt auf, daß für die Entscheidung des Tests nur ein sehr kleiner lokaler Teil des Objektes wirklich relevant ist. Diese Beobachtung legt es nahe, die Objekte in einfache lokale Bereiche einzuteilen und für die Durchführung des Tests möglichst nur einen dieser lokalen Bereiche zu betrachten. Dies führt zum Konzept der *Objektzerlegung*, bei dem ein Polygon in eine Menge einfacherer lokal beschränkter Teilkomponenten zerlegt wird. In [KHS 91] und [Kri 91a] wird der Zerlegungsansatz für einfache Polygone mit Löchern ausführlich vorgestellt und diskutiert. Als Teilkomponenten kommen einfache Objektklassen wie Dreiecke, Trapeze, konvexe Polygone usw. in Frage.

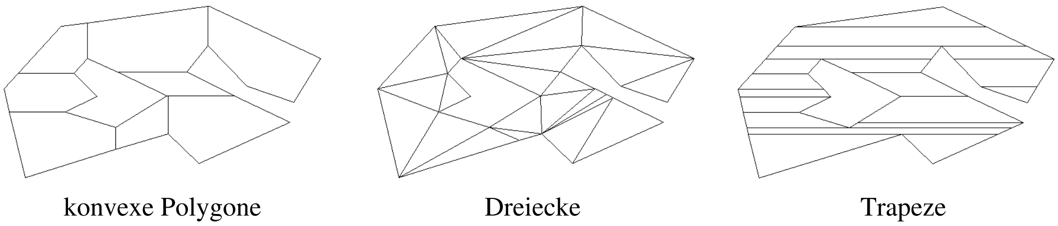

Bild 7: Verschiedene Zerlegungsverfahren für einfache Polygone mit Löchern

Geometrische Tests werden dann z.B. lediglich auf einem Trapez durchgeführt, was wesentlich effizienter ist, als das gesamte Polygon für den Test zu betrachten. Um effizient entscheiden zu können, welche Komponente oder Komponenten für den jeweiligen Test tatsächlich relevant sind, werden sämtliche Teilkomponenten eines Objektes in einem speziellen R*-Baum - *TR*-Baum* genannt - räumlich organisiert abgespeichert. Der TR*-Baum ist eine Hauptspeicher-Indexstruktur zur Repräsentation eines Objektes, die zur Durchführung des Tests vollständig in den Hauptspeicher geladen wird. In [SK 91] zeigen wir, daß der TR*-Baum verschiedene geometrische Anfragen und Operationen effizient unterstützt.

4.4 Szenen-Organisation

Eine wichtige Anforderung an Geo-Datenbanksysteme ist die *Mengenorientierung*. Neben kleinräumigen Anfragen muß ein Geo-Datenbanksystem in der Lage sein, großräumige Anfragen effizient zu bearbeiten. Bei solchen Anfragen werden große Datenmengen vom Sekundärspeicher in den Hauptspeicher übertragen. Die bisher in diesem Beitrag vorgestellten Konzepte unterstützen durch selektiven Objektzugriff insbesondere Punkt- und kleinräumige Bereichsanfragen; großräumige Anfragen erfahren durch diese Ansätze keine besondere Effizienzsteigerung. Daher benötigen wir in unserer Architektur ein Konzept, das eine Mengenorientierung unterstützt; wir werden es später als *Szenen-Organisation* bezeichnen.

Betrachten wir die bisherige Speicherorganisation und die zu speichernden Geo-Objekte, so können folgende Beobachtungen gemacht werden:

- Die Geo-Objekte sind oft im Vergleich zu den Seiten, auf denen die Objekte gespeichert werden, sehr groß. Oft passen nur wenige Objekte in eine Seite. Etliche Objekte benötigen auch bei einer Seitenkapazität von 4 KByte mehrere Seiten Speicherplatz (vgl. Abschnitt 2.2).
- Die Seiten werden räumlich unabhängig voneinander auf dem Sekundärspeicher abgelegt; d.h. räumlich benachbarte Seiten sind nicht physisch nah gespeichert. Größere Bereichsanfragen lesen aber eine Reihe räumlich benachbarter Seiten in den Hauptspeicher ein. Folglich müssen die Seiten auf dem Sekundärspeicher von verschiedenen Stellen "zusammengesammelt" werden. Dadurch entsteht ein erheblicher Leistungsverlust.

Die in diesem Abschnitt präsentierten Konzepte stellen somit nur eine *lokale Ordnung* innerhalb von Seiten sicher [Wid 91]. Um die Mengenorientierung geeignet zu unterstützen, ist eine *globale Ordnung* notwendig, d.h. größere räumlich zusammenhängende Bereiche der Geo-Daten werden physisch nahe gespeichert.

Um größere Speicherbereiche zu handhaben, lassen sich unterschiedliche Wege gehen. In [Wei 89] werden die *Nutzung größerer Seiten*, *Seiten mit variabler Größe*, *verschiedene Pufferstrategien* und *physikalische Clusterung von Seiten* kombiniert mit einer *mengenorientierten Seitenschnittstelle* zur Handhabung großer komplexer Objekte detailliert diskutiert. Innerhalb dieser Arbeit wird die physikalische Clusterung von Seiten favorisiert und bietet sich als adäquater Ansatz zur Speicherung sog. *Szenen* (Begriffsdefinition folgt) innerhalb unserer Geo-Architektur an. Um diesen Ansatz praktisch umzusetzen, wird eine mengenorientierte Schnittstelle zwischen Datenbanksystem und Sekundärspeichermedium benötigt. Ein derartiges Interface erlaubt den effizienten Transfer von Mengen physikalisch benachbarter Seiten vom Sekundär- in den Hauptspeicher. Die Implementierung eines solchen Interfaces soll hier allerdings nicht weiter erörtert werden.

In [HSW 88] wurde vorgeschlagen, den Erhalt der globalen Ordnung geometrischer Punktdaten über *dynamisches z-hashing* zu realisieren. Dieser Ansatz wurde in [HWZ 91] auf räumliche Daten mit Ausdehnung übertragen. Dabei wird allerdings nur die globale Ordnung bezüglich der Approximationen sichergestellt. Außerdem ist dieser Ansatz für Hash-Verfahren und nicht für Verfahren mit beliebiger Raumaufteilung einsetzbar. Daher haben wir ein Konzept entwickelt, das auf der Partitionierung des Raumes durch den R*-Baum beruht.

Der Aufbau der Szenen-Organisation

Wie bereits erwähnt, nutzen wir den R*-Baum wegen seines guten Leistungsverhaltens und seiner Robustheit als integrale Komponente unserer Geo-Architektur. Der R*-Baum verwendet ein sehr effizientes Schema zur Partitionierung, ohne die Objekte zu clippen oder in einen anderen Raum zu transformieren. So liegt die Idee nahe, Partitionen, d.h. ausgewählte Teilbäume des R*-Baumes als Einheiten der physikalischen Clusterung zu verwenden. Im folgenden ist eine *Szene* als ein Teilbaum des R*-Baumes definiert und physikalisch auf dem Sekundärspeicher geclustert. Eine Szene besteht aus einer größeren Menge physisch benachbarter Seiten, die alle zugehörigen Geo-Objekte umfaßt.

Benötigt ein Geo-Objekt mehrere Seiten Speicherplatz, so werden diese Seiten innerhalb der Szene physisch zusammenhängend abgespeichert. Damit kann auch ein einzelnes, größeres Objekt durch eine Operation vom Sekundärspeicher vollständig in den Hauptspeicher übertragen werden (vgl. Schritt 3 im Phasenmodell). Ansonsten unterliegen die Seiten innerhalb einer Szene keiner Ordnung. Sie werden wie bisher vom R*-Baum verwaltet.

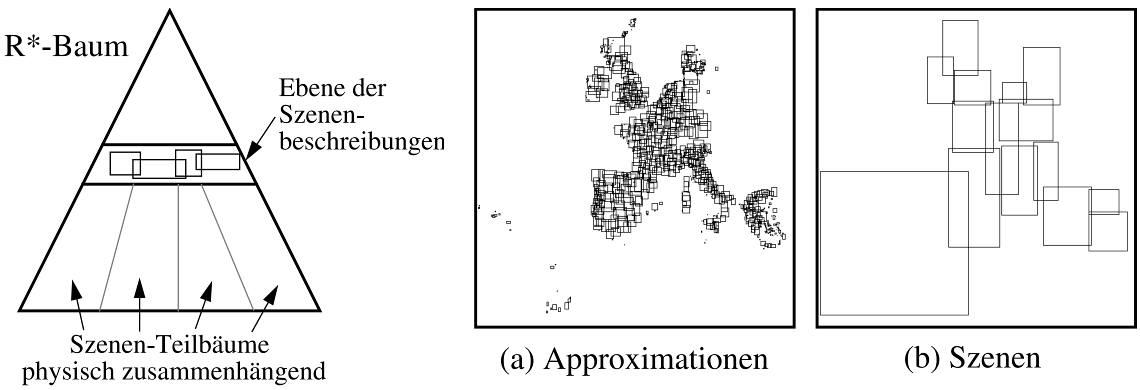

Bild 8: Schematische und beispielhafte Darstellung der Szenen-Organisation

Neben einer schematischen Darstellung der Szenen-Organisation zeigt Bild 8 den Szenenaufbau an einem Beispiel. Die Daten sind die Verwaltungsbezirke der EG (siehe Bild 1). In einem R*-Baum sind die Polygone, die diese Bezirke geometrisch modellieren, deren Teilkomponenten und die Objektapproximationen (Bild 8 (a)) organisiert. Bild 8 (b) zeigt die Partitionierung des R*-Baumes auf einer höheren Directory-Ebene; diese Rechtecke beschreiben die Szenen im Baum; die zugehörigen Teilbäume sind physisch zusammenhängend gespeichert.

Anfragebearbeitung

In einer Szenen-Organisation können sowohl klein- als auch großräumige Anfragen effizient bearbeitet werden. Kleinräumige Anfragen werden wie bisher seitenweise abgearbeitet. Besitzt eine Bereichsanfrage hingegen einen größeren Anfragebereich, werden die betroffenen Szenen, die den Anfragebereich schneiden, vollständig in den Hauptspeicher gelesen. Dieses kann sehr effizient erfolgen, da die Seiten der Szene physisch benachbart sind. Damit ist auf dem Plattenspeicher nur eine Suchoperation notwendig, um die Szene einzulesen. Ohne die Szenen-Organisation müßte jede Seite einzeln angesteuert werden. Allerdings kann eine Szene auch eine Reihe von Geo-Objekten umfassen, die die Anfrage nicht erfüllen *(Fehltreffer, "False Hits")*. Dieses führt dazu, daß das Übertragungsvolumen bei der Szenen-Organisation ansteigt. Eine relativ kleine Zahl von False Hits beeinträchtigt die Effizienz aber nicht wesentlich, da der Aufwand für die Suche einer Seite den Aufwand für die Übertragung einer Seite bei weitem übersteigt [PH 90]. Zusätzlich kann die Größe des Schnittbereiches mit dem Anfragebereich als Maß für die Entscheidung dienen, ob eine Szene vollständig eingelesen wird oder ob die Anfrage ohne Nutzung der Szenen-Organisation über den R*-Baum abläuft. Eine genauere Untersuchung des Leistungsverhaltens der Szenen-Organisation folgt in Abschnitt 5.1.

Nachdem die Szene in den Hauptspeicher eingelesen wurde, kann die Anfragebearbeitung in der bisherigen Form weiter ablaufen. Eine detaillierte algorithmische Beschreibung der dynamischen Organisation der Szenen-Architektur findet sich in [Sch 92] und [BKS 93b]. Die Ergänzung der bisherigen Techniken zur Anfragebearbeitung um eine Szenen-Organisation erlaubt sowohl eine effiziente kleinräumige Selektion als auch eine Unterstützung großräumiger Bereichsanfragen.

4.5 Einbindung thematischer Attribute

Die bisher vorgestellten Techniken sind vollständig auf die Unterstützung räumlicher Anfragen ausgerichtet. Neben räumlichen Anfragen spielen aber auch Anfragen bezüglich thematischer Attribute eine wichtige Rolle in geographischen Informationssystemen. Die thematischen Attribute (vgl. Abschnitt 2.1) sind in unserem Ansatz der geometrischen Komponente zugeordnet.

Um thematische Anfragen effizient unterstützen zu können, benötigen wir somit neben dem räumlichen Index *Sekundärindizes* (z.B. B-Bäume) für die relevanten thematischen Attribute. Der R*-Baum bestimmt in Verbindung mit der Szenen-Organisation den Ort der physischen Speicherung. Damit dies losgelöst von den Sekundärindizes erfolgen kann, benötigen wir eine *Link-Tabelle*. Die Link-Tabelle ordnet jedem Geo-Objekt, repräsentiert durch ein eindeutiges *Surrogat*, eine Datenseite im R*-Baum zu. Wenn sich nun die Datenseite des Geo-Objektes ändert, wird nur der Eintrag in der Link-Tabelle geändert; Änderungen in den Sekundärindizes sind nicht notwendig. Damit der Zugriff auf die Link-Tabelle vom Raumindex aus erfolgen kann, müssen die Einträge der Geo-Objekte in den Datenseiten um das Surrogat erweitert werden.

Bild 9 zeigt im Rahmen der Gesamtarchitektur das Konzept zur Einbindung von Sekundärindizes über Link-Tabellen und Surrogate. In [Kri 91b] und [Sch 92] wird dieses Konzept ausführlicher erläutert.

4.6 Die Geo-Architektur

Bisher haben wir eine Reihe von prinzipiellen Konzepten und Techniken zur effizienten Anfragebearbeitung in Geo-Datenbanken vorgestellt. Diese betten wir nun in eine Gesamtarchitektur, unsere *Geo-Architektur*, ein. Bild 9 stellt die gesamte Architektur schematisch dar.

Die Basis unserer Architektur bildet der R*-Baum. Er organisiert die Daten seitenweise auf dem Sekundärspeicher und ermöglicht eine effiziente räumliche Indizierung. Damit kann der Suchbereich von räumlichen Anfragen eingeschränkt werden.

Eine räumliche Anfrage durchläuft bei der Wurzel beginnend den R*-Baum. Dabei trifft sie auf eine oder mehrere Szenenbeschreibungen. Übersteigt die Größe des Schnittbereiches von Anfrageregion und Szene einen Schwellenwert, wird eine Szene vollständig in den Hauptspeicher geladen und die weitere Anfragebearbeitung läuft im Hauptspeicher ab; damit entfallen aufwendige Suchoperationen auf dem Sekundärspeicher. Anderenfalls werden die erforderlichen Datenbereiche seitenweise in den Hauptspeicher transferiert.

Die nächste Stufe in der Architektur bilden die Approximationen. Sie unterstützen eine einfache Vorauswahl, ob ein Geo-Objekt eine Anfrage erfüllt oder nicht. Dazu wird die Approximation in den Einträgen der Datenseiten gespeichert, z.B. das minimal umgebende Fünfeck.

Erfüllt die Approximation eines Geo-Objektes die Anfragebedingung, muß das Objekt weiter untersucht werden. Dazu befindet sich in jedem Objekteintrag ein Verweis auf die exakte Geometrierepräsentation. Diese wird durch den TR*-Baum verwaltet. Er strukturiert die Zerlegungskomponenten des Geo-Objektes räumlich. Damit kann die Lokalität von Anfragen ausgenutzt und auf die entscheidenden Teilkomponenten direkt zugegriffen werden. Anhand dieser einfachen geometrischen Teilobjekte kann die Anfragebedingung überprüft werden. Dieses Vorgehen erspart den aufwendigen Test mit dem Gesamtobjekt.

Abgerundet wird die Architektur durch Sekundärindizes für thematische Attribute. Eine thematische Anfrage läuft den B-Baum ab. Dadurch erhält man einen oder mehrere Surrogate. Über das Surrogat greift man auf die Link-Tabelle zu und erhält die Nummer der Datenseite, die den zugehörigen Objekteintrag enthält.

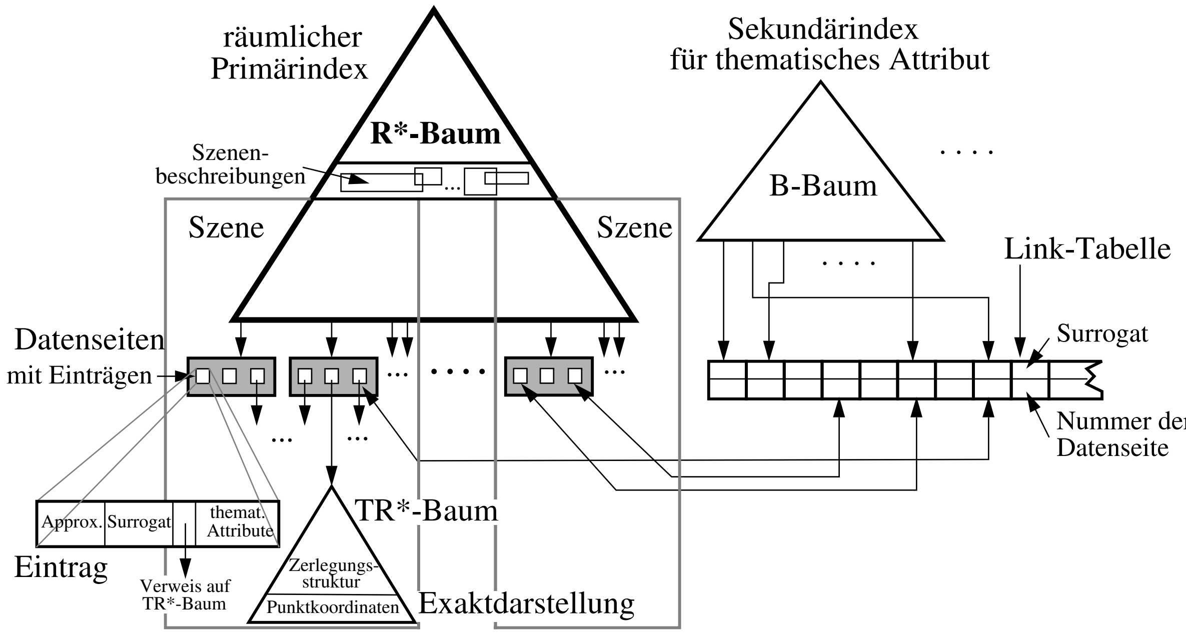

Bild 9: Integration einzelner effizienter Bausteine in unsere Geo-Architektur

5 Empirische Evaluierung der Geo-Architektur

Die Techniken, die in der von uns vorgeschlagenen Architektur für Geo-Datenbanksysteme verwendet werden, wurden bereits intensiv empirisch untersucht. Die Basis des Systems ist der R*-Baum. Leistungsdaten wurden in [BKSS 90] vorgestellt und zeigen gegenüber anderen R-Baum-Varianten einen beträchtlichen Leistungsgewinn. Ein Leistungsvergleich mit dem R^+-Baum und dem PMR-Quadtree findet sich in [HS 92]. Die unterschiedlichen Approximationen haben wir in [BKS 93a] auf ihre Eignung untersucht und das minimal umgebende Fünfeck als die beste Approximation ermittelt.

Den Zerlegungsansatz haben wir in [Kri 91a] und [KHS 91] untersucht. Insbesondere für kleinräumige Anfragen konnten für die konvexe und die Trapez-Zerlegung entscheidende Leistungsgewinne gegenüber dem unzerlegten Ansatz erzielt werden. Die integrierte Repräsentation eines in Trapeze zerlegten Geo-Objektes durch einen TR*-Baum wurde von uns in [SK 91] betrachtet. Dabei zeigte sich der R*-Baum als Hauptspeicherversion mit niedrigen Verzweigungsgrad geeignet, verschiedenste Anfrage- und Operationstypen gleichmäßig effizient zu bearbeiten.

Die Organisation der Geo-Objekte in Szenen wurde dagegen bisher noch nicht untersucht. Daher wollen wir im folgenden die Szenen-Organisation näher betrachten und sie innerhalb eines Leistungsvergleiches detailliert beurteilen.

5.1 Evaluierung der Szenen-Organisation

Prinzipiell lassen sich drei Ansätze für die Speicherung von Geo-Objekten unterscheiden:

1.) Speicherung der Exaktdarstellung außerhalb der Datenseiten (Ansatz 1)

In den Datenseiten der Indexstruktur befinden sich neben den Approximationen nur Verweise auf die Exaktdarstellung der Geo-Objekte. Die Exaktdarstellung wird außerhalb der Speicherungsstruktur, nicht räumlich organisiert gespeichert, z.B. in sequentiellen Dateien. Dieser Ansatz wird beispielsweise bei Quadtrees verwendet [HS 92]. Anders formuliert stellt die räumliche Indexstruktur für die Approximationen einen Primärindex und für die eigentlichen Geo-Objekte einen Sekundärindex dar. Bild 10 zeigt den Ansatz schematisch. Ein Vorteil dieses Ansatzes ist, daß eine hohe Zahl von Approximationen gemeinsam in einer Datenseite gespeichert wird, d.h. es wird eine maximale lokale Ordnung auf den Approximationen hergestellt. Außerdem gibt es keine prinzipielle Größenbeschränkung für die exakte Repräsentation der Geo-Objekte. Wesentlicher Nachteil ist, daß die Ordnungserhaltung sich nur auf die Approximationen bezieht. Folglich ist auch bei Bereichsanfragen für jeden Zugriff auf die exakte Objektrepräsentation ein zusätzlicher Seitenzugriff notwendig, da diese nicht räumlich organisiert wird.

2.) Speicherung der Exaktdarstellung innerhalb der Datenseiten (Ansatz 2)

Die exakte Repräsentation der Objekte wird zusätzlich zu den Approximationen innerhalb der Datenseiten gespeichert. Damit sind auch diese physisch nah gespeichert und können bei Anfragen gemeinsam in einem Zugriff in den Hauptspeicher eingelesen werden [Wid 91]. Die Indexstruktur ist im Gegensatz zum ersten Ansatz damit ein Primärindex, der die physische Speicherung bestimmt. Wesentlicher Nachteil dieses Ansatzes ist, daß so nur wenige Objekte in eine Seite passen und häufig räumlich benachbarte Geo-Objekte in verschiedene Datenseiten fallen. In Abschnitt 2.2 wurde aufgezeigt, daß in Geo-Datenbanken sehr große Objekte auftreten können, die mehrere Seiten Speicherplatz benötigen. Solche Objekte lassen sich durch diesen Ansatz nur schwer organisieren.

3.) Speicherung der Objekte in einer Szenen-Organisation (Ansatz 3)

Diesen Ansatz haben wir in Abschnitt 4.4 vorgestellt. Hierbei werden größere Teile der Daten in physisch zusammenhängenden, aber nur lokal sortierten Speicherbereichen zusammengefaßt und über den R*-Baum organisiert.

Bild 10 zeigt die drei Ansätze schematisch:

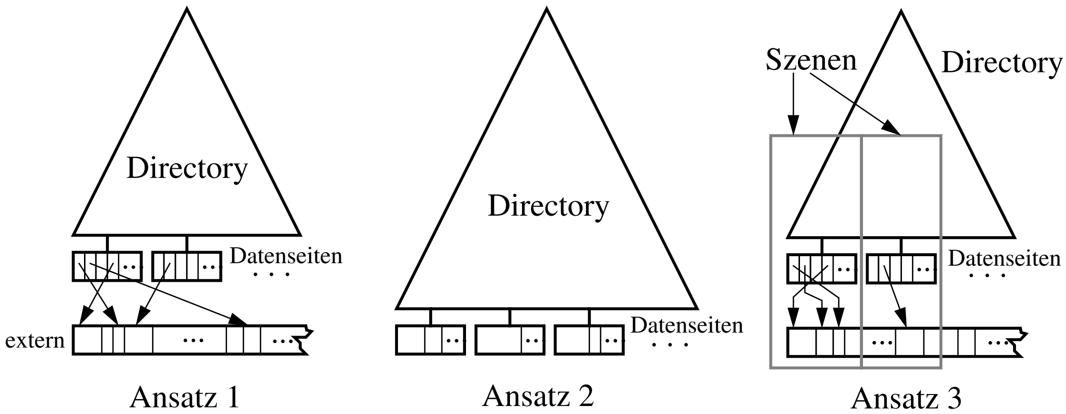

Bild 10: Ansätze zur Speicherung von Geo-Objekten

Die Szenen-Organisation ist für die Unterstützung großräumiger Bereichsanfragen entworfen. Für solche mengenorientierten Anfragen stellen sich zwei Fragen, die im folgenden näher untersucht werden:

- Welches Leistungsverhalten zeigt jeder der drei Ansätze? Kann durch die Szenen-Organisation gegenüber den beiden anderen Speicherungsverfahren ein entscheidender Leistungsgewinn erzielt werden ?
- Bei welcher Szenengröße zeigt die Architektur das beste Leistungsverhalten? Hängt diese Größe stark von der Größe der Bereichsanfragen ab ?

Testumgebung

Um diese Fragestellungen beantworten zu können, haben wir die drei Ansätze in empirischen Untersuchungen miteinander verglichen. Testdaten waren reale Daten vom US Bureau of the Census [Bur 89], die die Verwaltungsgrenzen, Straßenzüge, Eisenbahnlinien und Flüsse von vier kalifornischen Counties repräsentieren. In dieser Datenbasis befanden sich 119.151 Linienzüge, die jeweils aus 2 bis 349 Punkten bestehen. Jede Koordinate wird durch eine Real-Zahl von 8 Byte dargestellt, so daß sich ein Gesamtdatenvolumen von 15,9 MByte ergibt. Approximiert wurden diese Linienzüge durch minimal umgebende Rechtecke, für deren Repräsentation 16 Byte zur Verfügung stehen. Diese Rechtecke sind in Bild 11 (a) dargestellt.

Aus diesen Daten wurden gemäß der drei Ansätze R*-Bäume erzeugt. Die Seitenkapazität betrug dabei 4 KByte.

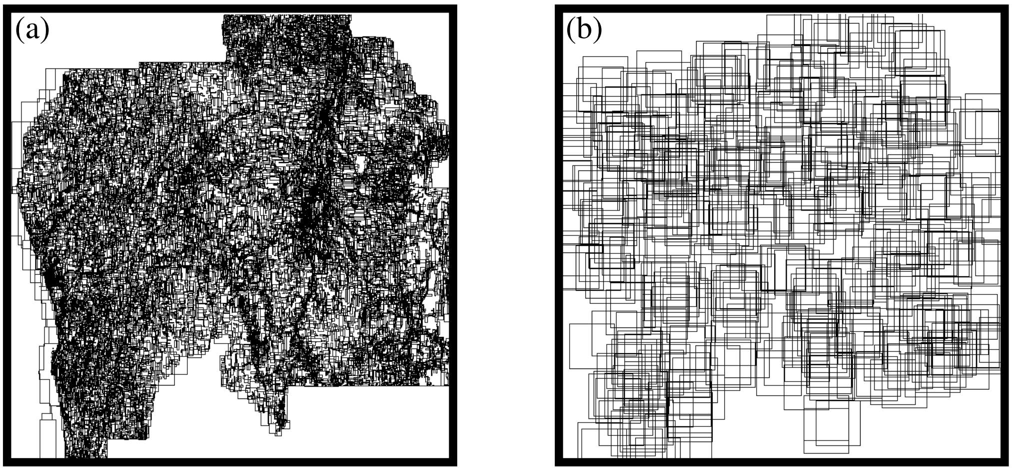

Bild 11: Darstellung der Testdaten und -anfragen

Um Aussagen über das Leistungsverhalten der Verfahren für größere Bereichsanfragen zu gewinnen, haben wir vier Testserien mit unterschiedlich großen Anfragebereichen untersucht. Jede Testserie bestand aus 464 quadratischen Bereichsanfragen (Window Queries), die über den von Objekten überdeckten Datenraum gleichverteilt waren. Die Größe der Anfragebereiche variiert zwischen 0,25% und 16% der Fläche des Gesamtdatenraumes. Bild 11 (b) zeigt die Anfragen bei einer Fläche von 1%. In Tabelle 1 sind die Anfragen der vier Testserien genauer spezifiziert.

Testserie	Größe der Anfragen (Prozent vom Gesamtdatenraum)	pro Testserie		durchschnittlich pro Anfrage	
		eingelesene Datensätze	eingelesene Daten (KByte)	eingelesene Datensätze	eingelesene Daten (KByte)
I	0,25 %	189.229	29.392	408	63
II	1 %	714.937	105.521	1.541	227
III	4 %	2.687.648	382.483	5.792	824
IV	16 %	9.462.455	1.315.236	20.393	2.835

Tabelle 1: Charakteristika der Testserien

Für die Bewertung der Effizienz der Verfahren wird ein Maß zur Beschreibung des Zugriffsaufwandes benötigt. Wie bereits erwähnt, lassen sich für einen Seitenzugriff die Zeit für die Suche der Seite auf dem Sekundärspeicher und die Zeit für das Lesen und den Transfer der Seite vom Sekundärspeicher in den Hauptspeicher unterscheiden. Für die Bestimmung des Gesamtaufwandes bewerten wir im folgenden den Aufwand für die Suche im Vergleich zum Übertragungsaufwand im Verhältnis 10 zu 1. Dieses Verhältnis entspricht in etwa realen Magnetplattenspeichern [PH 90]. Ist N_S die Anzahl der Suchoperationen und N_T die Anzahl der Transfers, gilt für den Zugriffsaufwand A:

$$A = 10N_S + N_T$$

Für Bereichsanfragen sind die Zugriffskosten, die durch den R*-Baum verursacht werden, im Verhältnis zu den Kosten für die Zugriffe auf exakte Repräsentationen gering. Daher betrachten wir im folgenden nur die Zugriffe, die durch das Einlesen der exakten Darstellung der Geo-Objekte entstehen.

Testergebnisse

Tabelle 2 zeigt den Zugriffsaufwand bei Speicherung der Linienzüge außerhalb der Datenseiten (Ansatz 1). Die Anzahl der Suchoperationen (N_S), die Zahl der Transfers (N_T) und der Zugriffsaufwand A sind dort gerundet in Tausend dargestellt. Durch das Runden ergibt sich A in den folgenden Tabellen nicht genau aus N_S und N_T.

Testserie I (0,25 %)			Testserie II (1 %)			Testserie III (4 %)			Testserie IV (16 %)		
N_S	N_T	A	N_S	N_T	A	N_S	N_T	A	N_S	N_T	A
189	189	2.082	715	715	7.864	2.688	2.689	29.585	9.462	9.463	104.088

Tabelle 2: Zugriffsaufwand Ansatz 1 (in Tausend, gerundet)

Bedingt durch den Ansatz, die Exaktdarstellung auszulagern, ist für jeden angefragten Datensatz mindestens eine (teure) Suchoperation notwendig, da die exakten Repräsentationen nicht räumlich organisiert sind. Nach jeder Suchoperation ist mindestens ein Transfer notwendig. Die Tatsache, daß die Transferzahl nicht wesentlich höher als N_S ist, resultiert aus der geringen Zahl von Datensätzen, die größer als eine Seite sind.

In Tabelle 3 sind die Ergebnisse für das zweiten Speicherverfahren angegeben, bei dem die Linienzüge innerhalb der Datenseiten gespeichert werden.

Testserie I (0,25 %)			Testserie II (1 %)			Testserie III (4 %)			Testserie IV (16 %)		
N_S	N_T	A	N_S	N_T	A	N_S	N_T	A	N_S	N_T	A
16	16	175	52	52	573	180	180	1.985	610	610	6.710

Tabelle 3: Zugriffsaufwand Ansatz 2 (in Tausend, gerundet)

Gegenüber dem ersten Ansatz kommt diese Vorgehensweise mit deutlich weniger Suchoperationen aus. Dies ist dadurch bedingt, daß mehrere Linienzüge in einer Datenseite gespeichert werden und durch einen Zugriff eingelesen werden. Der Gewinn hängt nur unwesentlich von der Größe der Bereichsanfragen ab. Da es in den Testdaten kaum Datensätze gibt, deren Größe die Seitengröße übersteigt, ist N_T praktisch genauso groß wie N_S.

Bei der Szenen-Organisation sind die Ergebnisse wesentlich von der durchschnittlichen Größe der Szenen abhängig. Während beim ersten Ansatz die exakte Darstellung nur dann eingelesen wird, wenn sie aufgrund der Anfrage benötigt wird, werden beim zweiten Ansatz am Rand von Anfragen unweigerlich auch Exaktrepräsentationen eingelesen, die die Anfrageregion nicht berühren *(Fehltreffer)*. Sehr große Szenen kommen mit sehr wenig Suchoperationen aus, benötigen aber dafür um so mehr Operationen zum Transfer vom Sekundärspeicher in den Hauptspeicher, da die Zahl der Fehltreffer mit zunehmender Szenengröße steigt. Je kleiner die Szenen werden, desto mehr steigt der Aufwand zum Suchen und desto geringer wird der Übertragungsaufwand. Um diesen Effekt näher zu untersuchen, haben wir unterschiedliche Szenengrößen getestet. Die Ergebnisse sind in Tabelle 4 aufgeführt; die Einträge mit dem geringsten Zugriffsaufwand sind dunkel unterlegt.

Wie erwartet, sinkt N_S mit zunehmender Szenengröße, während N_T steigt, je größer die Szenen sind. Dabei bildet sich je nach Größe der Anfragebereiche ein Optimum beim Zugriffsaufwand, das etwa zwischen 25 und 100 KByte liegt. Je größer die Anfragen sind, desto größer ist die optimale Szenengröße. Allerdings ist die Abhängigkeit nicht sehr einschneidend: Während 64 der Faktor zwischen Testserie I und IV bezüglich der Größe der Anfragebereiche ist, beträgt der Faktor zwischen den entsprechenden optimalen Szenengrößen nur rund 4. Außerdem verläuft die Aufwandsfunktion im Bereich dieser Optima bei allen Testserien sehr flach, so daß mit rund 77 KByte eine für alle Testserien nahezu optimale Szenengröße bestimmt werden kann.

durchschnittl. Szenengröße (Byte)	Testserie I (0,25 %)			Testserie II (1 %)			Testserie III (4 %)			Testserie IV (16 %)		
	N_S	N_T	A	N_S	N_T	A	N_S	N_T	A	N_S	N_T	A
1.852.750	1,1	698	709	1,2	781	794	1,6	943	959	2,1	1.666	1.188
757.943	1,2	275	287	1,6	343	345	2,4	505	529	4,1	837	877
273.357	1,5	128	144	2,3	185	208	4,2	324	365	8,6	638	725
140.124	1,8	78	96	3,0	123	153	6,0	238	298	14,1	528	669
91.619	2,2	62	84	3,9	103	142	8,5	214	299	20,8	503	711
79.027	2,1	51	72	3,9	90	130	8,8	191	280	22,7	474	701
63.402	2,3	46	70	4,5	85	130	10,4	187	291	27,5	467	742
33.283	3,2	36	68	6,7	70	137	17,3	167	340	48,8	447	936
18.610	4,2	26	68	10,0	57	158	27,8	151	429	82,8	432	1.260
10.716	6,1	23	84	15,4	54	209	45,6	153	609	140	452	1.853
8.367	6,8	21	87	18,2	52	235	55,6	152	708	175	460	2.210

Tabelle 4: Zugriffsaufwand Szenen-Organisation (Ansatz 3) (in Tausend, gerundet)

Zusammenfassung

In Tabelle 5 ist der Zugriffsaufwand für alle drei untersuchten Speicherungsansätze dargestellt. Dabei ist der Aufwand auf das erste Verfahren auf 1 normiert; bei den beiden anderen Ansätzen ist der Faktor angegeben, um den sich der Aufwand vermindert hat. Die durchschnittliche Szenengröße für Ansatz 3 beträgt 79.027 Byte.

Ansatz	Verbesserung des Zugriffsaufwands A (Faktoren)			
	I (0,25 %)	II (1 %)	III (4 %)	IV (16 %)
1: Geometrie außerhalb der Datenseite	1,0	1,0	1,0	1,0
2: Geometrie innerhalb der Datenseite	11,9	13,7	14,9	15,5
3: Szenen-Organisation	28,9	60,5	105,7	148,5

Tabelle 5: Verbesserung des Zugriffsaufwandes der Ansätze 2 und 3 im Vergleich zu Ansatz 1

Die Ergebnisse lassen sich wie folgt vergleichen und zusammenfassen:

- Der Ansatz, die Exaktdarstellung innerhalb der Datenseiten zu speichern, ist in unseren Tests dem ersten Ansatz, die exakte Repräsentation auszulagern, um einen Faktor von rund 12 bis 15 überlegen. Die Größe der Anfragebereiche spielt dabei nur eine untergeordnete Rolle.

  Bei diesem Ergebnis ist allerdings ein wichtiger Aspekt zu berücksichtigen: Die untersuchten Geo-Objekte sind im Vergleich zu der Größe der Datenseiten relativ klein. In realen Karten treten häufig wesentlich umfangreichere Objekte auf (vgl. Abschnitt 2.2). Das heißt, daß bei solchen Daten die

Objekte zwangsläufig ausgelagert werden müssen. In Folge wird dieser Ansatz sich bei einem größeren Anteil umfangreicher Objekte stark an das Leistungsverhalten des ersten Verfahrens annähern.

- Eindeutiger Gewinner dieses Vergleiches ist die Szenen-Organisation. Bereits bei kleinen Anfragebereichen tritt ein erheblicher Leistungsgewinn ein. So ist die Szenen-Organisation um einen Faktor von knapp 30 besser als der Auslagerungsansatz. Bei größeren Anfragebereichen steigt dieser Faktor sogar auf 106 und 148.

  Eine weitere wichtige Beobachtung ist, daß eine nahezu optimale Szenengröße weitgehend unabhängig von der Größe der Anfragebereiche ist. Damit ist der Deckungsbereich für eine Implementierung mit einer festgelegten Szenengröße sehr groß.

  Außerdem kann durch den flachen Verlauf der Aufwandfunktion ein starker Leistungsgewinn auch dann erzielt werden, wenn durch Einfügen oder Löschen von Geo-Objekten die durchschnittliche Szenengröße während der Laufzeit schwankt.

6 Zusammenfassung und Ausblick

Wir haben eine *Speicher- und Zugriffsarchitektur für geographische Datenbanksysteme* vorgeschlagen. Dazu wurden eine Reihe von prinzipiellen Konzepten und Techniken zur effizienten Anfragebearbeitung in Geo-Datenbanken in eine Gesamtarchitektur eingebettet.

Der *R*-Baum* bildet die Basis-Komponente unserer Geo-Architektur. Er organisiert die Daten auf dem Sekundärspeicher und ermöglicht eine effiziente räumliche Indizierung. So kann der Suchbereich von räumlichen Anfragen rasch eingeschränkt werden. Die nächste Stufe in der Architektur bilden die *Objektapproximationen*. Sie unterstützen eine einfache Vorauswahl, ob ein Geo-Objekt eine Anfrage erfüllt oder nicht. Gegenüber dem üblicherweise verwendeten minimal umgebenden Rechteck hat das Fünfeck eine sehr gute Approximationsgüte. Die exakte Geometrierepräsentation eines Objektes wird durch den *TR*-Baum* verwaltet. Er strukturiert das Geo-Objekt in *Zerlegungskomponenten* und organisiert diese räumlich. Damit kann die Lokalität von Anfragen ausgenutzt und auf die entscheidenden Teilkomponenten direkt zugegriffen werden. Aufgrund der einfachen geometrischen Teilobjekte entfallen aufwendige Tests mit dem Gesamtobjekt. Thematische Anfragen werden durch entsprechende *Sekundärindizes für thematische Attribute* unterstützt, die über eine Link-Tabelle mit dem räumlichen Primärindex verbunden sind.

Die beschriebene Architektur hat insbesondere den räumlich-selektiven Objektzugriff und kleinräumige Bereichsanfragen unterstützt. Für einen effizienten mengenorientierten Zugriff großräumiger Bereichsanfragen haben wir die *Szenen-Organisation* in die Geo-Architektur integriert. Hierbei werden größere Teile der Daten in physisch zusammenhängenden, aber nur objektweise geordneten Speicherbereichen zusammengefaßt. Diese Szenen werden über den primären R*-Baum organisiert. Das Leistungsverhalten der Szenen-Organisation haben wir näher untersucht. Für großräumige Bereichanfragen konnten massive Leistungssteigerungen um einen Faktor bis zu 150 gegenüber anderen Speicherungsansätzen erzielt werden.

Für unsere zukünftige Arbeit planen wir die Integration unserer Geo-Architektur in existierende erweiterbare Datenbanksysteme für geometrische Anwendungen. Vielversprechende Kandidaten für diese Idee sind DASDBS, GRAL und POSTGRES. Weiterhin ist die Entwicklung einer *parallelen Geo-Architektur* eine herausfordernde Aufgabe für zukünftige Foschungsaktivitäten. Diese zielt in zwei Richtungen. Durch Nutzung von Multiprozessor-Systemen besteht die Möglichkeit, die Anfragebearbeitung zu parallelisieren. Insbesondere der Zerlegungsansatz dürfte hier entscheidende Leistungssteigerungen erlauben. Das große Datenvolumen andererseits macht den Einsatz von Multidisk-Systemen sinnvoll. Eine zentrale Frage dabei ist die Bestimmung einer geeigneten Verteilung der Geo-Daten auf die verschiedenen Platteneinheiten.

Die Übertragung der vorgestellten Konzepte auf *dreidimensionale Geo-Objekte* ist ein weiteres künftiges Arbeitsfeld. Eine Anwendungsmöglichkeit liegt beispielsweise im Bereich der Bioinformatik. Erster Schritt in diese Richtung ist die Entwicklung und Implementierung dreidimensionaler Approximations- und Zerlegungsverfahren.

Literatur

[Arn 90] Arnold F.: *'GIS Geo-Informations Systeme',* Technischer Bericht, Bundesforschungsanstalt für Naturschutz und Landschaftsökologie, 1990

[Bar 88] Bartelme N.: *'GIS Technologie: Geoinformationssysteme, Landinformationssysteme und ihre Grundlagen',* Springer, 1988

[BKS 93a] Brinkhoff T., Kriegel H.-P., Schneider R.: *'Comparison of Approximations of Complex Objects used for Approximation-based Query Processing in Spatial Database Systems',* Proc. 9th Int. Conf. on Data Engineering, Vienna, Austria, 1993.

[BKS 93b] Brinkhoff T., Kriegel H.-P., Schneider R.: *'A New Technique to Support Large Window Queries in Spatial Database Systems'*, 1993, in Vorbereitung.

[BKS 93c] Brinkhoff T., Kriegel H.-P., Seeger B.: *'Efficient Processing of Spatial Joins Using R-trees'*, 1993, eingereicht zur Veröffentlichung.

[BKSS 90] Beckmann N., Kriegel H.-P., Schneider R., Seeger B.: *'The R*-tree: An Efficient and Robust Access Method for Points and Rectangles',* Proc. ACM SIGMOD Int. Conf. on Management of Data, Atlantic City, NJ., 1990, pp. 322-331.

[Bur 86] Burrough P.A.: *'Principles of Geographical Information Systems for Land Resources Assessment',* Oxford University Press, 1986.

[Bur 89] Bureau of the Census: *'TIGER/Line Percensus Files, 1990 Technical Documentation'*, Washington, DC., 1989.

[CDRS 86] Carey M. J., DeWitt D. J., Richardson J. E., Shekita E. J.: *'Object and File Management in the EXODUS Extensible Database System',* Proc. 12th Int. Conf. on Very Large Data Bases, Kyoto, Japan, 1986, pp. 91-100.

[Cra 90] Crain I.K.: *'Extremely Large Spatial Information Systems - A Quantitative Perspective',* Proc. 4th Int. Symp. on Spatial Data Handling, Zurich, Switzerland, 1990, pp. 632-641.

[Fra 91] Frank, A.U.: *'Properties of Geographic Data'*, Proc. 2nd Symp. on Large Spatial Databases, Zurich, Switzerland, 1991, in: Lecture Notes in Computer Science, Vol. 525, Springer, 1991, pp. 225-234.

[GC 87] Gorny A.J., Carter R.: *'World Data Bank II: General Users Guide',* Technical report, U.S. Central Intelligence Agency, Washington, 1987.

[Gue 89] Güting R. H.: *'Gral: an extensible relational database system for geografic applications',* Proc. 15th Int. Conf. on Very Large Data Bases, Amsterdam, Netherland, 1989, pp. 33-44.

[Gut 84] Guttman A.: *'R-trees: A Dynamic Index Structure for Spatial Searching',* Proc. ACM SIGMOD Int. Conf. on Management of Data, Boston, MA., 1984, pp. 47-57.

[HS 92] Hoel E.G., Samet H.: *'A Qualitative Comparison Study of Data Structures for Large Line Segment Databases',* Proc. SIGMOD Conf., San Diego, CA., 1992, pp 205-214.

[HSW 88] Hutflesz A., Six H.-W., Widmayer P.: *'Globally Order Preserving Multidimensional Linear Hashing'*, Proc. 4th Int. Conf. on Data Engineering, Los Angeles, CA., 1988, pp. 572-579.

[HWZ 91] Hutflesz A., Widmayer P., Zimmermann C.: *'Global Order Makes Spatial Access Faster'*, Int. Workshop on Database Management Systems for Geographical Applications, Capri, Italy, 1991, in: Geographic Database Management Systems, Springer, 1992, pp. 161-176.

[KBS 91] Kriegel H.-P., Brinkhoff T., Schneider R.: *'An Efficient Map Overlay Algorithm based on Spatial Access Methods and Computational Geometry',* Int. Workshop on Database Management Systems for Geographical Applications, Capri, Italy, 1991, in: Geographic Database Management Systems, Springer, 1992, pp. 194-211.

[KHS 91] Kriegel H.-P., Horn H., Schiwietz M.: *'The Performance of Object Decomposition Techniques for Spatial Query Processing',* Proc. 2nd Symp. on Large Spatial Databases, Zurich, Switzerland, 1991, in: Lecture Notes in Computer Science, Vol. 525, Springer, 1991, pp. 257-276.

[Kri 91a] Kriegel H.-P., Heep P., Heep S., Schiwietz M., Schneider R.: *'An Access Method Based Query Processor for Spatial Database Systems',* Int. Workshop on Database Management Systems for Geographical Applications, Capri, Italy, 1991, in: Geographic Database Management Systems, Springer, 1992, pp. 273-292.

[Kri 91b] Kriegel H.-P., Heep P., Heep S., Schiwietz M., Schneider R.: *'A Flexible and Extensible Index Manager for Spatial Database Systems',* Proc. 2nd Int. Conf. on Database and Expert Systems Applications, Berlin, Germany, 1991, pp. 179-184.

[Ore 89] Orenstein J. A.: *'Redundancy in Spatial Databases',* Proc. ACM SIGMOD Int. Conf. on Management of Data, Portland, USA, 1989, pp. 294-305.

[PH 90] Paterson D., Hennessy J.: *'Computer Architecture: A Quantitative Approach'*, Morgan Kaufman, 1990.

[PS 88] Preparata F.P., Shamos M.I.: *'Computational Geometry',* Springer, 1988.

[Sam 90] Samet H.: *'The Design and Analysis of Spatial Data Structures',* Addison Wesley, 1990.

[Sch 92] Schneider R.: *'Eine Speicher- und Zugriffsarchitektur für Geo-Datenbanken',* Dissertation, Institut für Informatik, Universität München, 1992, in Vorbereitung.

[SK 91] Schneider R., Kriegel H.-P.: *'The TR*-tree: A New Representation of Polygonal Objects Supporting Spatial Queries and Operations',* Proc. 7th Workshop on Computational Geometry, Bern, Switzerland, 1991, in: Lecture Notes in Computer Science, Vol. 553, Springer, 1991, pp. 249-264.

[SV 89] Scholl M., Voisard A.: *'Thematic Map Modelling',* Proc. 1st Symp. on the Design and Implementation of Large Spatial Databases, Santa Barbara, CA., 1989, in: Lecture Notes in Computer Science, Vol. 409, Springer, 1990, pp. 167-190.

[SR 86] Stonebraker M., Rowe L.: *'The Design of POSTGRES',* Proc. ACM SIGMOD Conf. on Management of Data, Washinton D.C., 1986.

[SW 86] Schek H.-J., Waterfeld W.: *'A Database Kernel System for Geoscientific Applications',* Proc. 2nd Int. Symp. on Spatial Data Handling, Seattle, Washington, 1986, pp. 273-288.

[Wei 89] Weikum G.: *'Set-Oriented Disk Access to Large Complex Objects',* Proc. 5th Int. Conf. on Data Engineering, Los Angeles, CA., 1989, pp. 426-433.

[Wid 91] Widmayer P.: *'Datenstrukturen für Geodatenbanken',* Entwicklungstendenzen bei Datenbanksystemen, Oldenbourg Verlag, 1991, pp. 317-361.