2.2 Ähnlichkeitsmodelle für Sequenzdaten

- Eine Sequenz der Länge n ist eine Abbildung der Indexmenge $I_n = \{1, \ldots, n\}$ in einen Wertebereich W: $I_n \rightarrow W$
- Sequenzen lassen sich anhand ihres Wertebereichs klassifizieren:
  — nominale Werte (Kategorien, Alphabete, allgemein: Aufzählungstypen)
  Beispiele:
  Texte: $I_n \rightarrow$ Buchstaben
  DNA-Sequenzen: $I_n \rightarrow$ Nucleinsäuren {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
- Klassen von Anfragen
  — Vollsequenzsuche: Sequenzen sind über ihre gesamte Länge ähnlich
  — Teilsequenzsuche: Suche nach Vorkommen von kurzen Anfragesequenzen in (längeren) Datenbanksequenzen

2.2.1 Edit-Distanz: Ähnlichkeit von allgemeinen Sequenzen

- Idee: Edit Distance D(q, s) ist die minimale Anzahl von Editierungsoperationen (Einfügen, Löschen, Ändern), um die Sequenz s in die Sequenz q zu überführen.
- Beispiel:
  D(“TÜRSCHLOSS”, “ABSCHUSS”) = 5, da zwei Löschungen (^) und drei Änderungen (:) nötig sind, um die erste Sequenz in die zweite umzuformen. Fünf Buchstaben bleiben unverändert erhalten (|), wie folgende Zuordnung (*Alignment*) zeigt:

```
T Ü R S C H L O S S
^ : : | | | ^ : | |
  A B S C H   U S S
```

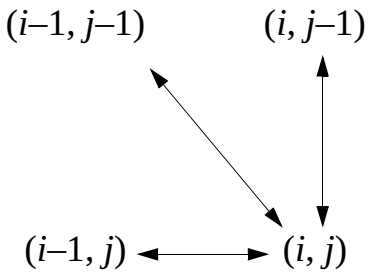
- Formale Definition:
  D(q, s) = D($q_{1..\text{len}(q)}$, $s_{1..\text{len}(s)}$) mit

$$\text{D}(q_{1..i}, s_{1..j}) = \begin{cases} j & \text{falls } i = 0 \\ i & \text{falls } j = 0 \\ \text{D}(q_{1..i-1}, s_{1..j-1}) & \text{falls } i > 0, j > 0 \text{ und } q_i = s_j \\ \min\{\text{D}(q_{1..i-1}, s_{1..j-1})+1, \text{D}(q_{1..i-1}, s_{1..j})+1, \text{D}(q_{1..i}, s_{1..j-1})+1\} & \text{sonst} \end{cases}$$

- Berechnungsschema (“dynamische Optimierung”):
  D(q, s) kann in O(len(q) · len(s)) Zeit berechnet werden, wie folgendes Schema zeigt:

D	i	0	1	2	3	4	5	6	7	8	9	10
j			T	Ü	R	S	C	H	L	O	S	S
0												
1	A											
2	B											
3	S											
4	C											
5	H											
6	U											
7	S											
8	S											

- Deutung des Berechnungsschemas
  - — Horizontaler Schritt: (i–1, j) → (i, j)
    Löschen des aktuellen Zeichens q_i
  - — Vertikaler Schritt: (i, j–1) → (i, j)
    Einfügen des Zeichens s_{j-1} in q an der Position i
  - — Diagonaler Schritt: (i–1, j–1) → (i, j)
    Beibehalten bzw. Ändern des aktuellen Zeichens



- Die Änderungs-minimale Zuordnung zweier Sequenzen ist nicht immer eindeutig:

```
B A N A N E          B A N A N E
| | | : ^ ^   oder   | ^ ^ | | :
B A N D              B     A N D
```

- Neben dem paarweisen Alignment betrachtet man oft auch die änderungsminimale Zuordnung mehrerer Sequenzen (*Multiples Alignment*), etwa in der molekularen Bioinformatik: Vergleich mehrerer Protein- oder DNA-Sequenzen.

Verallgemeinerung: Gewichtung der Operationen

- Grundidee
  - — Bisher: Einheitliche Kosten für Änderungen, Einfügungen, Entfernungen
  - — Neu: Unterschiedliche Kosten $\gamma(x, y)$ für verschiedene Operationen $x \rightarrow y$
- Beispiele für die Kosten $\gamma(x, y)$ von Änderungsoperationen:
  - — für nominale Wertebereiche: $\gamma(x, y) = 0$ falls $x = y$, 1 sonst (wie oben)
  - — für numerische Wertebereiche: $\gamma(x, y) = | x - y |$
  - — für beliebige Wertebereiche: $\gamma(x, y)$ aus Tabelle lesen
- Die Kosten von Einfüge- und Löschoperationen werden ebenfalls individualisiert:
  - — Einfügen eines Zeichens y: $\gamma(\lozenge, y)$ (d.h. Ändern des 'leeren' Zeichens $\lozenge$)
  - — Löschen eines Zeiches x: $\gamma(x, \lozenge)$ (d.h. Ändern zum 'leeren' Zeichen $\lozenge$)
- Die Definition der γ-gewichteten Edit-Distanz $D_\gamma(i, j > 0)$ lautet damit:

  $D_\gamma(\lambda, \lambda) = 0$ — λ bezeichnet die leere Sequenz

  $D_\gamma(\lambda, s_{1..j}) = D_\gamma(\lambda, s_{1..j-1}) + \gamma(\lozenge, s_j)$— Einfügen der Sequenz $s_{1..j}$

  $D_\gamma(q_{1..i}, \lambda) = D_\gamma(q_{1..i-1}, \lambda) + \gamma(q_i, \lozenge)$ — Entfernen der Sequenz $q_{1..i}$

  $D_\gamma(q_{1..i}, s_{1..j}) = \min \{D_\gamma(q_{1..i-1}, s_{1..j-1}) + \gamma(q_i, s_j)$, — ggf. Ändern von q_i zu s_j

  $D_\gamma(q_{1..i-1}, s_{1..j}) + \gamma(q_i, \lozenge)$, — Löschen von q_i

  $D_\gamma(q_{1..i}, s_{1..j-1}) + \gamma(\lozenge, s_j)\}$ — Einfügen von s_j

2.2.2 Ähnlichkeitsmodell für Zeitreihen fester Länge

[AFS 93] Agrawal R., Faloutsos C., Swami A.: *Efficient Similarity Search in Sequence Databases*. Proc. FODO 1993 (LNCS 730), 69-84.

- 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."
- Charakterisierung des Verfahrens
  - — Beschränkung auf Vollsequenzsuche (Teilsequenzen werden nicht betrachtet).
  - — Sequenzen $I_n \rightarrow \Re$ werden als n-dimensionale Vektoren aus $\Re^n$ betrachtet.
  - — Mit Hilfe der Diskreten Fourier-Transformation (DFT) werden die Sequenzen vom Zeitbereich in den Frequenzbereich abgebildet.
  - — Aufgrund der Beobachtung, dass nur die tiefsten Frequenzen eine praktische Bedeutung haben, werden die Vektoren gekürzt.
  - — Lange Zeitreihen werden also durch kurze Sequenzen von Frequenzen dargestellt.
- Diskrete Fourier-Transformation (DFT)
  - — Gegeben sei ein Signal $\boldsymbol{x} = [x_t]$, $t = 0, \ldots, n - 1$
  - — Die DFT von $\boldsymbol{x}$ ist eine Sequenz $\boldsymbol{X} = [X_f]$ von n komplexen Zahlen, $f = 0, \ldots, n - 1$ mit

$$X_f = \frac{1}{\sqrt{n}} \sum_{t=0}^{n-1} x_t \exp(-i 2\pi f t / n), \quad f = 0, 1, \ldots, n-1 \text{ (}f\text{: Frequenzen)}$$

wobei i die komplexe Einheit bezeichnet, d.h. $i^2 = -1$.

— Durch die inverse DFT wird das ursprüngliche Signal $\boldsymbol{x}$ wiederhergestellt:

$$x_t = \frac{1}{\sqrt{n}} \sum_{f=0}^{n-1} X_f \exp(i 2\pi f t / n), \quad t = 0, 1, \ldots, n-1 \text{ (}t\text{: Zeitpunkte)}$$

$[x_t] \leftrightarrow [X_f]$ bezeichne ein Fourier-Paar, d.h. $\text{DFT}([x_t]) = [X_f]$ und $\text{DFT}^{-1}([X_f]) = [x_t]$.

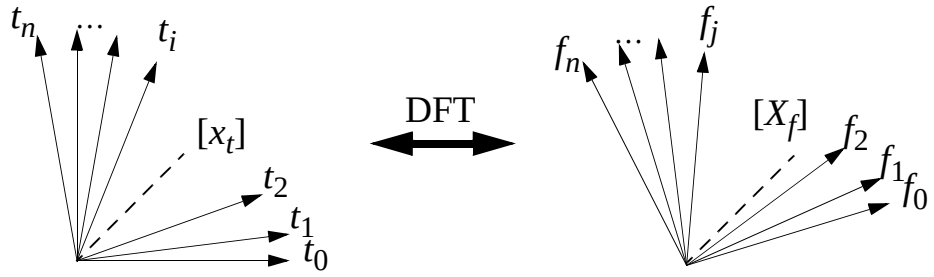
- Die DFT ist eine *lineare Abbildung*, d.h. mit $[x_t] \leftrightarrow [X_f]$ und $[y_t] \leftrightarrow [Y_f]$ gilt auch
  (*i*) $[x_t + y_t] \leftrightarrow [X_f + Y_f]$ und
  (*ii*) $[ax_t] \leftrightarrow [aX_f]$ für ein Skalar $a \in \Re$

- Verschiebungsinvarianz
  — $A = |c|$ heißt die *Amplitude* und φ die *Phase* einer komplexen Zahl $c = a + ib = A \cdot e^{i\varphi}$.
  — Eine Verschiebung im Zeitbereich verschiebt nur die Phase, nicht jedoch die Amplitude der Fourierkoeffizienten: $[x_{t-t0}] \leftrightarrow [X_f \ \exp(i 2\pi f t_0 / n)]$

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

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

- Satz von Parseval
  Die Energie eines Signals im Zeitbereich ist gleich der Energie im Frequenzbereich.
  Formal: Sei $\boldsymbol{X}$ die DFT von $\boldsymbol{x}$, dann gilt: $\sum_{t=0}^{n-1} |x_t|^2 = \sum_{f=0}^{n-1} |X_f|^2$



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

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

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

  - Der Satz von Parseval ermöglicht nun, die Distanzen im Frequenzbereich statt im Zeitbereich zu berechnen: $D(\boldsymbol{x}, \boldsymbol{y}) = D(\boldsymbol{X}, \boldsymbol{Y})$

- Kürzen der Sequenzen für die Indexierung
  - In praktischen Beispielen haben die tiefsten Frequenzen die größte Bedeutung.
  - Die ersten Frequenz-Koeffizienten enhalten 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:

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

  - Diese Eigenschaft ist wichtig für die Suche, weil sie die Vollständigkeit der Ergebnisse aus Index garantiert, d.h. die Ergebnisse aus dem Index bilden eine Obermenge der tatsächlichen Ergebnisse. Es ergibt sich also folgender Anfrageablauf:

- Anfragebearbeitung
  - Ausgegeben werden sollen alle Objekte $\boldsymbol{o}$ in der Datenbank, die sich höchstens um ε vom Anfrageobjekt $\boldsymbol{q}$ (“query”) unterscheiden: $\{\boldsymbol{o} \in \text{DB} \mid D(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon\}$

  - *Vollständigkeit* der Anfragebearbeitung
    Ein Filterschritt basierend auf einem Index mit $c < n$ Koeffizienten kann nun eine Obermenge der Ergebnisse finden (Kandidaten), d.h. der Filterschritt ist *vollständig*:

    $$\{\boldsymbol{o} \in \text{DB} \mid D(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon\} \quad \subseteq \quad \{\boldsymbol{o} \in \text{DB} \mid D_c(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon\}$$
    $$\text{(tatsächliche Ergebnisse)} \qquad \text{(Kandidaten aus Index)}$$

    Beweis: Wegen $D_c(\boldsymbol{o}, \boldsymbol{q}) \leq D(\boldsymbol{o}, \boldsymbol{q})$ gilt für jedes $\boldsymbol{o}$ mit $D(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon$ auch $D_c(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon$. Es gibt also keinen Treffer $\boldsymbol{o'}$ mit $D(\boldsymbol{o'}, \boldsymbol{q}) \leq \varepsilon < D_c(\boldsymbol{o'}, \boldsymbol{q})$, der im Index verloren geht.

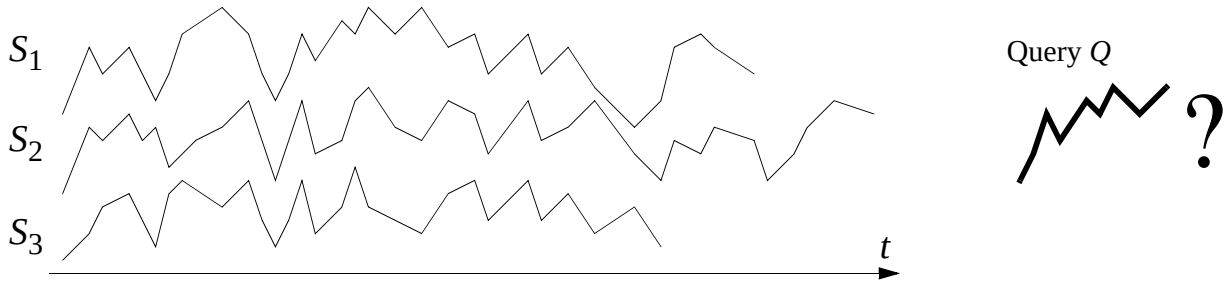
  - *Korrektheit* der Anfragebearbeitung
    Ein nachgeschalteter Verfeinerungsschritt berechnet die exakten Distanzwerte $D(\boldsymbol{o}, \boldsymbol{q})$ auf den vollständigen Sequenzen und gewährleistet so, daß die ausgegebenen Resultate tatsächlich das Ähnlichkeitskriterium $D(\boldsymbol{o}, \boldsymbol{q}) \leq \varepsilon$ erfüllen.

- 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 Fouriertransformation, 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 niedriger-dimensionale Sequenzen von Frequenzwerten.
  - — Mehrstufige Anfragebearbeitung mit R*-Baum im Filterschritt.

2.2.3 Modell für die Suche nach Teilsequenzen

[FRM 94] Faloutsos C., Ranganathan M., Manolopoulos Y.: *Fast Subsequence Matching in Time-Series Databases*. Proc. ACM SIGMOD Int. Conf. on Management of Data, 1994, 419-429.

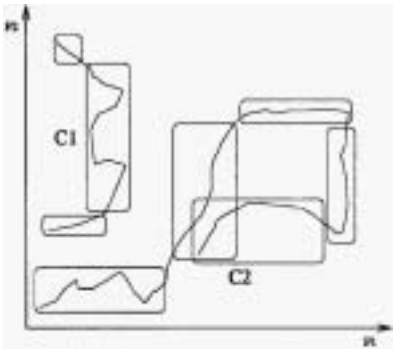
- Hier: Suche nach Teilsequenzen (nicht mehr nur vollständige Sequenzen)
  - — Gegeben seien N Sequenzen $S_1, S_2, \ldots, S_N$ beliebiger Längen, eine Anfragesequenz Q, eine Ähnlichkeitstoleranz ε sowie eine Distanzfunktion D für Sequenzen.
  - — Gesucht sind diejenigen S_i, die Teilsequenzen $s = S_i[k, k+\text{len}(Q) -1]$ beinhalten, deren Abstand $D(s, Q)$ höchstens ε beträgt. Zu jedem S_i soll auch die Position k der entsprechenden Teilsequenz s ausgegeben werden.



- 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:
    Zwei Sequenzen S_1 und S_2
    mit den DFT-Folgen C_1 und C_2
    (hier im 2D):



Quelle: [FRM 94]

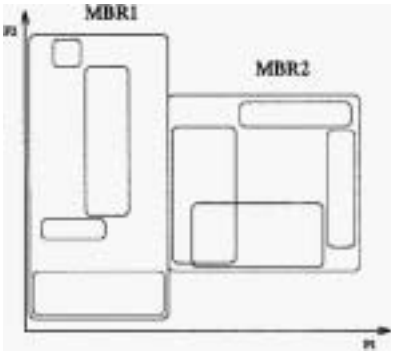
- Suche über den Featurepunkten
  - Sequentieller Scan: Durchlaufe die Menge aller Punkte in der Datenbank
  - Verwendung mehrdimensionaler Indexstrukturen (z.B. R*-Baum, X-Baum, …)

- Index für die einzelnen Punkte
  - 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.

- 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.

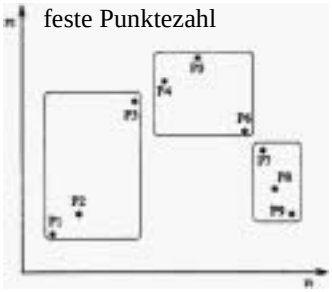
  - Im Beispiel:
    * Speicherung weniger Rechtecke statt vieler Punkte und

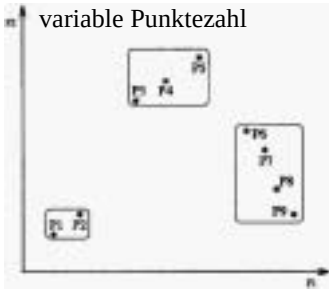
    * (hierarchische) Zusammenfassung der Rechtecke im Index (MBR = minimum bounding rectangle)



Quelle: [FRM 94]

- 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]

  - 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 verlorengehen.

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: Falls zwei Sequenzen *s* und *q* derselben Länge *l* sich (bezüglich des euklidischen Abstands D) um nicht mehr als ε unterscheiden, dann stimmen auch jeweils zwei entsprechende Teilsequenzen $s[i{:}j]$ und $q[i{:}j]$ im selben Rahmen ε überein:

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

Beweis: Die Behauptung folgt aus folgender Eigenschaft:

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

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.

- Vollständigkeit des Algorithmus
  — Seien zwei Sequenzen *s* und *q* gleich lang, d.h. len(*s*) = len(*q*) = *p*·*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: Falls zwei Sequenzen *s* und *q* derselben Länge *l* sich höchstens um ε unterscheiden, dann gibt es mindestens ein Paar s_i und q_i von sich entsprechenden Teilsequenzen mit einem Abstand kleiner oder gleich $\varepsilon/\sqrt{p}$:

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

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:

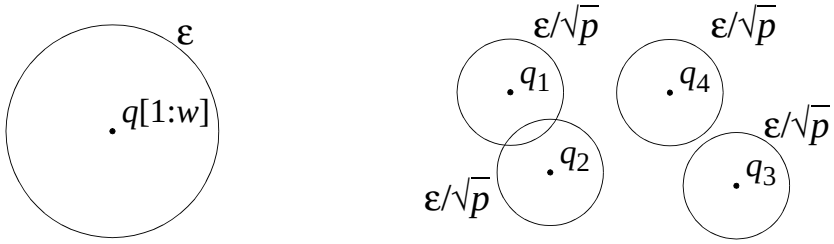
Aus $D(s_i, q_i) = \sqrt{\sum_{j=iw+1}^{(i+1)\cdot w} (s_i[j] - q_i[j])^2} > \varepsilon/\sqrt{p}$ für alle $i = 0, \ldots, p\text{–}1$

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

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

- Vergleich von *Präfixsuche* und *Zerlegungssuche*:
  - Betrachte das Volumen V der Anfragekugeln im r-dimensionalen Featureraum:
  - *Präfixsuche*: $V(\varepsilon) = K\,\varepsilon^r$ (K bezeichne das Volumen der r-dim. Einheitskugel).
  - *Zerlegungssuche*: p Kugeln á $K\cdot(\varepsilon/\sqrt{p})^r$, also $V(\varepsilon) = K\,p\,\varepsilon^r/\sqrt{p^r} = K\,\varepsilon^r\,p^{1-r/2}$

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



Präfixsuche Zerlegungssuche

Referenzen

[AFS 93] Agrawal R., Faloutsos C., Swami A.: *Efficient Similarity Search in Sequence Databases*. Proc. FODO 1993 (LNCS 730), 69-84.

[ALSS 95] Agrawal R., Lin K.-I., Sawhney H. S., Shim K.: *Fast Similarity Search in the Presence of Noise, Scaling, and Translation in Time-Series Databases*. Proc. Int. Conf. on Very Large Data Bases (VLDB), 1995, 490-501.

[BYÖ 97] Bozkaya T., Yazdani N., Özsoyoglu M.: *Matching and Indexing Sequences of Different Lengths*. Proc. Int. Conf. on Information and Knowledge Management (CIKM), 1997, 128-135.

[Fal 96] Faloutsos C.: *Searching Multimedia Databases by Content*. Boston [u.a.]: Kluwer, 1996.

[Fal 97] Faloutsos C: *Indexing of Multimedie Data*. In: Apers P. M. G., Blanken H. M., Houtsma M. A. W. (eds.): *Multimedia Databases in Perspective*. London: Springer, 1997, chapter 10, 219-245.

[FRM 94] Faloutsos C., Ranganathan M., Manolopoulos Y.: *Fast Subsequence Matching in Time-Series Databases*. Proc. ACM SIGMOD Int. Conf. on Management of Data, 1994, 419-429.

[WF 74] Wagner R. A., Fischer M. J.: *The String-to-String Correction Problem*. Journal of the ACM 21(1), 1974: 168-173.