6. Hochdimensionale Zugriffsstrukturen

- Motivation
  - Verwaltung von Merkmalsvektoren komplexer Objekte (z. B. Farbhistogramme von Bildern)
  - Suche nach ähnlichen Objekten
- Probleme in hochdimensionalen Datenräumen ($[0,1]^d$)
  - Leere Bereichsanfragen:
    Ein Datensatz qualifiziert sich für eine Bereichsanfrage, die in jeder Dimension ein Intervall der Länge r abfragt, mit Wahrscheinlichkeit r^d.
  - Leere Kugelanfragen:
    Betrachte die Anfrage, die alle Datensätze sucht, die vom Mittelpunkt des Datenraums bzgl. der L_2-Norm den Abstand 0,5 besitzen. Das Volumen der Kugel ist für d gerade:

$$\frac{(\sqrt{\pi}/2)^d}{(d/2)!}$$

– Anzahl der k-dim. Randstücke im d-dimensionalen Datenraum

$$\binom{n}{k} 2^{d-k}$$

– Erwartungswert für den Abstand bei gleichverteilten Daten (unabhängige Dimensionen)

$$\frac{E(min\ dist(x,y))}{E(max\ dist(x,y))} \to 1 \quad (d \to \infty)$$

Probleme bei Indexstrukturen

R-Bäume

- Überlappungsgrad nimmt mit der Anzahl der Dimensionen zu
  - Aufspalten der Seite zu ineffektiv
- Speicheraufwand für einen Eintrag ist 2d (linear in der Anzahl der Dimensionen)
  - kleiner Fanout und großer Index
- Anfragen benötigen Zugriff auf nahezu alle Seiten
- Auswertung des Suchprädikats teuer

Wann sollte man überhaupt einen Index benutzen?

- Vergleich Organisation mit einem Index und einer Listenorganisation, in der alle Indexeinträge physisch geclustert vorliegen.
  - Zugriffskosten pro Blatt im Index (100% gefüllt): N Seek + N Transfer
  - Zugriff der Liste: 1 Seek + N Transfer
- Beträgt das Verhältnis Seek zu Transfer 10, so folgt, dass
  - Der Zugriff über den Index nur dann lohnt, wenn weniger als 10% der Seiten benötigt werden.

6.1 Erweiterungen des R-Baums

Ziel

- Entwurf eines R-Baums zur effizienten Verwaltung hochdimensionaler Daten

Ideen

- Verwendung von minimal umgebenden Kreisen
- Anpassung der Algorithmen zum Einfügen
  - Vermeidung von Überlappung
  - Wahl der Splitachse
  - Zuordnung eines Punkts zu einem Teilbaum

SR-Baum

- Katayama & Satoh, SIGMOD 1997
- zusätzlich zu Rechtecken: Verwendung von minimal umgebende Kreisen
  - Zentrum eines Kreis ist der Schwerpunkt der dazugehörigen Daten
  - Schwerpunkt läßt sich iterativ berechnen:

$$S_n = \frac{n-1}{n} S_{n-1} + \frac{1}{n} R_n$$

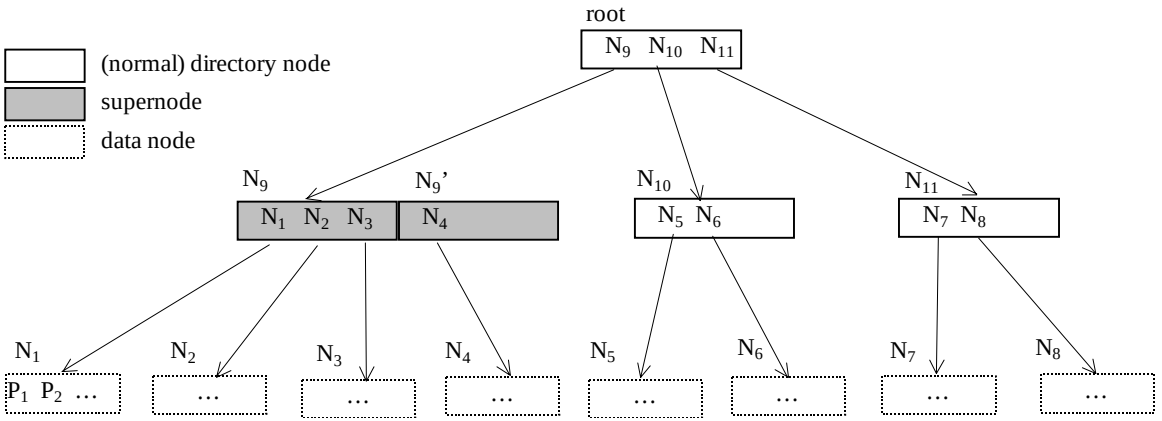
- Einfügealgorithmus (ChooseSubtree(Node, R))
  - Ordne den Datensatz R dem Teilbaum mit dem nächsten Schwerpunkt zu.
- Wahl der Splitdimension:
  - Bestimme die Dimension mit der höchsten Varianz.
- Experimente
  - Überlegenheit gegenüber gewöhnlichen R-Bäumen
  - Gleichzeitige Verwendung von Rechtecken und Kreisen zahlt sich aus!

X-Baum

- Berchthold, Keim, Kriegel, VLDB 1996

Wesentliche Ideen

- Statt einen “schlechten” Split einer Indexseite auszuführen, verzichtet man auf das Aufspalten und vergrößert die Seite (ähnlich zu elastischen Seiten).



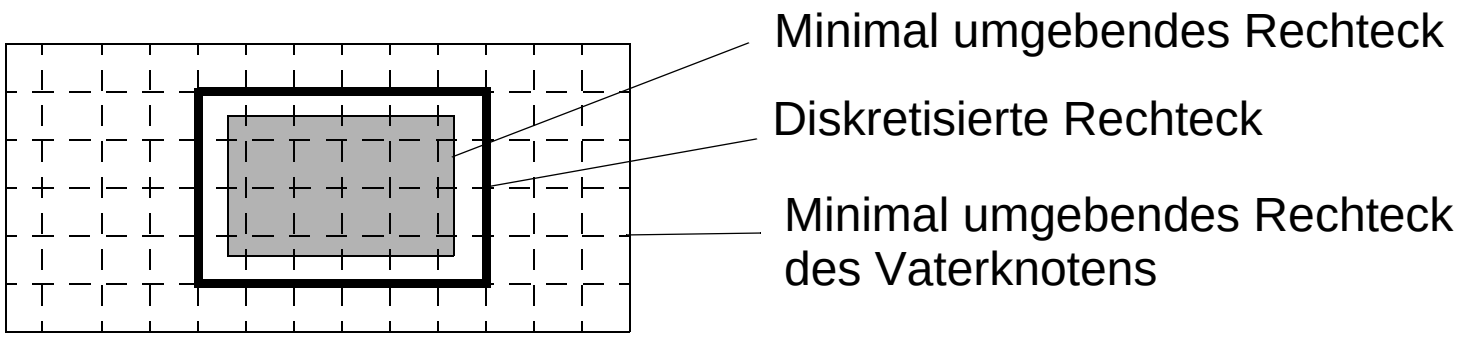
  - Parameter **MaxOverlap**: Ein Split wird dann akzeptiert, wenn der Overlap kleiner als MaxOverlap ist.

Eigenschaften

- Verbesserung der Speicherplatzausnutzung
- Spezialfälle:
  - keine großen Seiten: R-Baum
  - eine große Seite: sequentielle Listenstruktur
- Pufferorganisation muß die Verwaltung verschieden großer Seiten akzeptieren.
- Komplexere Sperrmechanismen bei gleichzeitig ablaufenden Operationen
- Experimente
  - Erhebliche Leistungsverbesserung im Vergleich zum R*-Baum

A-Baum

- Sakurai et al, VLDB 2000
- Experimentelle Beobachtung
  - Bei Nachbaranfragen im hochdimensionalen Raum muß auf sehr viele Indexknoten und nur wenige Datenknoten zugegriffen werden.
- Wesentliche Idee: Erhöhung des Verzweigungsgrads der Indexknoten
  - Verwendung von umschließenden Rechtecken, die nicht notwendigerweise minimal sind.
  - Indexeinträge enthalten nur ein **diskretisiertes Rechteck**, welches das MUR



    enthält.
  - MUR eines Knotens wird im Knoten (und nicht im Vater) abgespeichert.

- Verwendung des Einfügealgorithmus des SR-Baums
- Experimenteller Leistungsvergleich
  - A-Baum ist besser als SR-Baum und andere Techniken.

6.2 Dimensionsreduktion

- Hauptachsentransformation
  - Betrachte Covarianzmatrix Cov
  - Berechne Diagonalisierung der Matrix Cov = $P\ D\ P^T$
- Verwende die Achsen der Hauptachsentransformation mit den größten Eigenwerten.
  - Abbildung aller Datensätze in einen niedrigdimensionalen Datenraum (verlustbehaftete Kompression)
- Berechne die k-nächsten Nachbarn im niedrigdimensionalen Datenraum
- Laufzeit der Transformation: $O(n*d^2)$
  - d: Dimension der Datensätze
  - n: #Datensätze
- Dynamische Berechnung der Hauptachsentransformation
  - auf Basis von aggregierten Daten (Schwerpunkte der Blätter)
  - siehe Kanth et al, SIGMOD 1999

6.3 Metrische Indexstrukturen

- Was ist zu tun, wenn die Daten nicht in einem Vektorraum liegen, sondern die Distanzfunktion nur die Eigenschaft einer Metrik erfüllt
  - Abbildung des metrischen Raums in einen Vektorraum (z. B. durch FastMap, Faloutsos et al, 1995)
  - Entwurf einer Indexstruktur zur Verwaltung von metrischen Daten

M-Baum (VLDB 1997)

- basiert auf dem Prinzip des R-Baums
  - balancierter Baum mit logarithmischer Höhe
- Datenobjekte liegen nur in den Blättern
- Indexeinträge liegen in den inneren Knoten
  - Referenzpunkt P, Radius r, Referenz auf Teilbaum, Abstand a zum Referenzpunkt des Vaterknotens
  - Alle Daten im Teilbaum liegen in der Kugel mit Mittelpunkt P und Radius r

Bestimmung des Einfügepfads(Knoten K, Datensatz R)

Berechne zunächst die Menge aller Einträge in deren Kugel R liegt.

Falls diese Menge nicht leer ist

Berechne den Eintrag dessen Referenzpunkt minimalen Abstand zu R hat.

Ansonsten

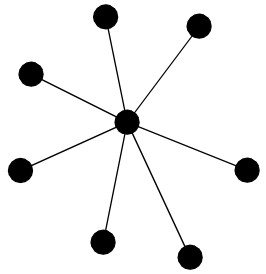
Berechne den Eintrag dessen Kugel minimalen Abstand zu R hat.

Aufspalten einer Seite(Knoten K)

- Berechne zwei neue Referenzpunkte P_1 und P_2. Verschiedene Strategien, z. B.
  - Das Paar (P_1,P_2), bei dem die Summe der Kugelradien minimal ist (Alle Daten müssen von den Kugeln überdeckt sein).
- Ordne die Daten aus K den Referenzpunkten zu, so dass jeder der Referenzpunkte mindestens m Datenobjekte (und höchstens M-m+1) Datenobjekte besitzt. Verschiedene Strategien, z. B.:
  - Ordne ein Datenobjekt zum nächsten Referenzpunkt zu.
- Speichere jede der zwei Partitionen in eine neue Seite und propagiere die entsprechenden Indexeinträge in den Vaterknoten.
  - Berechnung einer umgebenden Kugel (minimal ?)

Alternativer Ansatz

- Berechne den minimal spannenden Baum des Abstandsgraphen der Punkte
- Entferne die kleinsten Kanten des Graphen bis dieser in zwei Teile zerfällt.
- Problemfall



Ein Punkt hat minimalen Abstand zu allen anderen Punkten im metrischen Raum.

Kugelanfragen

- Suche alle Punkte, die innerhalb der Distanz **q** zu einem Punkt **Q** liegen.
- Top-Down Ansatz analog zu einer Bereichsanfrage im R-Baum
  - Bestimme für jeden Eintrag E in einem Knoten, ob

$$dist(Q,E.P) \leq q + E.r$$

Vermeidung teurer Distanzvergleiche durch Anwendung der Dreiecksungleichung

- d_{Q-V} = dist(Q, Vater)
- d_{E-V} = dist(E,Vater) = E.a
- Es gilt also $d_{Q-V} - d_{E-V} \leq d(E,O)$ und
- $d_{E-V} - d_{Q-V} \leq d(E,O)$
- Gilt $|d_{Q-V} - d_{E-V}| > q$, kann sich also E nicht mehr qualifizieren.