1. Übung zur Vorlesung "Internet-Suchmaschinen" im Sommersemester 2009

– mit Lösungsvorschlägen –

Prof. Dr. Gerd Stumme, M.Sc. Wi-Inf. Beate Krause

22. April 2009

1 Information Retrieval – Grundlagen

1. Was unterscheidet Information Retrieval von der Suche in Datenbanken?
   - Strukturierte Daten
   - Genau spezifizierte Anfrage
   - Exakt definierte Resultate
   - Deterministisches Modell
   - Komplexe Anfragesprache
2. Nennen Sie jeweils drei Beispiele für Web-IR Anwendungen und herkömmliche (ruhig auch digitale) IR Anwendungen.
   - Suchmaschinen (Google, Ask, MSN), Web-Kataloge (Yahoo, Web.de), Nachrichtendienste (BBC World Services, Yahoo Nachrichten)
   - Digitale Bibliotheken (CiteSeer, Uni-Bibliothek Kassel), Bildarchive, Desktopsuche
3. Grenzen Sie Web-IR Anwendungen von den anderen Anwendungen ab: Welche Besonderheiten weist die Web-Suche auf?
   - Größe des Korpus
   - Ständige Veränderung des Korpus, Update-Problematik
   - Semistrukturierte Daten, Layoutinformation (Überschriften, etc.)
   - Metadaten verfügbar
   - Ausnutzen der Linkstruktur
4. Geben Sie Beispiele für den unterschiedlichen Gebrauch der Begriffe "Worte", "Wörter". Zwiebelfischzitate
   - Beim Scrabble legt man Wörter.
   - Wer sprichwörtlich große Worte macht, der spuckt nur große Töne.

- “Ich bin ein Berliner”, “Wer zu spät kommt, den bestraft das Leben” sind große Worte berühmter Politiker.
- Ein Satz besteht aus mehreren Wörtern.

2 Relevanz

1. Verschiedene Benutzer suchen via Google nach “Titanic” und erhalten folgende Seite (`www.titanic-online.com`):

   Werden die Benutzer diese Seite relevant finden? Diskutieren Sie drei verschiedene Suchziele und mögliche Beurteilungen durch die Benutzer.

   Benutzer 1 wollte sich über die Geschichte und das Schicksal der Titanic informieren. Er findet die Seite relevant, weil sie genau diese Art von Inhalten bietet.

   Benutzerin 2 interessiert sich für den Film von James Cameron und wollte Filmkritiken dazu lesen. Trotzdem findet sie die Seite relevant, da sie interessante Hintergrundinformationen zum Film bietet.

   Benutzer 3 sieht das anders. Er wollte nur wissen, ob Victor Garber in dem Film mitspielte oder nicht. Er findet die Seite irrelevant.

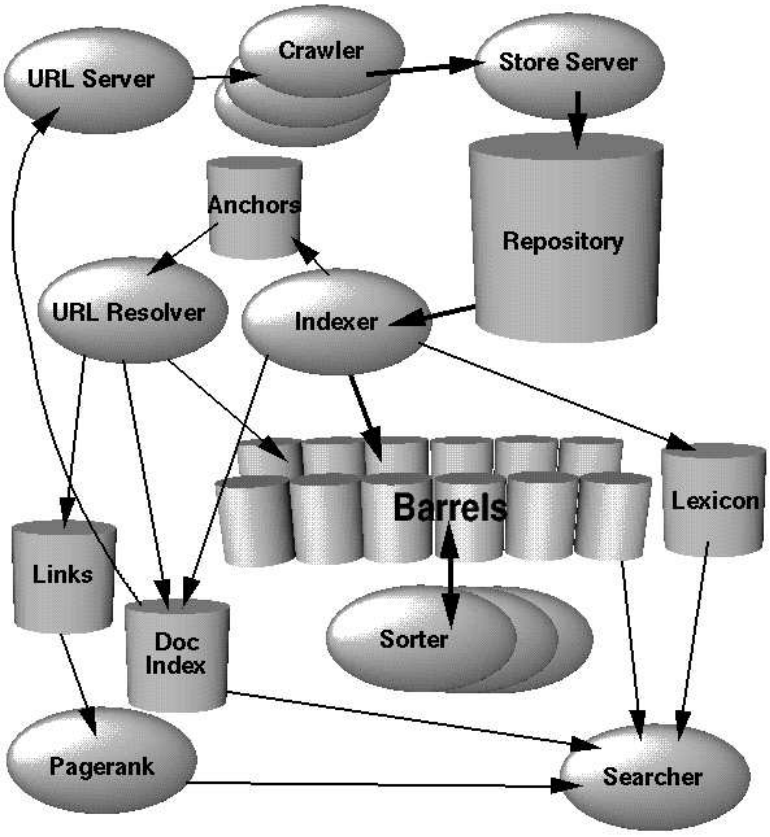
   Benutzerin 4 wollte sich über den Titanic-Algorithmus zur Berechnung von Iceberg-Begriffsverbänden informieren. Für sie ist das Resultat nicht relevant.

2. Sammeln Sie Ideen, welche Informationen rund um Webseiten die Relevanz einer Webseite beeinflussen könnte.
   - Linkstruktur (z.B. Verlinkung durch andere Seiten)
   - Keywords (Bag of words model)
   - Unterscheidung, an welcher Stelle im Dokument die Suchbegriffe erscheinen
   - Unterscheidung, wie weit im Dokument zwei Suchbegriffe auseinander liegen
   - Metatags

- Anzahl der Klicks auf die entsprechende Webseite im Logfile
- Analyse der Suchgeschichte des Anfragenden
- Zuverlässigkeit der Quelle

3 Suchmaschinenarchitektur

1. Auf der folgenden Abbildung sehen Sie eine grobe Skizze, die Google's Architektur aus dem Jahre 1998 abbildet. Diskutieren Sie die Aufgaben der einzelnen Komponenten. Als Hilfe kann die Architektur auf der Vorlesungsfolie S. 28 dienen.



Die Architektur wird in "The Anatomy of a Large-Scale Hypertextual Web Search Engine" von Sergey Brin und Lawrence Page (1998) beschrieben.

- Crawler: Laden Webseiten aus dem WWW. Diese werden im "Store Server" komprimiert und im "Repository" gespeichert. Die URLs die die Crawler herunterladen werden vom URL Server geliefert.
- Indexer: Liest das Repository, und zerlegt die Wörter in den Dokumenten. Aus den Dokumenten werden mehrere Informationen gewonnen.

  - Barrels: Enthalten Informationen über Wörter, deren Positionen in einem Dokument, die Größe der Schrift
  - Anchors: Enthält die Links, die vom Indexer aus den Dokumenten extrahiert werden
  - DocIndex: Enthält Informationen über Dokumente, z.B. einen Zeiger auf das Repository, Checksummen, Statistiken, Dokumenteninfo (z.B. Titel) oder Zeiger auf URL Liste
- URL Resolver: Liest die Anchor Datei und erstellt Links zwischen Dokumenten ("Links"). Diese werden für die PageRank Berechnung (das eigentliche Ranking) benötigt.
- Lexicon: Enthält Ids von Wörtern, sowie die zugehörigen Dokumentnummer, e.g. in welchen Dokumenten tauchen die Wörter auf.
- Searcher: Web Interface. Dieses nimmt die Informationen aus PageRank, Lexikon und DokumentenIndex und liefert eine Liste von Suchergebnissen basierend auf einer Anfrage.

4 Boolesches Retrieval

Betrachten Sie folgenden Text. Jede Zeile stelle ein Dokument dar:

D_1 pease porridge hot
D_2 pease porridge cold
D_3 pease porridge in the pot
D_4 nine days old
D_5 some like it hot
D_6 some like it cold
D_7 some like it in the pot
D_8 nine days old

1. Können Sie für jedes Dokument eine Anfrage mit Hilfe von booleschen Operatoren angeben, die genau dieses Dokument zurückliefert? Unter welchen Bedingungen gelingt dies?

   Dies gelingt, wenn kein Dokument Teilmenge eines anderen ist. Dann kann man durch Aufzählung t_1 *AND* $\ldots$ *AND* t_k der Terme $t_i, i = 1 \ldots k$ des Dokumentes das Dokument auswählen. Wenn jedoch ein Dokument Teilmenge eines anderen ist (hier etwa $D_4 = D_8$), geht das nicht.

   Im konkreten Beispiel würden weniger Terme ausreichen, etwa *pease AND hot* für D_1 usw.

2. Welche Wörter in den vorliegenden Dokumenten sind besonders ungeeignet, um bestimmte Dokumente auszuwählen?

   Einige Wörter, hier *pease, porridge, some, like, it*, kommen in vielen Dokumenten vor, so daß sie nicht gut zur Unterscheidung einzelner Dokumente dienen können.

3. Wie geht man in der Regel mit solchen Wörtern um?

   Solche *Stoppwörter* wie z. B. Artikel, Präpositionen, Personalpronomen usw. werden oft in der Vorverarbeitung entfernt.

4. Können Sie sich denken, warum man den Hamlet-Monolog *to be or not to be* mit dieser Anfrage in einfachen Retrieval-Systemen nicht gut findet?

   Die Anfrage besteht nur aus Stoppwörtern. In frühen Suchmaschinen bekam man daher keine Ergebnisse. Heutige Suchmaschinen versuchen z. B. automatisch eine Suche nach der genauen Phrase und liefern relevante Ergebnisse.

5 Grundlegendes zu den Praxisübungen

1. Die Webseite zur Übung befindet sich unter `http://www.kde.cs.uni-kassel.de/lehre/ss2009/IR/uebungen`. Dort liegt der Programmcode und ein Textkorpus `texte.zip`, der in den Übungen zu Grunde gelegt wird.
2. Machen Sie sich – soweit nicht schon geschehen – mit der Java-API-Dokumentation und einer Java-Entwicklungsumgebung vertraut. Wir empfehlen die Benutzung von Eclipse. (`http://www.eclipse.org`).
3. Zur Orientierung, ob Ihr Programm auch die Anforderungen der Aufgabe erfüllt, wird zu jeder Aufgabe eine Testklasse des Java-Frameworks jUnit mitgeliefert. Um diese auszuführen, müssen Sie das JAR-Archiv junit.jar in den CLASSPATH mit aufnehmen. Das Framework ist unter `http://sourceforge.net/projects/junit/` erhältlich.

6 Praxisübung – User Interface (Abgabe: 05.05.2009 bis 14:00 Uhr)

Im Laufe der Zeit soll eine eigene Suchmaschine entwickelt werden. Die einzelnen Komponenten dafür werden in den Praxisübungen erarbeitet.

Als anfänglicher Korpus wird der Textkorpus `texte.zip` der Webseite genutzt. Erstellen Sie die Klassen, um Texte in ein Bag-of-Words-Modell einzulesen und darauf (einfache) boolesche Anfragen nach Termen zu ermöglichen.

1. Implementieren Sie das Interface `Document`, welches ein Dokument repräsentiert. Ein Document zählt, welcher Term wie oft vorkommt und kann seinen Inhalt aus einem InputStream einlesen. Sie können davon ausgehen, daß der Text schon vorverarbeitet vorliegt, also ohne Großschreibung, Satzzeichen usw.
2. Implementieren Sie ebenso das Interface `Corpus`, das eine Sammlung von `Document` repräsentiert.
3. Überprüfen Sie Ihre Implementierung anhand des Programms `BooleanTest`.

- Die Anfrage `corpus.getDocumentsContainingAll`("cocoa", "shipment") sollte die Nummern der Dokumente 1, 5258, 8961 und 13462 liefern.
- Die Anfrage `corpus.getDocumentsContainingAny`("alternative", "daily") sollte die Nummern der Dokumente 49, 2310, 5258, 6657, 12179, 12772, 12924, 13462 liefern.

Zur Visualisierung soll auf der Basis von Apache Tomcat mit Hilfe von Java-Servlets oder Java Server Pages eine Web-Schnittstelle entwickelt werden, die folgende Funktionalität anbietet:

1. Anfrageformular, in das der Benutzer zu suchende Terme eingeben kann.
2. Ergebnisliste, die die Treffer für die Anfragen angibt (provisorisch erst einmal die Dokumentennummern).