Ludwig Maximilians Universität München
Institut für Informatik
Lehr- und Forschungseinheit für Datenbanksysteme

Skript zur Vorlesung

Datenbanksysteme II

Sommersemester 2006

Kapitel 6: Ähnlichkeitsmodelle für Sequenzdaten

Vorlesung: Christian Böhm
Übungen: Elke Achtert, Peter Kunath, Alexey Pryakhin

http://www.dbs.informatik.uni-muenchen.de/Lehre/DBSII

Inhalt

Inhalt

Einführung (1)

- Eine **Sequenz** der Länge n ist eine Abbildung der Indexmenge $I_n = \{1, \ldots, n\}$ in einen Wertebereich W: $I_n \rightarrow W$

- Klassifikation von Sequenzen anhand des Wertebereichs:
  - **nominale Werte** (Kategorien, Alphabete, allgemein: Aufzählungstypen)
    Beispiele:
    - Texte: $I_n \rightarrow$ Buchstaben
    - DNA-Sequenzen: $I_n \rightarrow$ Nukleinsäuren {C, G, A, T}
    - Proteinsequenzen: $I_n \rightarrow$ Aminosäuren {LEU, ARG, …}
  - **kontinuierliche Werte** (reelle Zahlen)
    Beispiele (allg. Zeitreihen):
    - Aktienkurse: $I_n \rightarrow$ Kurswerte
    - Fieberkurven: $I_n \rightarrow$ Temperaturwerte

Einführung (2)

- Klassen von Anfragen
  - **Vollsequenzsuche**: Sequenzen sind über ihre gesamte Länge ähnlich
  - **Teilsequenzsuche**: Suche nach Vorkommen von kurzen Anfragesequenzen in (längeren) Datenbanksequenzen

Inhalt

Suche auf Zeitreihen fester Länge (1)

- **Anfragebeispiele**
  - "Identifiziere Unternehmen mit einem ähnlichen Umsatzverlauf."
  - "Bestimme Produkte mit einer ähnlichen Entwicklung der Verkaufszahlen."
  - "Ermittle Aktien mit einem ähnlichen Kursverlauf."
  - "Stelle fest, ob ein Musikstück zu einem der geschützten Werke ähnlich ist."

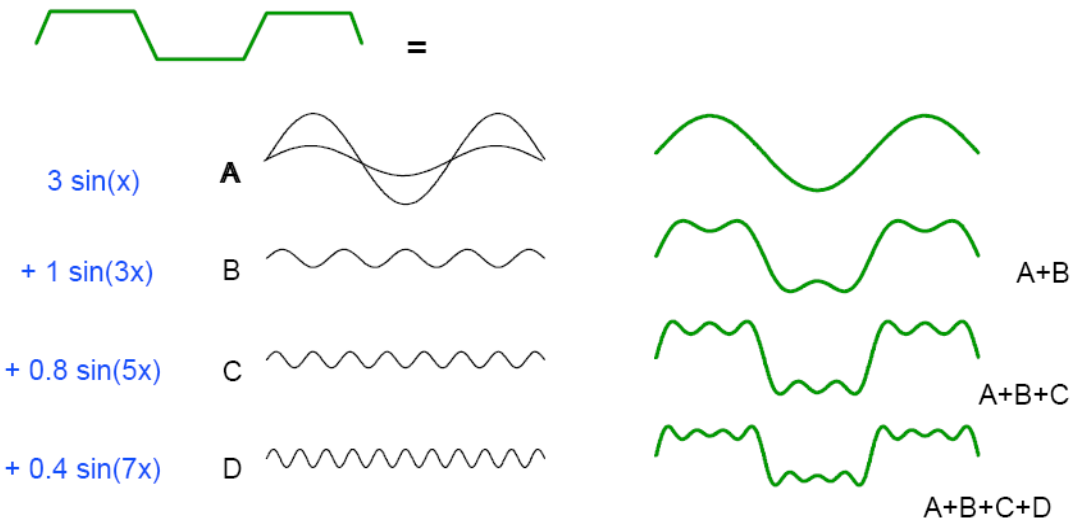
Suche auf Zeitreihen fester Länge (2)

- **Anfragebearbeitung**
  - Beschränkung auf Vollsequenzsuche
  - Sequenzen $I_n \rightarrow IR$ sind n-dimensionale Vektoren aus IR^n
  - Diskrete Fourier Transformation (DFT): Abbildung der Sequenzen vom Zeit- in den Frequenzbereich
  - Kürzung der Sequenzen aufgrund der Beobachtung, dass nur tiefe Frequenzen eine praktische Bedeutung haben
  - D.h. Darstellung langer Zeitreihen durch kurze Sequenzen von Frequenzen

Diskrete Fourier-Transformation (1)

Grundidee:

Beschreibe beliebige periodische Funktion als gewichtete Summe periodischer Grundfunktionen (Basisfunktionen) mit unterschiedlicher Frequenz



Diskrete Fourier-Transformation (2)

- **Fouriers Theorem**:
  Jede beliebige periodische Funktion lässt sich darstellen als Summe von Kosinus- und Sinus-Funktionen unterschiedlicher Frequenzen.

- Transformation verändert eine Funktion nicht, sondern stellt sie nur anders dar
- Transformation ist umkehrbar → inverse DFT
- Analog zum Basiswechsel in der Vektorrechnung

Diskrete Fourier-Transformation (3)

Formal

- Gegeben sei ein Signal $x = [x_t]$, $t = 0, \ldots, n-1$
- Die **DFT** von x ist eine Sequenz $X = [X_f]$ von n komplexen Zahlen für die Frequenzen $f = 0, \ldots, n-1$ mit

$$X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp\left(\frac{-i2\pi ft}{n}\right) = \underbrace{\frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \cos\left(\frac{2\pi ft}{n}\right)}_{\text{Realteil}} \underbrace{- i \cdot \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \sin\left(\frac{2\pi ft}{n}\right)}_{\text{Imaginärteil}}$$

  wobei i die imaginäre Einheit bezeichnet, d.h. $i^2 = -1$.
- Der Realteil gibt den Anteil der Kosinus- und der Imaginärteil den Anteil der Sinusfunktionen in der jeweiligen Frequenz f an.

Diskrete Fourier-Transformation (4)

- Durch die **inverse DFT** wird das ursprüngliche Signal x wieder hergestellt:

$$x_t = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} X_f \exp(i2\pi ft / n) \qquad t = 0, \ldots, n-1 \text{ } (t\text{: Zeitpunkte})$$

- $[x_t] \leftrightarrow [X_f]$ bezeichnet ein **Fourier-Paar**, d.h. $\text{DFT}([x_t]) = [X_f]$ und $\text{DFT}^{-1}([X_f]) = [x_t]$.
- Die DFT ist eine **lineare Abbildung**, d.h. mit $[x_t] \leftrightarrow [X_f]$ und $[y_t] \leftrightarrow [Y_f]$ gilt auch:
  - $[x_t + y_t] \leftrightarrow [X_f + Y_f]$ und
  - $[ax_t] \leftrightarrow [aX_f]$ für ein Skalar $a \in IR$

Satz von Parseval (1)

Energie einer Sequenz

- Die *Energie* $E(c)$ von c ist das Quadrat der Amplitude: $E(c) = |c|^2$.
- Die *Energie* $E(x)$ einer Sequenz $\boldsymbol{x}$ ist die Summe aller Energien über die Sequenz:

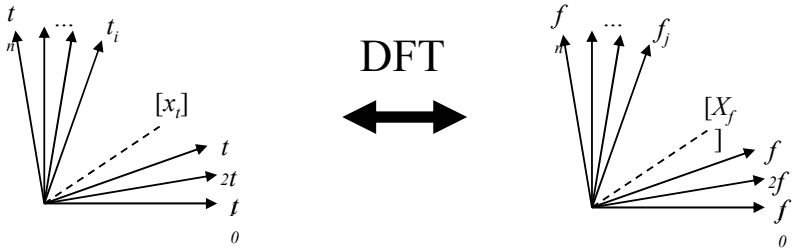
$$E(x) = \|x\|^2 = \sum_{t=0}^{n-1} |x_t|^2$$

Satz von Parseval

- Die Energie eines Signals im Zeitbereich ist gleich der Energie im Frequenzbereich.
  Formal: Sei X die DFT von x, dann gilt:

$$\sum_{t=0}^{n-1} |x_t|^2 = \sum_{t=0}^{n-1} |X_f|^2$$

Satz von Parseval (2)



- Mit anderen Worten: Die euklidische Distanz zweier Signale x und y stimmt im Zeit- und im Frequenzbereich überein:
  $\| x - y \|^2 = \| X - Y \|^2$

- **Grundidee der Anfragebearbeitung**:
  Als Ähnlichkeitsfunktion für Sequenzen wird die euklidische Distanz verwendet:

$$D(x, y) = \|x - y\| = \sqrt{\sum_{t=0}^{n-1} |x_t - y_t|^2}$$

- Der Satz von Parseval ermöglicht nun, die Distanzen im Frequenz- statt im Zeitbereich zu berechnen:

$$D(x, y) = D(X, Y)$$

Kürzen der Sequenzen

Kürzen der Sequenzen für die Indexierung

- In der Praxis haben die tiefsten Frequenzen die größte Bedeutung.
- Die ersten Frequenz-Koeffizienten enthalten also die wichtigste Information.
- Für den Aufbau eines Index werden die transformierten Sequenzen gekürzt, d.h. von $[X_f], f = 0, 1, \ldots, n-1$ werden nur die ersten c Koeffizienten $[X_{f<c}]$, $c < n$, indexiert.
- Im Index kann dann eine untere Schranke der echten Distanz berechnet werden:

$$D_c(x,y) = \sqrt{\sum_{f=0}^{c-1} |x_f - y_f|^2} \leq \sqrt{\sum_{f=0}^{n-1} |x_f - y_f|^2} = D(x,y)$$

- Diese Eigenschaft ist wichtig für die Suche, weil sie die Vollständigkeit der Ergebnisse des Index garantiert, d.h. die Ergebnisse aus dem Index bilden eine Obermenge der tatsächlichen Ergebnisse.

Anfrageablauf

- Gesucht sind alle Objekte o in der Datenbank, die sich höchstens um ε vom Anfrageobjekt q unterscheiden: $\{o \in DB \mid D(o, q) \leq \varepsilon\}$

- **Filterschritt**: Basierend auf einem Index mit $c < n$ Koeffizienten wird eine Obermenge der Ergebnisse ermittelt (Kandidaten).
  - Der Filterschritt ist *vollständig*:
    $\{o \in DB \mid D(o, q) \leq \varepsilon\} \subseteq \{o \in DB \mid D_c(o, q) \leq \varepsilon\}$
    (tatsächliche Ergebnisse) (Kandidaten aus Index)
  - Beweis der **Vollständigkeit der Anfragebearbeitung**:
    Wegen $D_c(o, q) \leq D(o, q)$ gilt für jedes $o \in DB$ mit $D(o, q) \leq \varepsilon$ auch $D_c(o, q) \leq \varepsilon$. D.h. es gibt keinen Treffer $o' \in DB$ mit $D(o', q) \leq \varepsilon < D_c(o', q)$, der nicht in der Kandidatenmenge enthalten ist.

- **Verfeinerungsschritt** : Berechnet die exakten Distanzwerte $D(o, q)$ auf den vollständigen Sequenzen und gewährleistet so, dass die ausgegebenen Resultate tatsächlich das Ähnlichkeitskriterium $D(o, q) \leq \varepsilon$ erfüllen → **Korrektheit der Anfragebearbeitung**

Zusammenfassung

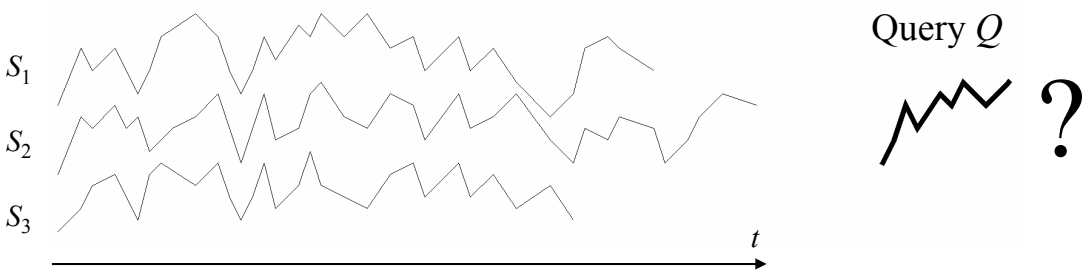
- Zeitreihen werden mit Hilfe der Diskreten Fourier-Transformation (**DFT**) vom Zeitbereich in den Frequenzbereich abgebildet.
- **Satz von Parseval**: Die euklidische Distanz ist invariant gegenüber der Fourier-Transformation, d.h. sie hat den gleichen Wert im Zeit- wie im Frequenzbereich.
- Beobachtung: nur die tiefsten Frequenzen haben eine praktische Bedeutung. Experimente zeigen, dass die ersten 1 bis 3 Koeffizienten genügen.
- Insgesamt: Abbildung von hochdimensionalen Zeitreihen in niedrig-dimensionale Sequenzen von Frequenzwerten.
- **Mehrstufige Anfragebearbeitung** z.B. mit R*-Baum im Filterschritt.

Inhalt

Suche nach Teilsequenzen (1)

- Gegeben seien N Sequenzen $S_1, S_2, \ldots, S_N$ beliebiger Längen, eine Anfragesequenz Q, eine Ähnlichkeitstoleranz ε sowie eine Distanzfunktion D für Sequenzen.

- Gesucht sind diejenigen S_i, die Teilsequenzen $s = S_i[k, k+\text{len}(Q)-1]$ beinhalten, deren Abstand $D(s, Q)$ höchstens ε beträgt. Zu jedem S_i soll auch die Position k der entsprechenden Teilsequenz s ausgegeben werden.



Suche nach Teilsequenzen (2)

Anwendungsbeispiele

- Finanz- und betriebswirtschaftliche Datenbanken: "Finde Beispiele aus der Vergangenheit, bei denen sich die Verkaufsentwicklung ähnlich entwickelt hat wie bei unserem Produkt in den vergangenen drei Monaten."
- Technisch-wissenschaftliche Datenbanken: "Wann konnte ein ähnliches Verhalten des Sonnenwindes gemessen werden wie heute vormittag von 10.00 bis 12.00 Uhr?"

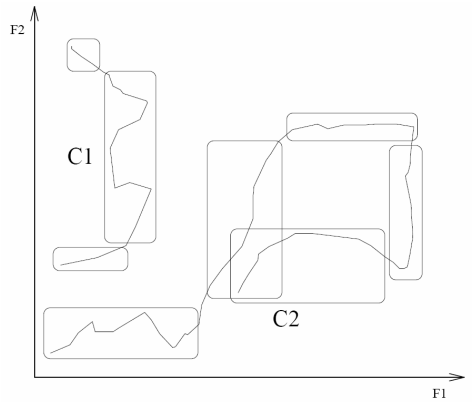
Mindestlänge *w* für Anfragesequenzen

- Im weiteren wird eine Mindestlänge w für Anfragesequenzen angenommen.
- Beispiel: Bei Aktienkursen ist man an wöchentlichen oder monatlichen Mustern der Kursverläufe interessiert, da sie weniger rauschanfällig sind.
- Kürzere Anfragen werden nach wie vor unterstützt (durch sequentielle Suche).

Abbildung der Zeitreihen

- Über jede der Sequenzen wird ein Fenster der Länge w geschoben.
- An jeder Fensterposition wird die sichtbare Teilsequenz (wie oben) durch DFT codiert und stellt dadurch einen Punkt im w-dimensionalen Frequenzenraum dar.
- Eine Sequenz S wird also durch eine Folge von len(S) – w + 1 vielen Punkten der Dimension w repräsentiert, d.h. ein (Feature-)Punkt für jede Fensterposition.

Beispiel (Quelle: [FRM 94]):
Zwei Sequenzen $S1$ und $S2$ mit den DFT-Folgen $C1$ und $C2$ (hier im 2D)



Suche über den Featurepunkten

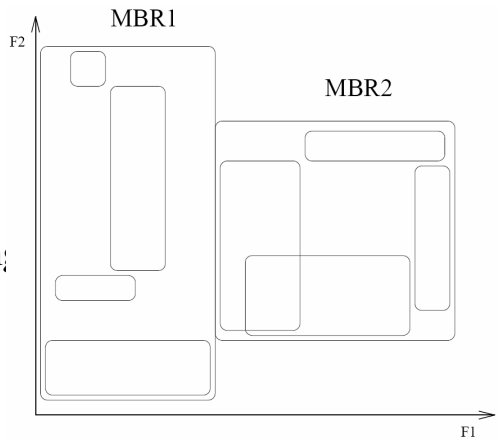
- **Sequentieller Scan**: Durchlaufe die Menge aller Punkte in der Datenbank
- Verwendung mehrdimensionaler **Indexstrukturen** (z.B. R*-Baum, X-Baum, …)
  - Bereichsanfrage: ε-Bereich um Anfragepunkt im Index anfragen, dann verfeinern
  - Dieser Suchalgorithmus ist vollständig (Beweis wie vorher)
  - Experimente zeigen, dass diese Methode etwa doppelt so langsam wie eine sequentielle Suche ist, eine bessere Lösung ist also dringend erwünscht

Indexstruktur zur Anfrageunterstützung (1)

Zusammenfassen von Punkten zu Bereichen.

- Beobachtung: Aufeinanderfolgende Punkte liegen oft nahe beieinander, da die Inhalte zweier stark überlappender Fenster sehr ähnlich sind.
- Idee: Aufspaltung der Punktefolgen in Teilfolgen, dann Repräsentation der Teilfolgen durch ihre minimal umgebenden (Hyper-)Rechtecke.

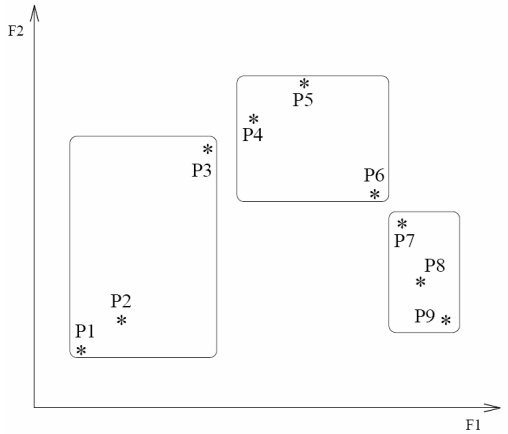
- Im Beispiel (Quelle: [FRM 94]):
  - Speicherung weniger Rechtecke statt vieler Punkte und
  - (hierarchische) Zusammenfassung der Rechtecke im Index (MBR = minimum bounding rectangle)

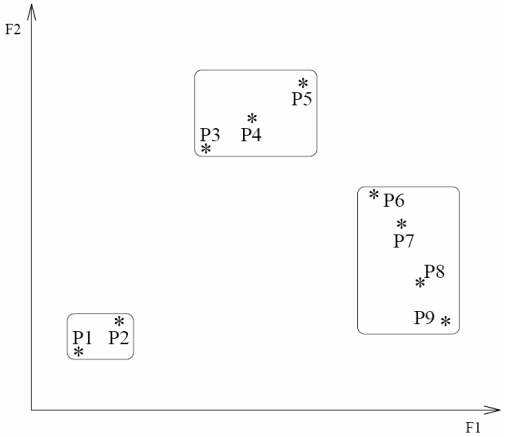


Indexstruktur zur Anfrageunterstützung (2)

Einfügen neuer Daten

- Wesentliche Aufgabe: Unterteilung von Punktfolgen C in "gute" Teilfolgen
- Punktezahl kann fix sein (z.B. 50) oder von Gesamtlänge abhängen (z.B. $\sqrt{\text{len}(C)}$)
- Aufteilung mit fester Punktanzahl pro Teilfolge kann schlecht sein (Quelle: [FRM 94]):





Indexstruktur zur Anfrageunterstützung (3)

- Idee: Heuristik zur Aufteilung bei Minimierung der erwarteten Plattenzugriffe
- Algorithmus zur Zerlegung einer Punktfolge in Teilfolgen:

Ordne den ersten Punkt einer (trivialen) Teilfolge zu.
Für jeden nachfolgenden Punkt *p*:
Falls *p* die Grenzkosten der aktuellen Teilfolge erhöht,
dann starte eine neue Teilfolge mit *p*,
sonst füge *p* in die aktuelle Teilfolge ein.

- Grenzkosten für *k* Punkte in einer entsprechenden Teilfolge *L*: Zugriffe(*L*) / *k*
- *Erwartete Zugriffe für ein n*-dim. Rechteck *L* aus $[0, 1)^n$ mit den Seitenlängen $L_1, L_2, \ldots, L_n$:

$$\text{Zugriffe}(L) = \prod_{i=1}^{n} (L_i + 0{,}5)$$

Anfragebearbeitung mit Anfragesequenzen der Minimallänge *w*

- Anfrage: "Suche Teilsequenzen, die zur Sequenz *q* höchstens den Abstand ε haben."
- Algorithmus:

1. Bilde die Anfragesequenz *q* auf den Punkt q_f im Featureraum ab.
2. Identifiziere mit Hilfe des Index diejenigen Teilfolgen, deren minimal umgebende Rechtecke die Kugel um q_f mit dem Radius ε schneiden.
3. Untersuche die zugehörigen Teilsequenzen und verwirf falsche Antworten.

- Die *Korrektheit* der Antworten wird im Schritt 3 sichergestellt ("Verfeinerung").
- Die *Vollständigkeit* der Antworten wird dadurch garantiert, dass im Schritt 2 ("Filterschritt") umgebende Rechtecke verwendet werden; es können also keine Resultate verloren gehen.

Präfixsuche für längere Anfragesequenzen

- **Anfragebearbeitung mit längeren Anfragesequenzen *q*, d.h. len(*q*) > *w***
- Problem: Index kennt nur Sequenzen der Länge *w*
- *Idee Präfixsuche*: Suche mit einer Teilsequenz von *q* der Länge *w*, z.B. dem Präfix.
- Die Präfixsuche stellt die **Vollständigkeit** sicher, da auf jeden Fall eine Obermenge der tatsächlichen Resultate ermittelt wird:
  - **Lemma:** Unterscheiden sich zwei Sequenzen *s* und *q* derselben Länge *l* (bezüglich des euklidischen Abstands D) um nicht mehr als ε, dann stimmen auch jeweils zwei entsprechende Teilsequenzen *s*[*i*:*j*] und *q*[*i*:*j*] im selben Rahmen ε überein:

$$\mathrm{D}(s,q) \le \varepsilon \Rightarrow D(\mathrm{s[i:j]}, \mathrm{q[i:j]}) \le \varepsilon \text{ für } 1 \le i \le j \le l$$

  - **Beweis**: Die Behauptung folgt aus folgender Eigenschaft:

$$\mathrm{D}(s[i:j], q[i:j]) = \sqrt{\sum_{k=i}^{j} |s_k - q_k|^2} \le \sqrt{\sum_{k=1}^{l} |s_k - q_k|^2} = \mathrm{D}(s,q)$$

Zerlegungssuche für sehr lange Anfragesequenzen (1)

- **Anfragebearbeitung für sehr lange Anfragesequenzen *q*, d.h. len(*q*) >> *w***
- Problem: Volumen der Anfragekugel im Featureraum ist sehr groß
- Idee: Zerlege die Anfragesequenz in mehrere Teilsequenzen der Länge *w*
- *Annahme: Die Länge der Anfragesequenz q* ist ein Vielfaches von *w*: len(*q*) = *p*·*w* (andernfalls kann die obige Präfixlösung angewandt werden).
- Algorithmus *Zerlegungssuche*:
  1. Zerlege die Anfragesequenz *q* in *p* Teilsequenzen der Länge *w*, die *p* Kugeln mit Radius $\varepsilon/\sqrt{p}$ im Featureraum entsprechen.
  2. Hole mit Hilfe des Index alle Teilfolgen aus der Datenbank, deren umgebendes Rechteck eine der *p* Anfragekugeln schneidet.
  3. Untersuche die zugehörigen Teilsequenzen, um falsche Resultate zu verwerfen.

Zerlegungssuche für sehr lange Anfragesequenzen (2)

- **Vollständigkeit des Algorithmus**
  - Seien zwei Sequenzen s und q gleich lang, d.h. len(s) = len(q) = $p \cdot w$.
  - Für das folgende Lemma betrachten wir die p disjunkten Teilsequenzen
    $s_i = s[i \cdot w+1 : (i+1) \cdot w]$ und $q_i = q[i \cdot w+1 : (i+1) \cdot w]$ für $i = 0, \ldots, p-1$.
  - **Lemma:**
    Unterscheiden sich zwei Sequenzen s und q derselben Länge l höchstens um ε, dann gibt es mindestens ein Paar s_i und q_i von sich entsprechenden Teilsequenzen mit einem Abstand kleiner oder gleich $\varepsilon / \sqrt{p}$:

$$D(s,q) \le \varepsilon \Rightarrow \exists i = 0,\ldots,p-1 : D(s_i, q_i) \le \varepsilon / \sqrt{p}$$

Zerlegungssuche für sehr lange Anfragesequenzen (3)

- **Beweis** (durch Widerspruch):
  Sei $D(s, q) \le \varepsilon$. Falls alle p Teilsequenzen einen Abstand größer als $\varepsilon / \sqrt{p}$ hätten, wäre der Gesamtabstand größer als ε:

$$\text{Aus } D(s_i, q_i) = \sqrt{\sum_{j=i \cdot w+1}^{(i+1) \cdot w} (s_i[j] - q_i[j])^2} > \varepsilon / \sqrt{p}$$

$$\text{für alle } i = 0,\ldots, p-1$$

$$\text{folgt } D^2(s_i, q_i) = \sum_{j=i \cdot w+1}^{(i+1) \cdot w} (s_i[j] - q_i[j])^2 > \varepsilon^2 / p,$$

$$\text{und damit } D^2(s,q) = \sum_{j=1}^{p \cdot w} (s[j] - q[j])^2 > p \cdot \varepsilon^2 / p = \varepsilon^2,$$

$$\text{also } D(s,q) > \varepsilon$$

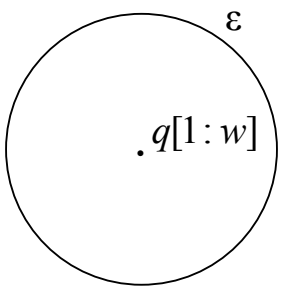
Präfixsuche vs. Zerlegungssuche

Volumen V der Anfragekugeln im r-dimensionalen Featureraum:

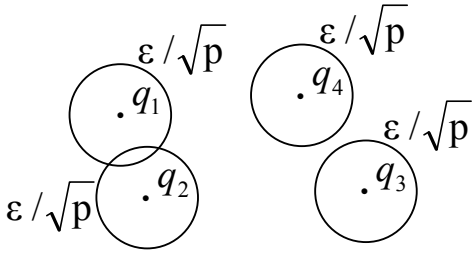
- **Präfixsuche**: $V(\varepsilon) = K \cdot \varepsilon^r$
  (K bezeichne das Volumen der r-dim. Einheitskugel).

- **Zerlegungssuche**: p Kugeln á $K(\varepsilon / \sqrt{p})^r$ also
  $$V(\varepsilon) = Kp\varepsilon^r / \sqrt{p^r} = K\varepsilon^r p^{1-r/2}$$

- **Ergebnis**:
  Zerlegungssuche ist effizienter, falls $p^{1-r/2} < 1$, d.h. falls $r > 2$.



Präfixsuche



Zerlegungssuche