Kapitel 4 Hochdimensionale Räume

- Anfrageleistung von Indexstrukturen verschlechtert sich mit zunehmender Dimension
  "Curse of Dimensionality"
  → Häufig haben scanbasierte Methoden bessere Anfrage-Performanz als z.B. R*-Bäume
- In diesem Kapitel:
  — Ermittlung der Ursachen mit Hilfe eines Kostenmodells
  — Optimierung der Indexstrukturen sowie der Algorithmen zur Anfragebearbeitung
  — Entwicklung neuer Indexstrukturen, die besonders an die Problemstellung hochdimensionaler Datenräume angepaßt sind

4.1 Ein Kostenmodell für R-Bäume

[Berchtold S., Böhm C., Keim D., Kriegel H.-P.: *A Cost Model for Nearest Neighbor Search in High-Dimensional Data Spaces*, PODS 1997]

Ziel: Schätzung der zu erwartenden Anzahl der Seitenzugriffe bei der Anfragebearbeitung

- für R-Bäume und verwandte Indexstrukturen
- verschiedene Anfragetypen:
  — Bereichsanfragen
  — nächste-Nachbar-Anfragen sowie *k*-nächste-Nachbar-Anfragen
  — verschiedene Metriken, hier nur euklidische und Maximums-Metrik
  (Ellipsoid-Anfragen sind schwierig zu modellieren)
- Einschränkungen:
  — idealisierte Indexstruktur:
  überlappungsfrei
  Seitenregionen sind annähernd quadratisch ("so quadratisch wie möglich")
  — Datenraum ist der Einheits-Hypercube $[0..1]^d$
  — zunächst: Punkte und Anfragen folgen einer unabhängigen Gleichverteilung
  — später: Beschreibung der Datenverteilung durch *fraktale Dimension*
  (genauere Darstellungsmethoden der Datenverteilung wie z.B. Histogramme sind im Hochdimensionalen schwierig)

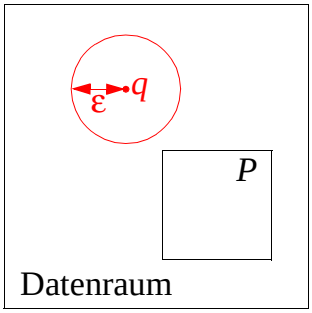
4.1.1 Bereichsanfragen

- Bekannt:
  — Radius ε der Anfrage

— Ausdehung der Seitenregion
→ später wird beides geschätzt werden

- Unbekannt:
  — relative Lage von Seitenregion und Zentrum der Anfrage (*Anfragepunkt*)
  — beides wird als unabhängig gleichverteilt angenommen, d.h. jede Position von Anfragepunkt und Seitenregion ist gleich wahrscheinlich

- Gesucht:
  — Wahrscheinlichkeit, mit der die Query q auf die Seite P zugreift (Zugriffswahrscheinlichkeit)
  — entspricht Wahrscheinlichkeit, mit der sich der Kreis mit der Seitenregion schneidet



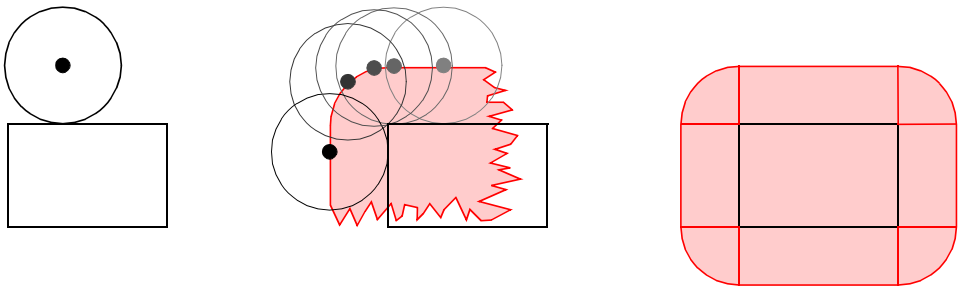
- Das Problem ist leicht zu lösen, wenn z.B. die Anfrage punktförmig ist:

$$\text{Zugriffswahrscheinlichkeit} = \frac{\text{Volumen Seitenregion}}{\text{Volumen Datenraum}}$$

ebenso bei punktförmiger Seitenregion
→ Wahrscheinlichkeitsrechnung (Kombinatorik) benötigt *punktförmige* Ereignisse

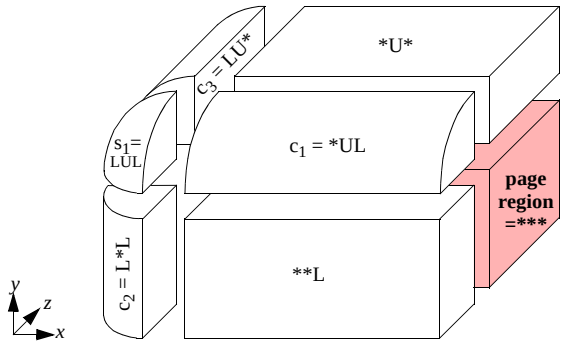
- Trick um punktförmige Ereignisse zu erhalten:

Transformiere Bereichsanfrage in eine äquivalente Punktanfrage:
— Verkleinere die Bereichsanfrage zum Punkt
— Vergrößere in gleichem Maß die Seitenregion
— so daß die neue Punktanfrage auf die vergrößerte Seitenregion zugreift gdw. die Bereichsanfrage auf die ursprüngliche Seitenregion zugreift



- Das entstehende Objekt heißt *Minkowski-Summe* von Anfrage- und Seitenregion:
  — ursprüngliches Rechteck
  — an jeder Kante ist ein Quader der Breite ε angehängt
  — an jeder Ecke ist ein Viertelkreis mit Radius ε angehängt

- Im dreidimensionalen Fall (Grafik unvollständig):



  - ursprünglicher Quader: 3-dimensional rechteckig, 0-dimensional rund
  - an jeder Oberfläche: Quader mit Grundfläche wie Oberfläche, Dicke ε
  - an jeder Kante: $\frac{1}{4}$ Zylinder: Länge wie Kante, Grundfläche ist Kreis mit Radius ε
  - an jeder Ecke: $\frac{1}{8}$ Kugel mit Radius ε

- Ein *d*-dimensionaler *Hypercube* hat neben Ecken (0-dimensional), Kanten (1d) und Flächen (2d) auch noch 3-dimensionale, 4-dimensionale .... $(d-1)$-dimensionale "Oberflächen"-Segmente (engl. *faces*)
  - an jedem *i*-dimensionalen Segment hängt ein Objekt ("*Hyperzylinder*"), das in *i* Dimensionen würfelförmig ist und in $(d-i)$ Dimensionen kugelförmig ist (genau genommen der 2^{d-i}-te Teil einer solchen Hyperkugel)

- Wie viele Ecken, Flächen, ... *i*-dimensionale Segmente hat ein *d*-dimensionaler Würfel? Hierzu führen wir eine Notation ein, die, jedem Oberflächensegment (incl. dem urspr. Hypercube) ein *d*-Tupel über dem Alphabet aus den drei Symbolen L, U, und * zuordnet. Hierbei bedeutet:
  - L die untere Grenze (lower bound) in einer Dimension
  - U die obere Grenze (upper bound) in einer Dimension
  - * den gesamten Bereich zwischen der unteren Grenze und der oberen Grenze

- Beispiel:
  - Enthält ein Tupel kein *, so bezeichnet es eine *Ecke* (z.B. LUL die linke, obere, vordere, Ecke des dreidimensionalen Würfels)
  - Enthält ein Tupel genau ein *, so bezeichnet es eine *Kante* (z.B. LU* die Kante links oben, von vorne nach hinten)
  - Ein Tupel mit *i* Sternchen bezeichnet ein *i*-dimensionale Oberflächensegment
  - Das Tupel, das *nur* aus *d* Sternchen besteht, bezeichnet den originalen Hypercube

- Anzahl der Tupel mit *i* Sternchen:

$$\binom{d}{i} \cdot 2^{d-i}$$

  - $\binom{d}{i}$ → verteile *i* Sternchen über *d* Positionen
  - 2^{d-i} → fülle die verbleibenden *d*–*i* Positionen mit L oder U

- Gesamte Formel für das Volumen der Minkowski-Summe aus Hypercube mit Seitenlänge a und Kugel mit Radius ε:

$$V_{\text{Mink}}(a, \varepsilon) = \sum_{0 \le i \le d} \binom{d}{i} \cdot 2^{d-i} \cdot a^i \cdot \frac{V_{(d-i)\text{-dim.Kugel}}(\varepsilon)}{2^{d-i}} = \sum_{0 \le i \le d} \binom{d}{i} \cdot a^i \cdot V_{(d-i)\text{-dim.Kugel}}(\varepsilon)$$

  Das Volumen einer j-dimensionalen Kugel läßt sich folgendermaßen ermitteln:

  $V_{j\text{-dim.Kugel}} = \frac{\pi^{j/2} \cdot r^j}{\Gamma(j/2+1)}$, wobei Γ die Gamma-Funktion (Erweiterung der Fakultät in reelle Zahlen) darstellt, mit: $\Gamma(x+1) = x \cdot \Gamma(x)$ $\Gamma(1) = 1$ $\Gamma(1/2) = \sqrt{\pi}$

- Mit V_{Mink} kann die Zugriffswahrscheinlichkeit einer einzelnen bekannten Seite bereits ermittelt werden. Dies werden wir später bei einer Optimierungstechnik anwenden.
- Im allgemeinen interessieren die Kosten für den gesamten Index;
  Summation der Zugriffswahrscheinlichkeiten aller Seitenregionen zu teuer

Schätzung der Seitenlänge des Hypercube

Benutze eine durchschnittliche Seite anstatt der konkreten Seiten.
Für jede Indexebene i läßt sich die Anzahl der Seiten n_i ermitteln, sofern die durchschnittliche Speicherauslastung (su_{eff}) bekannt ist (aus Data Dictionary):

Sei $C_{\text{eff}} := C \cdot su_{\text{eff}}$ die effektive Kapazität der Seiten (durchschnittliche Anzahl in einer Seite gespeicherter Einträge), N die Gesamtzahl von Featurevektoren

- $n_0 := N/C_{\text{eff,data}}$ (Anzahl der Datenseiten)
- $n_i := n_{i-1}/C_{\text{eff,dir}}$ (Anzahl der Seiten auf Directory-Ebene i)

Annahmen:

- Seitenregionen haben Volumen $1/n_i$ (1 ist das Volumen des Datenraums $[0..1]^d$)
- Seitenregionen sind annähernd (hyper-) würfelförmig.

Schätzwert für die Kantenlänge a_i einer Seitenregion auf Indexebene i: $a_i = \sqrt[d]{1/n_i}$
Kosten durch Addition der Zugriffswahrscheinlichkeiten aller Seiten auf allen Ebenen:

$$\#\text{Zugriffe}(\varepsilon) = \sum_i n_i \cdot V_{\text{Mink}}(\sqrt[d]{1/n_i}, \varepsilon)$$

4.1.2 Nächste-Nachbar-Anfragen

Annahme:

Die Anfragen werden mit dem Prioritätsalgorithmus [HS 95] bearbeitet (s. Seite 138 *ff.*)
Dies ermöglicht Ausnutzung der Optimalität der Anfragebearbeitung
Die Performanz der anderen NN-Algorithmen hängt u.a. davon ab, welcher Pfad als erstes verfolgt wird und ist deswegen schwerer zu schätzen.

Konsequenz aus dem Optimalitätsbeweis (Lemma 3):

- Der Algorithmus lädt genau die Seiten, die die Nearest-Neighbor-Kugel schneiden
- Der Algorithmus ist äquivalent zu Algorithmus für Bereichsanfragen, wobei ε durch die Nearest-Neighbor-Distanz ersetzt wird.
- Es reicht also prinzipiell, die NN-Distanz zu schätzen.

Einfache Methode zur Schätzung der NN-Distanz

Das Volumen einer Kugel mit Radius ε multipliziert mit N entspricht dem Erwartungswert der eingeschlossenen Datenpunkte. Bestimme die Kugel so daß der Erwartungswert 1 (bzw. k bei k-Nearest Neighbor Queries) ist:

$$\varepsilon \approx V^{-1}_{d\text{-dim.Kugel}}(k/N) = \sqrt[d]{\frac{k \cdot \Gamma(d/2+1)}{N \cdot \pi^{d/2}}}$$

Probleme:

- Stochastisch Vorgehensweise nicht korrekt (Operation "*Bildung des Erwartungswerts*" ist nicht umkehrbar)
- Auch unter Idealbedingungen (gleichverteilte Daten, niedrigdimensional) nicht sehr exakt für kleines k
- Für großes $k > 10$ aber hinreichend genau

Stochastisch korrekte Ermittlung des Erwartungswerts:

— Ermittlung der Verteilungsfunktion der NN-Distanz
— Daraus: Wahrscheinlichkeitsdichtefunktion durch Differenzieren
— Daraus: Erwartungswert durch Integration

Verteilungsfunktion $P(r)$:

"Wie hoch ist Wahrscheinlichkeit, daß die NN-Distanz ≤ stochastische Variable r ist"

⇔

"Wahrscheinlichkeit, daß in einer Kugel mit Radius r mind. 1 Datenpunkt enthalten ist"

⇔

1 – "Wahrscheinlichkeit, daß *keiner* der N Datenpunkte im Volumen der Kugel liegt"

⇔

1 – "Wahrscheinlichkeit, daß *alle* Datenpunkte im Volumen außerhalb Kugel liegen"

⇔

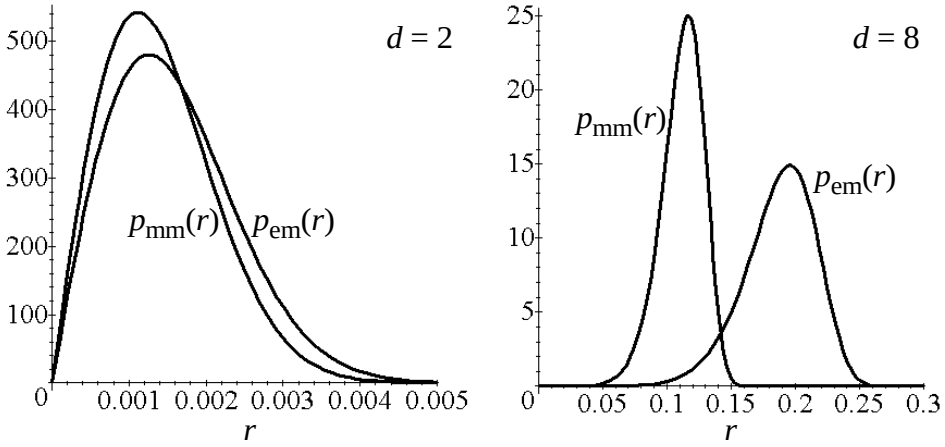
$$P(r) = 1 - (1 - V_{\text{Kugel}}(r))^N$$

Wahrscheinlichkeitsdichtefunktion:

$$p(r) = \frac{\partial P(r)}{\partial r} = \frac{d \cdot N}{r} \cdot \left(1 - \frac{\pi^{d/2}}{\Gamma(d/2+1)} r^d\right)^{N-1} \cdot \frac{\pi^{d/2}}{\Gamma(d/2+1)} r^d$$

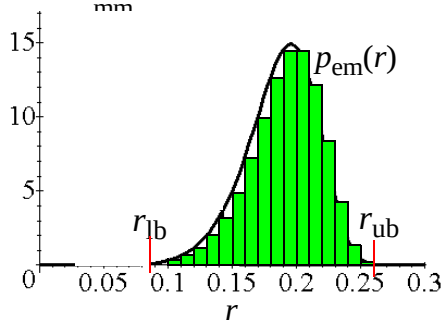
Maximumsmetrik: ähnlich.

Erwartungswert:



$$E[\varepsilon] = \int_{-\infty}^{\infty} r \cdot p(r) \partial r = \int_{0}^{\infty} r \cdot p(r) \partial r$$

- Analytisch schwierig zu integrieren
- Numerische Integration z.B. mittels Histogrammen:
  - links, rechts, oder mittig verankerte Rechtecke
  - Trapeze
  - Simpson's Regel
- Wichtig:
  Erst untere r_{lb} und obere r_{ub} Grenze des Integrationsbereichs ermitteln, etwa so daß
  - $P(r_{lb}) = 0{,}001$
  - $P(r_{ub}) = 0{,}999$



- Damit ergibt sich für den Erwartungswert:

$$E[\varepsilon] = \int_{r_{lb}}^{r_{ub}} r \cdot p(r) \partial r \approx \frac{r_{ub} - r_{lb}}{i_{max}} \cdot \sum_{0 \le i < i_{max}} \left(\frac{r_{ub} - r_{lb}}{i_{max}} \cdot i + r_{lb} \right) \cdot p\left(\frac{r_{ub} - r_{lb}}{i_{max}} \cdot i + r_{lb} \right)$$

- Um einen relativen Approximationsfehler von <1% zu erreichen sind lediglich $i_{max} = 5$ Rechtecke erforderlich.

Erwartungswert für die Anzahl der Seitenzugriffe

- Einfache Variante: E[ε] berechnen und dann #Zugriffe $= \sum_i n_i \cdot V_{\text{Mink}}(\sqrt[d]{1/n_i}, E[\varepsilon])$
- Genauere Methode:

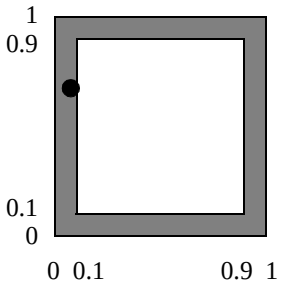
  $\text{E}[\#\text{Zugriffe}] = \int_0^\infty \#\text{Zugriffe}(r) \cdot p(r) \partial r$

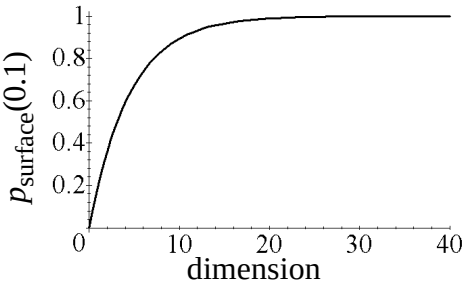
  Numerische Auswertung wie bei E[ε].

4.1.3 Effekte in hochdimensionalen Räumen

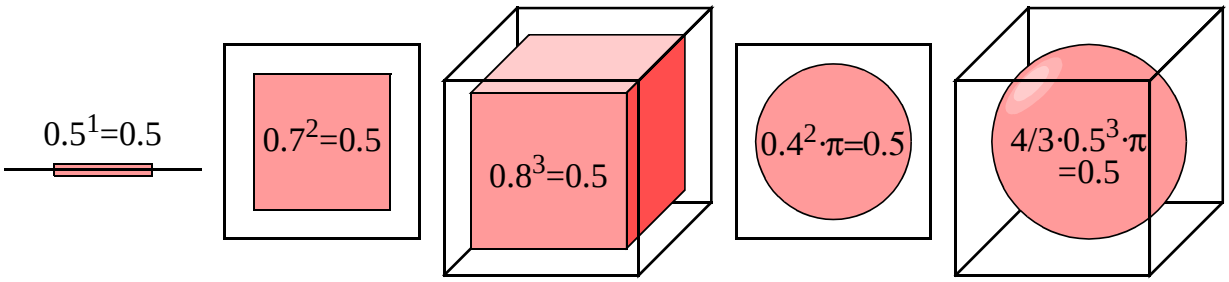
- Bisheriges Modell geht davon aus, daß sich sowohl
  — die NN-Kugel als auch
  — die Minkowski-Summe
  vollständig im Datenraum befinden. Bei niedrigdimensionalen Anwendungen ist dies auch (annähernd) korrekt.
- Dagegen wird im Hochdimensionalen die Oberfläche der Punktemenge so groß, daß nahezu jeder Punkt "außen" liegt. Es gibt so viele verschiedene "Richtungen", daß kaum Punkte "innen" liegen können
  → konvexe Hülle enthält nahezu alle Punkte

- Für den Einheitswürfel $[0..1]^d$ als Datenraum bedeutet dies etwa:
  Die Wahrscheinlichkeit, mit der ein beliebiger Punkt höchstens um δ von der Begrenzung des Datenraums entfernt ist nimmt mit steigender Dimension stark zu:
  $p_{\text{surface}}(\delta) = 1 - (1 - 2 \cdot \delta)^d$





- Folgende Größen sind nicht mehr *klein* gegenüber der Seitenlänge des Datenraums (=1)
  — typische Radien von Bereichsanfragen
  — die Distanz zum (*k*-ten) nächsten Nachbarn
  — die Seitenlänge der MBRs oder anderer Seitenregionen
  Grund: Die Länge ist proportional zur *d*-ten Wurzel des Volumens. Sie strebt gegen 1 für große *d*.



Folgende Größen sind in etwa proportional zum Volumen:

— die (k-)NN-Kugel: $V \approx k/N$ (normierter Datenraum!)
— die Seitenregion: $V \approx C_{\text{eff}}/N$
— in gewissem Sinn auch die Kugel bei Range-Queries, denn sinnvolle Anfrageergebnisse ergeben sich nur, wenn die Ergebnismenge nichttrivial ist (d.h. weder leer noch die gesamte Datenbank)

Konsequenzen für die Anfragekugeln:

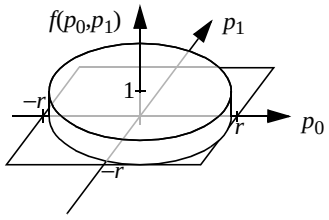
— häufig liegt der Kugeldurchmesser nahe bei 1 oder überschreitet 1 sogar erheblich
— die Kugel ragt also an mehreren Stellen über die Grenzen des Datenraums hinaus
— die Teile der Kugel, die außerhalb des Datenraums liegen, können zum Anfrageergebnis nicht beitragen (Clipping)

— dieses Volumen darf man also in Kostenmodellen nicht werten:
  - weder bei der Ermittlung der NN-Distanz
  - noch bei der Minkowski-Summe
— man benötigt eine um den Clipping-Effekt korrigierte Volumensfunktion
— weil die Kugeln ein bestimmtes Volumen aufweisen *müssen* (z.B. ca. k/N), werden die Kugelradien durch den Clipping-Effekt also *noch größer.*

Idee zur Ermittlung der korrigierten Volumensfunktion

- komplexes Raumintegral:
  gewöhnliches Kugelvolumen:

$$V(r) = \int_{-r}^{r} \dots \int_{-r}^{r} \begin{cases} 1 & \text{wenn } |P| \le r \\ 0 & \text{sonst} \end{cases} \partial p_0 \partial p_1 \dots \partial p_{d-1}$$



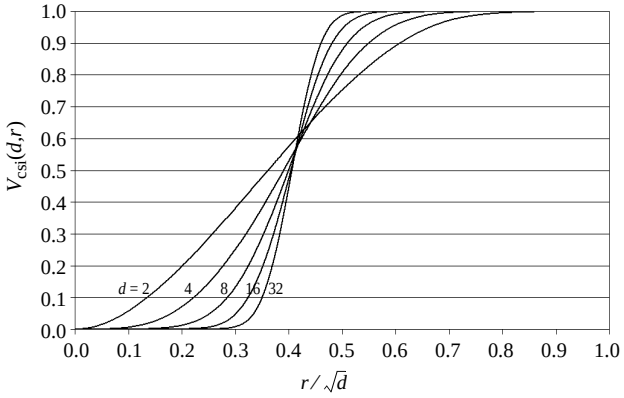
  Kugel um Q, geclippt an $[0..1]^d$:

$$V(r) = \int_{0}^{1} \dots \int_{0}^{1} \begin{cases} 1 & \text{wenn } |P-Q| \le r \\ 0 & \text{sonst} \end{cases} \partial p_0 \partial p_1 \dots \partial p_{d-1}$$

durchschnittliches Volumen um einen *beliebigen* Punkt Q (gleichverteilt)

$$V(r) = \int_0^1 \ldots \int_0^1 \int_0^1 \ldots \int_0^1 \begin{cases} 1 & \text{wenn } |P-Q| \le r \\ 0 & \text{sonst} \end{cases} \partial p_0 \partial p_1 \ldots \partial p_{d-1} \partial q_0 \partial q_1 \ldots \partial q_{d-1}$$

- Analytische Integration schwierig
  Trapezmethode etc. scheitert im Hochdimensionalen
- Montecarlo-Integration:
  Wähle z.B. n = 1.000.000 zufällige Punkt-Paare und schätze das Volumen anhand der relativen Häufigkeit des Ereignisses $|P-Q| \le r$



- Vorberechnung des Volumens für
  — alle vorkommenden Dimensionen $1 \le d \le d_{max}$
  — alle vorkommenden Radien in geeigneten diskreten Schritten $0 \le r_i \le r_{max} = \sqrt{d}$ mit $0 \le i \le i_{max}$; $r_i = r_{max} \cdot i / i_{max}$ und Speicherung in einem Array
  — die Vorberechnung der Volumensfunktion erfolgt zur Compilezeit der Programme, nicht erst, wenn das Kostenmodell ausgewertet wird

Konsequenzen für MURs

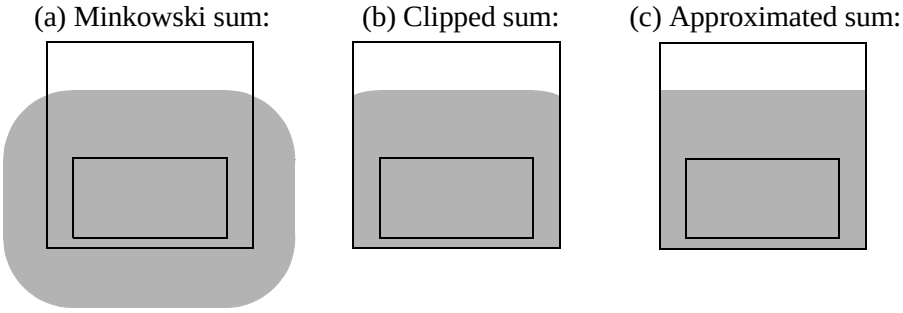
- Annahme würfelförmiger Seiten wird unrealistisch:
  Seiten entstehen durch Split-Operationen. Dabei wird der Datenraum entlang einer Dimension in 2 Teile aufgespalten, die gleich viele Punkte enthalten.
- In ausreichend hohen Dimensionen ist es unmöglich, daß jede Dimension gesplittet wird (bzw. daß eine Datenseite dadurch entsteht, daß in jeder Dimension etwas abgespaltet wird), denn ein Split pro Dimension führt zu 2^d Datenseiten:
  — $2^{20} \approx 1$ Million Seiten
  — $2^{30} \approx 1$ Milliarde Seiten usw.
- Typische Situation in hochdimensionalen Räumen
  — Datenseiten sind in einer Reihe von Dimensionen ungesplittet, d.h. [0..1]
  — in den restlichen Dimensionen 1 mal gesplittet, d.h. ungefähr [0..0.5] bzw. [0.5..1]
- Die Anzahl gesplitteter Dimensionen hängt von der Anzahl der Datenseiten ab:

$$d' = \log_2(N / C_{eff})$$

- Ist $d > d'$, dann ist das niedrigdimensionale Kostenmodell anzuwenden

Konsequenzen für die Minkowski-Summe

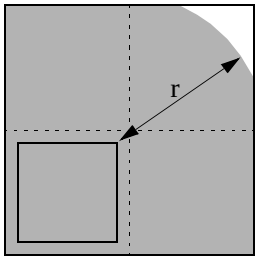
- Sehr große Teile der Minkowski-Summe ragen über den Datenraum hinaus und müssen geclippt werden
- Wird nicht geclippt, dann übersteigt das Volumen der Minkowski-Summe häufig das Volumen des Datenraums → Wahrscheinlichkeit > 1!



- Nur die gesplitteten Dimensionen der Seitenregionen werden durch die Minkowski-Summe vergrößert, und nur an einem Ende, nicht an beiden
- Bruchteile von Kugeln, die angefügt werden, unterliegen auch einer speziellen Clipping-Operation, sofern der Radius 1/2 überschreitet

- Hier wird ebenfalls eine spezielle Volumensfunktion benötigt: Zentrum in der linken unteren Ecke des Clippingraums

$$V(r) = \int_0^1 \ldots \int_0^1 \begin{cases} 1 & \text{wenn } |P| \le r \\ 0 & \text{sonst} \end{cases} \partial p_0 \partial p_1 \ldots \partial p_{d-1}$$



- Die Volumensfunktion wird wie vorher vorberechnet und tabelliert
- Für die Minkowski-Summe ergibt sich wieder eine Binomialformel:

$$V_{\text{Mink}}(r) = \sum_{0 \le i < d} \binom{d'}{i} \cdot \left(\frac{1}{2}\right)^{d'-i} \cdot V_{i\text{-dim. geclippte Kugel}}(r)$$

Auswirkung der hohen Dimension auf die Indexleistung

- Mit steigender Dimension steigt die Zugriffswahrscheinlichkeit deutlich an
- Komplexität exponentiell in der Anzahl der Dimensionen, bis ein Sättigungseffekt einsetzt, weil die Anzahl der Datenseiten, die gelesen werden können, beschränkt ist
- Im Sättigungsbereich: Suchperformanz ist nicht mehr logarithmisch sondern nur leicht sublinear; sequentieller Scan der Datenmenge ist häufig dem Index überlegen

4.1.4 Ungleichverteilung und Korrelation

Bisher wurde von Datenpunkten (und Anfragepunkten) ausgegangen, die einer unabhängigen Gleichverteilung folgen. Dies ist nicht sehr realistisch und führt zu Ungenauigkeiten im Modell.

Konzepte, die in Standard-Datenbanken (1-dimensional) zum Einsatz kommen, wie z.B. verschiedene Arten von Histogrammen oder parametrische Beschreibungen der Datenverteilung, lassen sich nicht auf den mehrdimensionalen Fall übertragen.

Bereichsanfragen reagieren sehr sensibel auf die Punktdichte, die im jeweiligen Bereich gilt, wenn geclusterte Daten vorliegen. Die Performanz von Bereichsanfragen ist deshalb meist nicht akurat vorherzusagen.

Im Ggs. dazu reagieren NN-Anfragen nicht sehr sensibel auf variierende Punktdichten, sofern diese

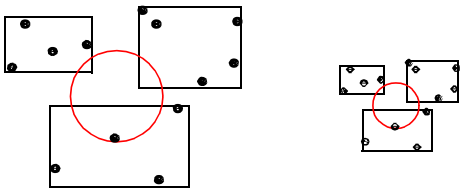
— unabhängig in den einzelnen Dimensionen sind und
— glatt verlaufen (d.h. die Punktdichte ändert sich nicht zu sprunghaft)

Anschauliche Begründung: Durch Erhöhung der Punktdichte verringern sich

— Volumen der NN-Kugel und
— Volumen der Seitenregion

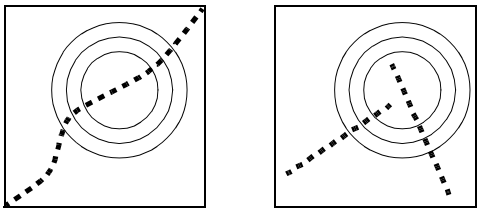
in gleichem Maß. Deshalb werden in beiden Fällen gleich viele Seiten zugegriffen.

Da sich nichts ändert, müssen solche Ungleichverteilungen nicht extra berücksichtigt werden.



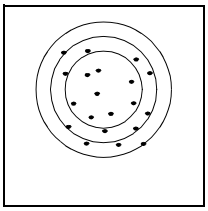
Korrelation

- Korrelierte Datenmengen verletzen Forderung nach Unabhängigkeit der Dimensionen
- Der Wert in einer Komponente des Vektors bestimmt also in gewissem Ausmaß den Wert einer anderen Komponente
- Nicht nur lineare Korrelationen, sondern auch komplexe Zusammenhänge und partielle Korrelationen wie z.B.

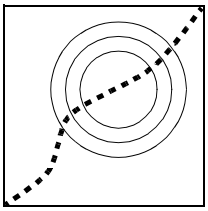


- Techniken zur Dimensionsreduktion wie z.B. PCA sind nur für (totale) lineare Korrelationen ausgelegt und können nur einen Teil der auftretenden Effekte erklären
- Allgemeineres Prinzip: Fraktale Dimension

- Die fraktale Dimension stellt einen (asymptotischen) Zusammenhang zwischen einem Volumen und der darin eingeschlossenen Anzahl von Punkten her:



unabhängige Verteilung:
Punkte proportional zu r^2
Fraktale Dimension $d_F = 2$

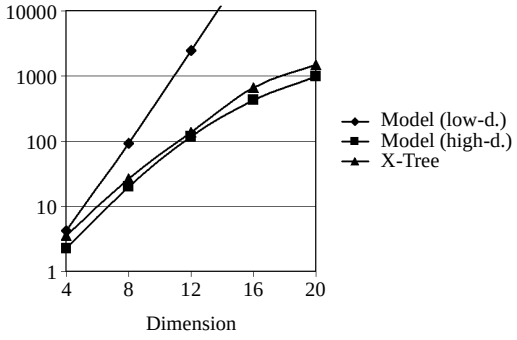


eindimensionale Korrelation:
Punkte proportional zu r
Fraktale Dimension $d_F = 1$

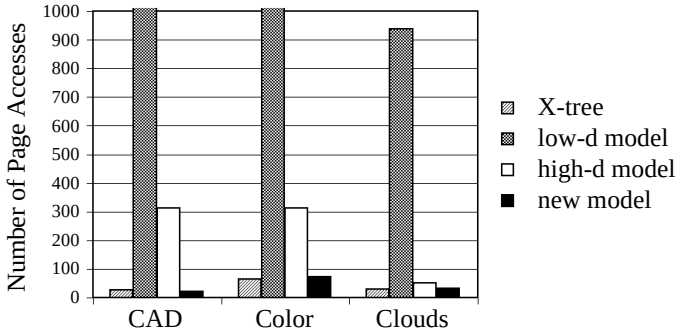
- Die fraktale Dimension "mißt" die Abhängigkeit zwischen Volumen und Anzahl eingeschlossener Punkte für die gesamte Datenmenge (im Durchschnitt)
- Es gibt verschiedene Definitionen von fraktalen Dimensionen
  — meist gitterbasierte Ansätze, was zu Problemen im Hochdimensionalen führt
- Im wesentlichen muß in den Kostenmodellen die Dimension des Datenraums durch die fraktale Dimension ersetzt werden.
- Ist die fraktale Dimension niedrig, dann ist auch das Kostenmodell für niedrigdimensionale Räume anzuwenden

Evaluation

Gleichverteilte Daten über Dimension:



Verschiedene Realdaten:

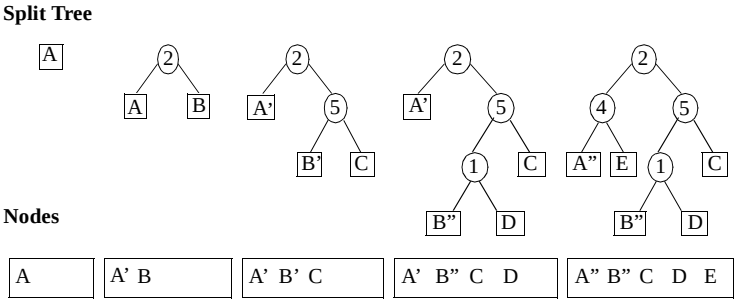


4.2 Indexstrukturen für hochdimensionale Räume

4.2.1 X-tree (eXtended node)

[BKK 96] Berchtold S., Keim D., Kriegel H.-P.: *The X-tree: An Index Structure for High-Dimensional Data*. VLDB 1996, 28-39.

- Weiterentwicklung des R*-Baums für hochdimensionale Räume
- Der Split-Algorithmus von R-Baum und R*-Baum führt zu hoher Überlappung der Directory-Seitenregionen bei hochdimensionalen Räumen
- Grund: Es gibt nur wenige (meist eine) geeignete Splitebene bei Split der Directoryseite
  — Seiten sind in vielen Dimensionen ungeteilt (Ausdehnung [0..1])
  — Benötigt wird eine Dimension, in der alle Kindseiten geteilt wurden
- Konzept hierfür: Split-Tree eines Directory-Knoten



- Idee des Split-Tree:
  — Jede neue Seite entsteht dadurch, daß eine alte Seite aufgeteilt wurde
  — Die Hierarchie von Split-Operationen läßt sich als binärer Baum darstellen, wobei in den inneren Knoten die Split-Dimension vermerkt ist
  — Ist d_i ein Vorgängerknoten eines Blatts X, dann wurde X in Dimension d_i geteilt
- Nur wenn in einem Level des Split-Tree alle Split-Ebenen identisch sind, kann Directoryknoten in dieser Dimension geteilt werden
  — Dies ist meist nur bei der Wurzel des Splitbaums der Fall
- Beispiel: Directoryknoten wird (fälschlicherweise) in Dimension 1 geteilt
  — Die Seite B" kommt in die 1. Nachfolgeseite
  — Die Seite D kommt in die 2. Nachfolgeseite
  — Die Seiten A', C und E sind in Dimension 1 nicht gesplittet. Werden sie einer der beiden Nachfolgeseiten zugeordnet, so überlappt diese die andere Nachfolgerseite vollständig
- Directoryknoten wird (richtig) in Dimension 2 geteilt
  — A' und E kommen in die erste Nachfolgeseite
  — B", C und D kommen in die 2. Nachfolgeseite
  — Die beiden Nachfolgeseiten sind überlappungsfrei
- Weiteres Problem:

Split-Tree kann unbalanciert sein => unbalancierter Split (z.B. nur eine Kindseite in der einen der Nachfolgeseiten)

- Lösung: Supernodes
  Für diesen Fall sieht der X-tree vor, die Directoryseite gar nicht zu splitten, sondern einen Supernode mit der doppelten Länge anzulegen
- Bei hohen Dimensionen entstehen deshalb zunehmend Supernodes

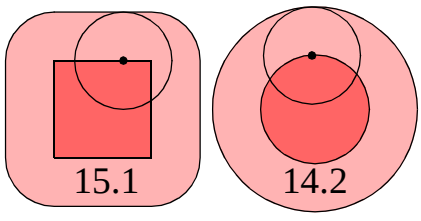
Nachteile des X-tree

- Durch Seiten unterschiedlicher Größe Probleme der Freispeicherverwaltung
- Überlappung entsteht nicht nur bei der Split-Operation sondern z.B. auch beim normalen Einfügen, wenn eine Seitenregion vergrößert werden muß
- Weniger Flexibilität der Struktur, um sich an die Datenverteilung anzupassen (meist steht nur eine Splitebene zur Auswahl)
- Konzept der Supernodes wird nur für Directory genutzt, und auch nur um unbalancierten Split zu verhindern.
  => Später: Wähle Blockgröße so, daß Anfragebearbeitung optimiert wird

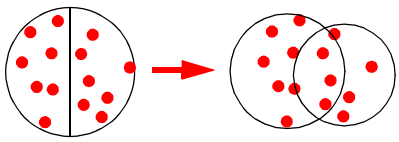
4.2.2 SS-tree (Similarity Search)

[WJ 96] White D.A., Jain R.: *Similarity Indexing with the SS-tree*. ICDE 1996, 516-523.

- Statt MBRs werden Kugeln als Seitenregionen verwendet
- Grundsätzlich weisen Kugeln Vorteile bei der Zugriffswahrscheinlichkeit auf, wenn auch kugelförmige Anfragen gestellt werden. Bei gleichem Volumen der Seitenregion ist die Minkowski-Summe kleiner.



- Dafür Probleme beim Split:
  Werden Kugeln geteilt, dann entstehen nicht wieder Kugeln. Beschreibt man die eingeschlossene Datenmenge doch wieder über die kleinste umgebende Kugel, entsteht hohe Überlappung

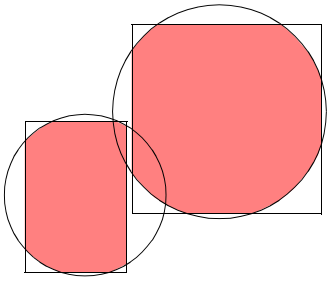


- Ermittlung der Kugeln – Zentroid-Konzept:
  Der Zentroid (Schwerpunkt) der Punktmenge einer Seite wird gebildet und der Radius minimal so bestimmt, daß alle Punkte eingeschlossen sind

- Einfüge-Algorithmus:
  Beim Baum-Durchlauf wird jeweils der Kindknoten gewählt, dessen Zentroid den geringsten Abstand vom neuen Punkt hat
- Überlauf-Behandlung:
  Re-Insert für die 30% am weitesten vom Zentroid entfernten Punkte
- Split-Kriterium
  basiert lediglich auf Vergleich der Varianzen:
  — Splitdimension ist die Dimension mit der höchsten Varianz der Punkte
  — Für die Splitebene werden alle Möglichkeiten betrachtet, die die Speicherauslastungsgarantie zulässt
  — Die Splitebene minimiert die Summe der Varianzen der beiden Nachfolgeseiten

4.2.3 SR-tree (Sphere Rectangle)

[KS 97] Katayama N., Satoh S.: *The SR-tree: An Index Structure for High-Dimensional Nearest Neighbor Queries*. ACM SIGMOD Conference, 1997, 369-380.



- Benutzt die Kombination (den *Schnittkörper*) aus einem Rechteck (MBR) und einer Kugel als Seitenregion
- Soll die Vorteile beider Ansätze kombinieren:
  — die geringere Zugriffswahrscheinlichkeit der Kugeln
  — die bessere Teilbarkeit der MBRs

4.2.4 TV-tree (Telescope Vector)

[LJF 94] Lin K., Jagadish H. V., Faloutsos C.: *The TV-tree: An Index Structure for High-Dimensional Data*. VLDB Journal 3, 1995, 517-542.

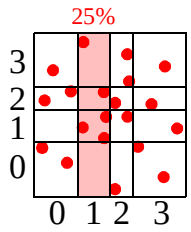
- Speziell für Vektoren, die mit Hilfe von dimensionsreduzierenden Verfahren wie z.B. Hauptachsentransformation (PCA) erzeugt wurden.
- Charakteristik solcher Vektoren:
  — hohe Varianz in den ersten Dimensionen
  — geringe Varianz in den letzten Dimensionen
- Eine Indexseite (Daten- oder Directoryseite) unterscheidet 3 Arten von Dimensionen
  — inaktive Dimensionen
  — aktive Dimensionen
  — sonstige Dimensionen
- Eine Dimension, die in einer Seite *inaktiv* ist, ist in einer Vergängerseite (bzgl. Baumhierarchie) *aktiv*
- Die Anzahl α der *aktiven* Dimensionen pro Seite ist im gesamten Index einheitlich
  — Systemparameter
  — experimentell wurde $\alpha = 2$ als optimal bestimmt,
  d.h. jeder Level des Index bestimmt $\alpha = 2$ Dimensionen
- Für eine Seitenregion sind also nur die α aktiven Dimensionen spezifiziert.
  — In diesen Dimensionen hat die Region die Form einer L_p-Kugel, $p \in \{0, 1, \infty\}$

— In den *inaktiven* Dimensionen "erbt" die Seite lediglich die Form der Vorgänger
— Die sonstigen Dimensionen sind unspezifiziert, d.h. gesamter Datenraum

4.2.5 VA-file (Vector Approximation)

[WSB 98] Weber R., Schek H.-J., Blott S.: *A Quantitative Analysis and Performance Study for Similarity-Search Methods in High-Dimensional Spaces.* Proc. 24th Int. Conf. on Very Large Databases (VLDB), New York, USA. Morgan Kaufmann, 1998, pp. 194-205

- Keine hierarchische Indexstruktur, sondern eine Variante des sequenziellen Scan (alle Punkte werden betrachtet)
- Einfache Einfüge-Operation: Lediglich am Ende anfügen.
- Die Daten werden verlustbehaftet komprimiert:
  — ein unregelmäßiges Gitter (quantilbasiert) über Datenraum gelegt
  — dadurch Aufteilung des Datenraums in $2^{d \cdot r}$ Zellen
  — statt der exakten Punktkoordinaten: Speicherung der Zellennummer
  — Reduktion des Datenvolumens auf $r/32$ (bei 32-Bit-**float**s)
  — die Lesezeit für die Daten reduziert sich um denselben Faktor



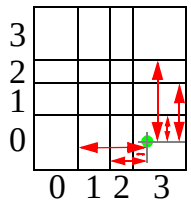
- Die CPU-Zeit für die Berechnung der Distanzen reduziert sich durch Trick geringfügig:
  — Pro Dimension werden die Abstandsquadrate zwischen dem Anfragepunkt und den einzelnen Partitionierungslinien vorberechnet und tabelliert

— Hierdurch wird die gewöhnliche Abstandsberechnung

$$\text{dist}^2 = \sum_{0 \le i < d} (p_i - q_i)^2$$

ersetzt durch folgende Operation

$$\text{dist}^2_{\text{va}} = \sum_{0 \le i < d} \text{lookup}_i[c_i]$$



- Durch verlustbehaftete Kompression Informationsverlust:
  — nur wenn Zelle ganz von Bereichsanfrage eingeschlossen, sicherer Treffer
  — fast immer Mehrdeutigkeiten bei nächstem Nachbarn
- Verfeinerungsschritt ist nötig:
  — Lade exakte Punktinfo (1 wahlfreier Zugriff pro Kandidat) und ermittle Distanz. Die exakten Punktinformationen sind in einer separaten Datei gespeichert, die genauso wie das VA-File sortiert ist.
  — Bereichsanfrage: Jede (echt) geschnittene Zelle muß verfeinert werden
  — NN-Algorithmus: Wie im indexbasierten Fall sind verschiedene Strategien denkbar
- Bei grober Gitterauflösung r hohe Kosten für die Verfeinerungsschritte
  — typischerweise 6-8 Bit optimal, jedoch abhängig von der Datenverteilung