Multimedia-Datenbanken

Kapitel 3: Multimedia-Daten – Text

Friedrich-Alexander-Universität Erlangen-Nürnberg
Technische Fakultät, Institut für Informatik
Lehrstuhl für Informatik 6 (Datenbanksysteme)

Prof. Dr. Klaus Meyer-Wegener

Wintersemester 2002 / 2003

Technische Universität Kaiserslautern
Fachbereich Informatik
AG Datenbanken und Informationssysteme

Dr. Ulrich Marder

Wintersemester 2003 / 2004

3. Multimedia-Daten

- **zunächst Medienobjekte, d. h. nur ein Medium**
- **bestehen aus (s. oben):**
  - Rohdaten
  - Registrierungsdaten
  - Beschreibungsdaten
- **Operationen dazu:**
  - anlegen (Eingabe, create)
  - ausgeben (an Programm oder Gerät, ggf. Formatumsetzung)
  - modifizieren
  - weitergeben
  - archivieren
  - auswerten (analysieren, aggregieren)
  - vergleichen (für die Suche)

Suche

- **insbesondere: nach Medienobjekten**
- **gewünschte Objekte beschreiben**
  - mit Hilfe von Merkmalen (Features, formatiert)
  - verbal (unformatiert)
  - durch Vergleichsobjekt (Muster)
- **Medienobjekte ohnehin mit formatierten Daten verknüpft**
  - anwendungsspezifischen (z. B. Archivnummer)
  - und Registrierungsdaten
  - evtl. auch Beschreibungsdaten (wenn formatiert)
- **Suchmechanismen zunächst nur über**
  - die formatierten Daten (klassische DB-Technik anwendbar)
  - Mustererkennung (Pattern Matching)
    - in den Rohdaten, z. B. Volltextsuche oder
    - in den Beschreibungsdaten (wenn unformatiert)

Inhaltsorientierte Suche

- **Verwendung dessen, was durch die Medienobjekte selbst dargestellt wird ("Inhaltsadressierung"):**
  - gefordert, aber kaum realisiert

- **Beispielanfragen**
  - gesucht werden:
    - Einträge in der Verbrecherkartei nach Zeugenaussagen
    - Krankheitssymptome in Röntgenbildern
    - Luftbildaufnahmen, die einen Flugplatz zeigen
    - Pressefotos von Mitterand und Kohl
    - Texte zum Thema Straßenverkehr
    - Rundfunkdiskussionen zum Für und Wider von Tempo 100
  - als Mustervergleich praktisch nicht formulierbar!

Inhaltsorientierte Suche (2)

- **Vorauswahl**
  - Benutzung der Einteilung in Kategorien (Klassifikation) im Schema der Datenbank:
    - verschiedene Relationen bzw. Objektklassen (Angestellte, Insassen, Rennpferde, ...)
  - nur grob
- **Analyse zur Laufzeit**
  - im Hinblick auf eine bestimmte Anfrage
  - Verfahren zur
    - Texterschließung
    - Bildanalyse und -erkennung
    - Spracherkennung
  - auf eine (große) Menge von Medienobjekten anwenden (möglichst parallel)
  - derzeit noch viel zu aufwändig
- **Browsing**
  - schnelles "Blättern" durch die Gesamtmenge
  - Abwälzen der Auswahlentscheidung auf den Benutzer;
    nur bei kleinen Mengen von Medienobjekten zumutbar

Inhaltsorientierte Suche (3)

- **deshalb: Inhaltserschließung im Voraus**
  - Erzeugung einer Inhaltsangabe
    - automatisch, soweit möglich (automatic abstracting)
    - durch Benutzer:
      Autor erstellt Kurzfassung und ordnet Schlagworte zu
- **Aufgabe des DBS dann:**
  - Speicherung der Inhaltsangabe zusammen mit den Medienobjekten
    (als integraler Bestandteil: Beschreibungsdaten, s. oben)
  - Benutzung bei der Suche
- **verschiedene Typen von Inhaltsangaben:**
  1. **formatierte Daten**
     - die dargestellten oder behandelten Gegenstände
       durch Tupel in Relationen beschreiben (s. oben : Rennpferde)
       - effiziente Suche
       - "unerwünschte" Entity-Typen (Relationen): Sturm, Nacht, Schnee
       - nicht mächtig genug

Inhaltsorientierte Suche (4)

2. **Schlagworte**
   - lange Erfahrung (Bibliotheken, Information Retrieval)
   - leicht erstellbar (auch automatisch)
   - nicht mächtig genug: Zusammenhänge, Abläufe, Kausalitäten usw. (komplexer Informationsgehalt etwa bei Bildern) nur schwer darstellbar
3. **Wissensrepräsentationstechniken**

   (Prädikate, Semantische Netze, Frames, Scripts, Conceptual Graphs, ...)
   - Suchtechniken verfügbar
   - hinreichend mächtig
   - für Menschen relativ schwer zu erstellen (verlangt einen "Wissensingenieur")
4. **freier Text**
   - für Menschen einfach zu erstellen
   - Volltextsuche? Abhängigkeit von Formulierungen!
5. **Schlagzeilen, Telegrammstil** (engl. Captions)
   - auch noch einfach zu erstellen
   - intern umsetzbar in formale Darstellungsarten, z. B. in Wissensrepräsentation

3.1 Text

- **Rohdaten:**
  - variabel lange Folge von Zeichen (Buchstaben, Ziffern, Sonderzeichen), abdruckbar
- **Registrierungsdaten:**
  - **Zeichenvorrat** (ISO Latin-1, Unicode, Kanji)
  - Zahl der Bits pro Zeichen (meist 8, d. h. Bytes)
  - Codierung der Zeichen (ASCII, EBCDIC)
  - Länge oder Endemarke (0x0)
  - evtl. Zeilenendezeichen (<RET>)
- **Beschreibungsdaten:**
  - Strukturinformation, z. B.

    Kapitel (nr, ueberschrift)

    Absatz (nr, kapitelnr, zeichenposition, laenge)
  - Inhaltsangabe, z. B. Schlüsselworte, Wissensrepräsentation, Abstract

Text (2)

- **Operationen:**
  - müssen auf jede Art von Text anwendbar sein
  - **lesend**:
    - Länge ermitteln (Ergebnis: ganze Zahl)
    - Textstelle ab einer bestimmten Zeichenposition in einer bestimmten Länge (Text)
    - Zeichenposition, an der eine gegebene Textstelle enthalten ist (ganze Zahl)
  - **ändernd**:
    - Anhängen eines anderen Textes (Konkatenation)
    - Ersetzen einer Textstelle ab Zeichenposition in einer bestimmten Länge durch anderen Text
  - Ein- und Ausgabeformat hängt ab von den Typen der verwendeten Programmiersprache (Typkonvertierung)
  - vgl. UNIX-Dateien

Text (3)

- **Vergleichsoperatoren:**
  - schon hier das Problem: wann sind zwei Texte gleich?
  - einfach: Übereinstimmung der Bitmuster
    – reicht nicht aus, zu "mechanisch"
- **der gleiche Text?**
  - Dies ist das Haus
  - _ • • • • • • • • • • • • _ _ • • • _ • • • • • • • _ • • _ • • •
- **der gleiche Text?**
  - "Eduard zu töten scheut Euch, nicht gut ist es."
  - "Eduard zu töten scheut Euch nicht, gut ist es."
  - (Brecht, Leben Eduards des Zweiten von England)
- **der gleiche Text?**
  - The house is red.
  - Das Haus ist rot.
- **codierungsunabhängig – sprachabhängig**

Text (4)

- **Operationen auf Beschreibungsdaten:**
  - Eintragen einer neuen Beschreibung (Inhaltsangabe)
    ggf. Überschreiben der vorhandenen
  - Ergänzen der vorhandenen Beschreibung
  - Ausgeben der Beschreibung
  - Vergleich der Beschreibung mit einem Suchmuster;
    Ergebnis: ja oder nein
  - (in dieser Allgemeinheit für alle Medienobjekte gleich)
- **Subtypen von Text**
  - troff-Text, C-Programm, PostScript, ...
  - kann in den Registrierungsdaten vermerkt werden
  - alle Text-Operationen sind anwendbar;
    es kommen noch spezielle Operationen hinzu:
    wordcount, deroff, spell, lint usw.

Textspeicherung

- **Volltext-Speicherungs- und -Retrievalsysteme**
  - [Groß96a]
- **Document Management Systems (DMS)**
  - heute vielfach mit multimedialen Erweiterungen
    (vor allem gescannte Papierdokumente: "Imaging Systems")
- **grundlegende Unterscheidung:**
  - Speicherung der Texte selbst
  - nur Indexierung
- **Attribute (Registrierungsdaten):**
  - Autor/Besitzer, Datum, Quellsystem, ...
- **tar, lharc, zoo, StuffIt, Compact Pro, WinZIP, ...**
- **Komprimierung:**
  - meist Lempel-Ziv-Welch [Ziv78a, Welc84a]

3.2 Suche nach und in Texten

- **hierarchisch (Browsing) und Mustererkennung (grep)**
- **Stichworte – kommen selbst im Text vor**
  (Schlagworte – werden von außen zugeordnet)
- **verfeinerte Verfahren:**
  - Stemming: Rückführung von Stichworten auf Grund- und Stammformen (Bäume -> Baum)
  - Phrase Search: Suche nach Mehrwortgruppen
  - Proximity: Abstandssuche
  - Ähnlichkeitssuche
- **Boolesche Ausdrücke zur Verknüpfung einzelner Suchterme**
- **nicht zu verwechseln mit "Booleschem Retrieval":**
  - Zweiteilung der Menge aller Texte in Ergebnis einer Anfrage und Rest (im Gegensatz zum „Ranking“: sämtliche Texte erhalten einen Wert, der ihre Relevanz in bezug auf die Anfrage darstellt; s. dazu unten)
- **Systeme am Markt:**
  Fulcrum SearchServer, FAUST, LARS II, Topic u.v.a.

Information Retrieval

- **Index: Bibliothekskatalog auf dem Rechner**
- **falls Texte selbst auch auf dem Rechner verfügbar: Volltextsuche (Stichwortsuche) als zusätzliche Option**
- **ansonsten aber Methoden der Bibliothekswissenschaft:**
  - Klassifikation (z. B. dezimal, ACM CRC)
    Nachteil: streng hierarchisch
  - Deskribierung (Indexierung):
    Zuteilung von Schlagworten (sog. Deskriptoren)
    müssen nicht selbst im Text vorkommen
    evtl. noch gewichtet
- **Benutzung eines Thesaurus (= "Wortschatz")**
  - Verzeichnis aller möglichen Schlagworte
  - Synonym-Beziehungen:
    DBS, DB-System, Datenbanksystem,
    Datenbank-System,....
    (ein Repräsentant als Deskriptor verwendet)
  - Oberbegriffs- und Unterbegriffs-Beziehungen

Information Retrieval (2)

- **Erstellung eines Thesaurus aufwändig, viel Fachwissen erforderlich**
- **ebenso: Zuteilung von Deskriptoren subjektiv! und kontextabhängig**
- **Automatic Indexing**
  - Programmsystem zur Deskribierung
  - Eingabe: Text (komplett, Kurzfassung oder nur Titel), Thesaurus
  - Ausgabe: Schlagworte
- **Probleme im Detail:**
  - Wortstamm bilden, Synonym verwenden
  - Umstellungen erkennen:
    - "IC-Entwurf"
    - "der Entwurf dieser winzigkleinen, kaum fingernagelgroßen integrierten Schaltkreise"
    - " .... sind IC's. Deren Entwurf .... "
  - Negationen:
    - " .... soll es hier nicht um IC's gehen, sondern .... "
  - u.v.a.m.

Information Retrieval (3)

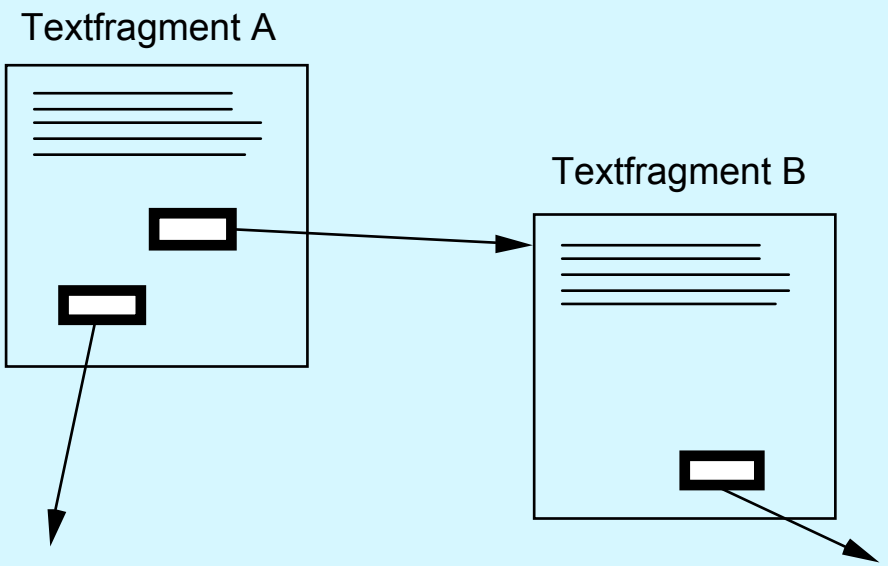
- **Anfrage**
  - besteht ebenfalls aus Schlagworten, ggf. mit Angabe der Wichtigkeit
- **keine Boolesche Entscheidung**
  - Dokument passt zur Anfrage oder nicht
- **sondern graduelle Einschätzung, Maß für die Eignung**
  - "Relevanz" des Dokuments in bezug auf die Anfrage
- **Präsentation des Suchergebnisses**
  - absteigend nach Relevanz geordnet, d. h. das "relevanteste" Dokument zuerst
- **Ranking der gespeicherten Dokumente:**
  - Vektorraummodell:
    Anfrage und alle Dokumente als Vektoren im mehrdimensionalen Raum darstellen, Abstand errechnen und als Relevanzmaß interpretieren
  - Probabilistisches Retrieval:
    Wahrscheinlichkeit der Relevanz schätzen
- **Anmerkung:**
  - Information Retrieval heute nicht nur für Texte,
    sondern gerade auch für multimediale Daten; s. unten

Hypertext

- **Loslösung von der Papierform (wird nachrangig)**
- **primäre maschinelle Organisation von Dokumenten (Bibliotheken, Archive)**
- **Ausgangspunkt:**
  - Memex-System [Bush45a] auf der Basis von Mikrofilm
- **sehr bald Nutzung von Computern (Anfang 60er Jahre)**
  - D. Engelbart: NLS/Augment
  - T. Nelson: Xanadu-Projekt
- **populär geworden durch**
  - Apple's HyperCard auf Macintosh (1987)
  - und vor allem das World-Wide Web (1991)
- **gute Übersicht: [Conk87a]**
- **was ist Hypertext?**
  - Sammlung von Textfragmenten (Artikeln, Notizen, Ausschnitten usw.)
  - systemunterstützte Verbindungen zwischen ihnen (Querverweise, Referenzen, Links)

Hypertext (2)

- **Benutzer folgen den Verbindungen und bauen selbst immer wieder neue auf: "nicht-linearer Text", "information web"**
- **Benutzerschnittstelle:**
  - Fenster auf dem Bildschirm, zeigen jeweils ein Textfragment
  - Verbindungen bzw. Anker graphisch markiert (Ikone oder Fettdruck, Farbe, Unterstreichung)
  - Anklicken mit Maus öffnet neues Fenster mit dem referenzierten Textfragment (verallgemeinertes Blättern)



Hypertext (3)

- **Datenbestand: Netz von Knoten (Textfragmenten), "Hyperdokument"**
- **Fenster: 1:1-Zuordnung zu Knoten**
- **jeder Knoten hat eindeutigen Namen**
- **"link icons" (Referenzsymbole): beliebig viele pro Fenster**
- **verschiedene Verbindungstypen: Beschriftung der Referenzsymbole**
- **einfaches Erzeugen neuer Knoten und neuer Verbindungen (Anmerkungen, Notizen)**
- **Suche**
  - Verbindungen folgen (Blättern)
  - Suche im ganzen Netz nach Zeichenkette, Schlagwort oder Attributwert
  - Graphische Darstellung des Netzes mit "Miniaturen" der Knoten, anklicken

Hypertext (4)

- **Bedeutung eines Browsers:**
  - Datenbestand kann sehr groß werden ("lost in hyperspace")
  - graphische Darstellung des Netzes
  - Protokollierung des Suchwegs („wo war ich?“)
  - Filterung von Knoten (Bildung von Teilnetzen)
  - (Achtung: Begriff wird heute anders verwendet –
    für WWW-Clients, auch mit geringerer Leistung)
- **Hypertext kann betrachtet werden als:**
  - Datenorganisationsmethode (eine Art Datenmodell)
    bis auf die Speicherungsstrukturen hinunter:
    Link ist Referenz (Pointer)
  - Darstellungsmethode (wie Semantische Netze)
    für den Systementwurf, dann Übersetzung in ein implementiertes Datenmodell
  - Benutzermodell, Benutzeroberfläche
    auf darunter liegenden (ggf. sogar relationalen) Strukturen

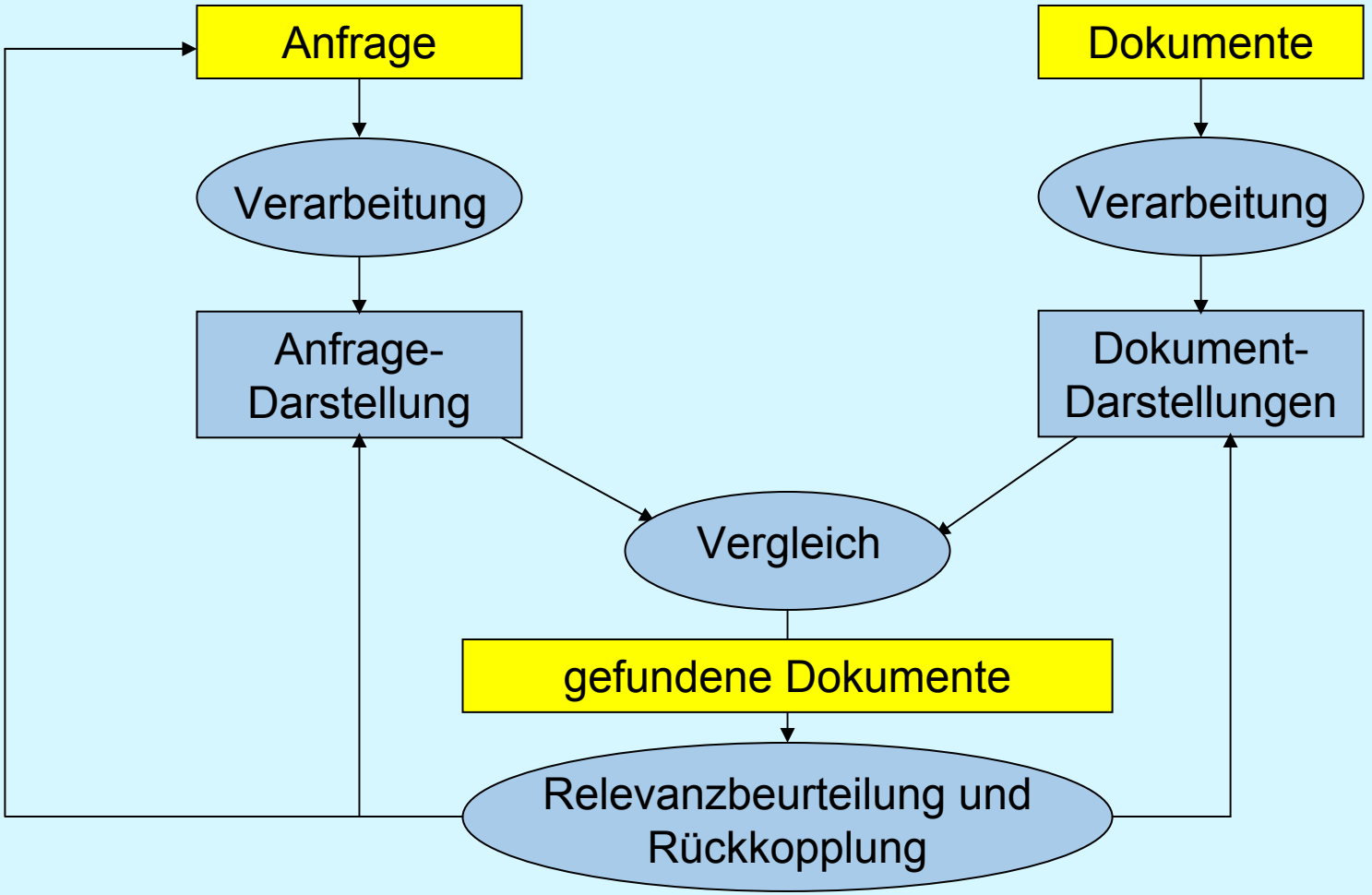
Suche nach Text

- **Information Retrieval (siehe oben)**
- **benötigt für:**
  - große Text-Sammlungen (Bibliotheken)
  - Annotationen von anderen Medien
- **Entwurfsentscheidungen:**
  - Darstellung von Dokumenten und Anfragen
  - Vergleich
- **vier wichtige Retrieval-Modelle:**
  - exakte Übereinstimmung (Boolean)
  - Vektorraum
  - probabilistisch
  - Cluster

Unterschied zwischen IR-Systemen und DBS

- **DBS**
  - homogen strukturierte **Sätze**, jeder mit der gleichen Menge von Attributen, Werte dieser Attribute beschreiben den Satz eindeutig und vollständig
  - **Suche** basiert auf exakter Übereinstimmung zwischen Anfrage und Attributwerten; Ergebnis immer relevant
- **IRS**
  - **Sätze** unstrukturiert, keine Attribute, evtl. beschrieben mit Hilfe von Schlagworten, Deskriptoren oder Indexierungstermen
  - Indexierungsterme:
    - beschreiben Satz (Dokument) nur teilweise und auch nicht eindeutig
    - u. U. große Menge davon mit einem Dokument verbunden
  - **Suche** basiert auf Ähnlichkeit (z. B. teilweiser Übereinstimmung); Ergebnis evtl. nur teilweise relevant

Information-Retrieval-Prozess



Das elementare Boolesche Retrieval-Modell

- **von vielen kommerziellen IR-Systemen verwendet**
- **Textmuster-Such-Systeme**
  - Zeichenketten oder reguläre Ausdrücke
  - alle Dokumente durchsuchen
  - verbreitet für kleinere Dokument-Sammlungen
  - z. B. die grep-Familie unter Unix
- **Boolesche IR-Systeme**
  - Indexierung der Dokumente mittels **Schlagworten**
  - Anfragen ebenfalls Menge von Schlagworten, verknüpft durch logische (Boolesche) Operatoren:
    - OR: Anfrageterme als Synonyme behandelt
    - AND: verbindet Anfrageterme (Schlagworte) zu Phrasen
    - NOT: Einschränkung, üblicherweise zusammen mit AND verwendet

Dateistrukturen

- **flach**
  - ein oder mehrere Dokumente in einer Datei, nicht indexiert
  - Textmuster-Suche (grep, awk, ...)
  - nicht sehr effizient, nur für kleine Dokumentmengen
- **Signaturen**
  - Bitmuster als Dokumentdarstellung (Hash)
  - Anfrage ebenfalls als Bitmuster, dann Vergleich
  - oft zur Grobauswahl, dann noch Textmuster-Suche auf kleinerer Menge zur endgültigen Entscheidung

Invertierte Dateien

- **zu jedem vorkommenden Schlagwort**
  - Liste der Ids sämtlicher Sätze (Dokumente), denen das Schlagwort zugeteilt wurde
- **Eintrag (Zeile):**
  - (Term: Satz-Id, Satz-Id, ...)
- **schneller Zugriff**
  - ausgehend von den Schlagworten in der Anfrage
  - zu den Einträgen
- **Auswertung der logischen Verknüpfungen:**
  - OR: Vereinigung der Listen (ohne Duplikate)
  - AND: Schnitt der Listen
  - NOT: Differenz der Listen

Erweiterte Nutzung invertierter Dateien

- **zwei wichtige Faktoren noch ignoriert:**
  - Reihenfolge der Terme
  - Wichtungen
- **Abstandsoperatoren**
  - (Term i *within sentence* Term j) – Terme sollen im selben Satz vorkommen
  - (Term i *adjacent* Term j) – Terme sollen unmittelbar benachbart vorkommen
- **zusätzliche Informationen in den Einträgen**
  - zu jeder Satz-Id auch noch (Absatz-Nr., Satz-Nr., Wort-Nr.)
    - (eigentlich Liste dieser Angaben – Term kann im Satz mehrfach vorkommen)
  - wird *nach* den Booleschen Verknüpfungen (AND) ausgewertet

Indexierung

- **wie erhält ein Dokument seine Schlagworte?**
  - Dokument-Darstellung erzeugen
- **Stoppworte**
  - Präpositionen und Artikel meist nicht hilfreich bei der Suche – entfernen
  - sprachabhängig
- **Stammbildung (stemming)**
  - Zusammenfassung von Worten, die in unterschiedlichen syntaktischen Formen auftreten, aber vom selben Wort abstammen (Suche, suchen, gesucht, suchte, ...)
  - Indexdatei kleiner, mehr Treffer - aber evtl. auch mehr nicht-relevante
- **Thesaurus**
  - siehe oben

Term-Gewichte

- **Häufigkeit des Auftretens in einem Dokument**
  - über alle Variationen hinweg, also vor Stammbildung
- **Annahme:**
  - häufig vorkommende Terme sind wichtiger für das Dokument
  - Gewicht drückt Übereinstimmung mit Anfrage aus -> Ranking
- **mit in den Einträgen der invertierten Datei speichern**
  - (Term i: (Satz-Id j, Gewicht j), ...)
  - normalisiert auf Intervall [0,1]
- **Auswertung der Booleschen Operatoren:**
  - OR: das höhere Gewicht wird genommen, wenn beide Terme vorkommen
  - AND: das niedrigere Gewicht wird genommen
  - NOT: die Differenz der Gewichte (d. h. das Vorkommen des nicht gewollten Terms in einem Dokument reduziert dessen Gewicht)

Ranking

- **ganze Dokument-Menge**
  - wenn Term in fast allen Dokumenten vorkommt: nicht gut geeignet für Suche, da zu wenig differenzierend
- **"gute" Index-Terme**
  - kommen in wenigen Dokumenten häufig vor, aber kaum in den anderen Dokumenten
  - außer der **Term Frequency** tf_{ij} (Häufigkeit, mit der Term j in Dokument i vorkommt) auch noch **Document Frequency** df_j (Zahl der Dokumente, in denen Term j vorkommt) verwenden
  - Gewicht W_{ij} eines Terms j für ein Dokument i:

    $$W_{ij} = tf_{ij} * \log(N/df_j)$$

    (N = Zahl aller Dokumente)
  - proportional zur Term Frequency und zur invertierten Document Frequency
  - bei $df_j = N$ Gewicht null

Bewertung Boolesches Modell

- **einfach**
- **in vielen kommerzielle Systemen genutzt**
- **Anfrageformulierung**
  - schwierig
  - aber zugleich entscheidend für Ergebnis
- **Gewichte**
  - meist nur in den Dokument-Termen, selten in Anfrage-Termen
  - Anfragen typischerweise kurz

Vektorraum-Retrieval-Modell

- **Annahme:**
  - feste Menge von Termen, die für Dokumente und Anfragen genutzt werden (d. h. Thesaurus)
- **Darstellungen:**
  - Dokument $D_i = (T_{i1}, T_{i2}, ..., T_{ik}, ..., T_{iN})$, T_{jk} Gewicht des Terms k im Dokument i
  - Anfrage $Q_j = (Q_{j1}, Q_{j2}, ..., Q_{jk}, ..., Q_{jN})$, Q_{jk} Gewicht des Terms k in der Anfrage j
  - N Anzahl aller Terme
  - Termgewichte können binär sein (0 oder 1), W_{ij} wie oben oder anders berechnete Gewichte
- **Ähnlichkeit (similarity) zwischen D_i und Q_j:**

$$S(D_i, Q_j) = \sum_{k=1}^{N} T_{ik} * Q_{jk}$$

Vektorraum-Retrieval-Modell (2)

- **Normalisieren**
  - Unterschiede in Dokument- und Anfragelängen berücksichtigen
  - durch Produkt der Längen der beiden Vektoren dividieren
- **Hauptprobleme:**
  - Terme ohne Beziehungen zueinander
  - arbeitet nur bei kurzen Dokumenten und Anfragen ordentlich

Relevanzrückkopplung (Relevance Feedback)

- **Benutzer kennzeichnet gefundene Dokumente als relevant oder irrelevant**
- **Anfragemodifikation**
  - Terme, die in als relevant gekennzeichneten Dokumenten vorkommen, werden hinzugefügt, oder ihr Gewicht wird erhöht
  - Terme, die in als irrelevant gekennzeichneten Dokumenten vorkommen, werden entfernt, oder ihr Gewicht wird reduziert

$$Q^{(i+1)} = Q^{(i)} + \alpha \sum_{Di \in Rel} D^i - \beta \sum_{Di \in NonRel} D^i$$

  - erhöht die Qualität der Ergebnisse
  - allerdings profitiert nur der aktuelle Benutzer

Relevanzrückkopplung (2)

- **Dokumentmodifikation**
  - Terme in der Anfrage, die in einem als relevant eingestuften Dokument **nicht vorkommen**, werden dem Dokument zugeteilt mit einem **initialen Gewicht**
  - Terme in der Anfrage, die auch in einem als relevant eingestuften Dokument **vorkommen**, erhalten in diesem Dokument ein um einen bestimmten Betrag **erhöhtes Gewicht**
  - Terme, die in der Anfrage **nicht vorkommen**, aber in einem als relevant eingestuften Dokument, erhalten ein etwas **geringeres Gewicht** (das Dokument wurde schließlich auch ohne sie gefunden)
  - wirkt sich positiv aus, wenn anschließend ähnliche Anfragen gestellt werden; kritisch allerdings, wenn ganz andere Anfragen gestellt werden

Probabilistisches Retrieval-Modell

- **vier Parameter:**
  - P(rel) – Wahrscheinlichkeit, dass ein Dokument relevant ist
  - P(nonrel) – Wahrscheinlichkeit, dass ein Dokument nicht relevant ist
  - a1 – Kosten, die mit der Rückgabe eines nichtrelevanten Dokuments verbunden sind
  - a2 – Kosten, die mit der Auslassung eines relevanten Dokuments verbunden sind
- **Entscheidung für Dokument als Kostenminimierung**
  - Dokument wird in die Ergebnismenge aufgenommen, wenn

$$a2\ P(rel) \geq a1\ P(nonrel)$$

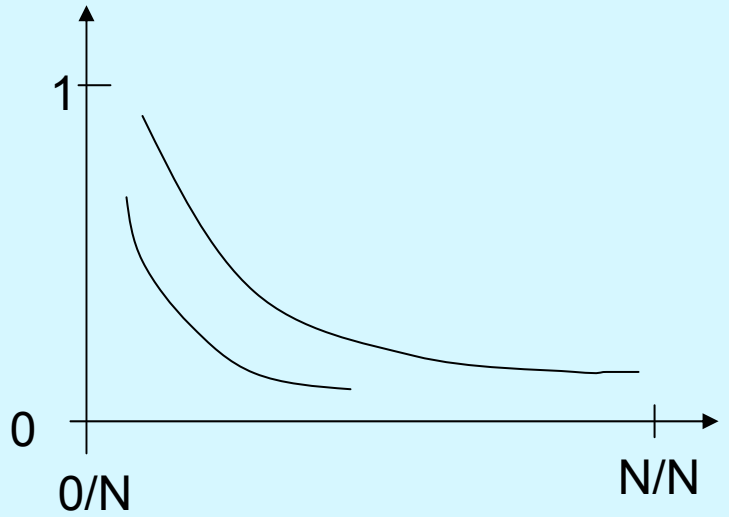
- **Hauptaufgabe:**
  - Abschätzung von P(rel) und P(nonrel) – siehe Literatur

Bewertung von Retrieval-Ergebnissen

- **Antwortzeit**
  - klar
- **Ausbeute (recall)**
  - Anteil der gelieferten relevanten Dokumente an **allen relevanten** Dokumenten im System
  - schwierig zu bestimmen
- **Präzision (precision)**
  - Anteil der gelieferten relevanten Dokumente an den **überhaupt gelieferten**
- **hängen zusammen:**
  - je größer die Ausbeute (immer mehr Dokumente abrufen, mit denen auch die relevanten kommen), desto kleiner die Präzision
  - System muss also Ausgleich finden

Bewertung von Retrieval-Ergebnissen (2)

- **Präzisions-Ausbeute-Graph:**



- Steigerung der Ausbeute-Werte von einem initialen Wert an
- für viele Anfragen erstellen
- viele Anwender heute wollen Präzision – die Wissenschaft und das Patentamt wollen Ausbeute

Bewertung von Retrieval-Ergebnissen (3)

- **Beispiel:**
  - Ergebnismenge: R, R, I, I, R, R, I, I, R, I

lauf. Nr.	R/I	Ausbeute	Präzision
1	R	1/N	1/1
2	R	2/N	2/2
3	I	2/N	2/3
4	I	2/N	2/4
5	R	3/N	3/5
6	R	4/N	4/6
7	I	4/N	4/7
8	I	4/N	4/8
9	R	5/N	5/9
10	I	5/N	5/10

Anwendung auf IR-Techniken

- **verschiedene Studien – einige Ergebnisse:**
  - automatisches Indexieren ist so gut wie manuelles, aber die besten Ergebnisse erzielt eine Kombination von beidem
  - bei gleichartigen Anfragen ist Ähnlichkeitssuche besser als exakte Übereinstimmung (Boolesches Retrieval)
  - das probabilistische Modell und das Vektorraummodell erreichen ungefähr die gleiche Qualität
  - falls nicht alle relevanten Dokumente gleich beim ersten Suchen gefunden werden, erreicht man mit Relevanzrückkopplung eine Verbesserung
  - bei Anfrageformulierung wie Relevanzrückkopplung liefert umfangreiche Eingabe durch den Anwender bessere Ergebnisse als knappe
  - Anwendungsbezug und Benutzerprofile nützen sehr viel