3. Übung zur Vorlesung "Internet-Suchmaschinen" im Wintersemester 2007/2008 – mit Musterlösungen –

Dr. Andreas Hotho, Prof. Dr. Gerd Stumme, Wi.Inf. Beate Krause

28. November 2007

1 Phrasensuche

Stellen Sie sich vor, daß unsere Suchmaschine auch nach Phrasen suchen können soll, also z. B. nach "knowledge management" als anstatt bloß nach Dokumenten, die "knowledge" und "management" enthalten.

1. Skizzieren Sie einen Algorithmus zur effizienten Phrasensuche auf invertierten Indizes!

   Sei die Phrase $t_1 t_2 \dots t_k$ gesucht.

   - für jeden Term t_i, erhalte nach Dokumenten und Positionen sortierte Liste $L_i = (d_{i,1}, p_{i,1}), (d_{i,2}, p_{i,2}), \dots$ von Dokumenten und Positionen mit dem normalen Algorithmus
   - *// Für jede Liste einen Zeiger*
     initialisiere Zeiger $k_i, i = 1, \dots, k$, mit 1
   - for $k_1 = 1, \dots, |L_1|$
     - Treffer = true
     - *// erster Term ist Referenz, jetzt alle weiteren prüfen*
       for $j = 2, \dots, k$
       - *// der nächste Term muß hinter dem vorigen kommen*
         führe Zeiger k_j weiter, bis $d_{k_j} \geq d_{k_{j-1}}$ oder $(\mathrm{d_{k_j}} = \mathrm{d_{k_{j-1}}}$ und $(\mathrm{p_{k_j}} \geq \mathrm{p_{k_{j-1}}} + 1)$
       - *//Treffer nur dann, wenn wir genau eine Position weiter sind*
         wenn $d_{k_j} \neq d_{k_{j-1}}$ oder $p_{k_{j-1}} \neq p_{k_j} + 1$: Treffer = false; break
     - end
     - wenn Treffer: $p_{i,1}$ ausgeben
   - end

2. Müssen Sie dazu die Datenstruktur erweitern? Was wurde bisher nicht berücksichtigt?

   Um den o. g. Algorithmus zu realisieren, muß man zusätzlich zur bisher vorhandenen Information in den Posting-Listen auch noch abspeichern, an welchen Positionen der Term im jeweiligen Dokument vorkommt.

2 String-Suche: Boyer-Moore

1. Finden Sie mit Hilfe des Boyer-Moore-Algorithmus den Substring "banana" in den folgenden Zeichenketten.

   `http://www.buonissimo.org/ricette/131_ananasconcremadibanana.asp`

   `banabasananabanabanasabananananannasana`

   `http://www.buonissimo.org/ricette/131_ananasconcremadibanana.asp`

   - Bestimmen Sie dazu (durch scharfes Hinsehen ;-) zuerst die Shifttabelle! Die Shifttabelle ist 662641.
   - Notieren Sie jeweils für die verschiedenen Strings, welche der beiden Strategien Sie wie oft weiterbringt. Sie können mit folgendem Beispiel weitermachen:

     ```
     http://www.buonissimo.org/ricette/131_ananasconcremadibanana.asp
     banana..................................................
     Mismatch bei j=6; D[6] = 1, 6 - last[/] = 6 O
     ......banana............................................
     Mismatch bei j=6; D[6] = 1, 6 - last[b] = 5 O
     ...........banana.......................................
     Mismatch bei j=6; D[6] = 1, 6 - last[s] = 6 O
     ```

     usw. `S` steht für die Suffix-, `O` für die Occurrence-Heuristik.

     Weiterführung des ersten Beispiels:

     ```
     ................banana....................................
     Mismatch bei j=6; D[6] = 1, 6 - last[o] = 6 O
     ......................banana..............................
     Mismatch bei j=6; D[6] = 1, 6 - last[c] = 6 O
     ............................banana........................
     Mismatch bei j=6; D[6] = 1, 6 - last[1] = 6 O
     ................................banana....................
     Mismatch bei j=3: D[3] = 2, 3 - last[_] = 3 O
     ....................................banana................
     Mismatch bei j=6; D[6] = 1, 6 - last[s] = 6 O
     ........................................banana............
     Mismatch bei j=6; D[6] = 1, 6 - last[e] = 6 O
     ............................................banana........
     ```

```
Mismatch bei j=5: D[5] = 4, 5 - last[b] = 4 SO
.....................................................banana....
Treffer!

banabasananabanabanasabananananannasana
banana
Mismatch bei j=5; D[5] = 4, 5 - last[b] = 5 - 1 = 4 SO
....banana.........................
Mismatch bei j=3; D[3] = 2, 3 - last[s] = 3 - 0 = 3 O
.......banana......................
Mismatch bei j=6; D[6] = 1, 6 - last[b] = 5 O
............banana..................
Mismatch bei j=5; D[5] = 4, 5 - last[b] = 5 - 1 = 4 SO
..............banana..............
Mismatch bei j=5; D[5] = 4, 5 - last[s] = 5 - 0 = 5 O
..................banana..........
Mismatch bei j=6; D[6] = 1, 6 - last[n] = 6 - 5 = 1 SO
....................banana........
Treffer!

http://www.bibsonomy.org/tag/ananas+banana+mango
banana
Mismatch bei j = 6; D[6] = 1, 6 - last[/] = 6 O
......banana.................................
Mismatch bei j = 6; D[6] = 1, 6 - last[b] = 5 O
..........banana.............................
Mismatch bei j = 6; D[6] = 1, 6 - last[n] = 1 SO
............banana...........................
Mismatch bei j = 6; D[6] = 1, 6 - last[o] = 6 O
..............banana.........................
Mismatch bei j = 6; D[6] = 1, 6 - last[g] = 6 O
......................banana.................
Mismatch bei j = 5; D[5] = 4, 5 - last[/] = 5 O
........................banana.............
Mismatch bei j = 6; D[6] = 1, 6 - last[s] = 6 O
................................banana.......
Mismatch bei j = 6; D[6] = 1, 6 - last[n] = 1 SO
..................................banana......
Treffer!
```

3 String-Suche: Reguläre Ausdrücke

Geben Sie reguläre Ausdrücke an für

- Eine deutsche Postleitzahl, möglicherweise mit Länderprefix “D-”.

`(D-)?\d{5}`

- Eine deutsche Postleitzahl mit Ortsnamen, evtl. mit Bindestrich, z. B. “34117 Kassel”, “D-34132 Kassel-Oberzwehren”

  `(D-)?\d{5}\s+\w+(-\w+)?`

- Eine Kasseler Straßenadresse. Diese besteht aus einem Präfix “Königs”, “Kaiser”, “Wilhelms” oder “Karls” mit einem Suffix “platz”, “straße”, “tor”, “allee” und einer ein- bis dreistelligen Hausnummer. Eventuell steht vor dem Straßennamen noch eines der Adjektive “Obere”, “Unterer”, “Großes”, “Kleines” (in einem beliebigen Genus).

  `((Große|Obere|Untere|Kleine)(r|s)?\s+)?`

  `(Königs|Kaiser|Wilhelms|Karls)(straße|platz|tor|allee)\s+\d{1,3}`

- Spezifizieren Sie die Abhängigkeit des Genus des Adjektivs vom Suffix im regulären Ausdruck, oder begründen Sie, warum das nicht möglich ist.

  Grundsätzlich ist es in regulären Ausdrücken nicht möglich, dass getrennte Bestandteile des Ausdrucks sich beeinflussen (→ Pumping Lemma!). Man bräuchte hierfür kontextfreie Grammatiken.

  In diesem Fall ist dies es allerdings möglich, durch Aufzählung aller Varianten im regulären Ausdruck sicherzustellen, dass der Genus passt.

4 Relevance Feedback, Anfrageerweiterung

1. Nehmen Sie an, ein Benutzer einer Suchmaschine hat die folgende Anfrage formuliert: *free music free downloads top free music*. Der Benutzer schaut sich zwei Dokumente, d_1 und d_2 genauer an. Er beurteilt d_1 mit dem Inhalt *music, free, software, free, music* relevant, und d_2 mit dem Inhalt *free, thrills, downloads* als nicht relevant. Nehmen Sie an, dass die Suchmaschine zur Anfrageerweiterung die Termhäufigkeit benutzt. Benutzen Sie Rocchio Relevance Feedback, wie in der Vorlesung auf Seite 7 angegeben, um einen neuen Anfragevektor zu bestimmen. Rechnen Sie mit $\alpha = 1$, $\beta = 0.75$ und $\gamma = 0.25$.

   Für die originale Anfrage, sowie für die beiden Dokumente lässt sich folgende Dokument-Term-Matrix aufstellen:

	free	music	top	downloads	software	thrills
q	3	2	1	1	0	0
d_1	2	2	0	0	1	0
d_2	1	0	0	1	0	1

Eingesetzt in die Gleichung ergibt dies:

$$\overrightarrow{q_m} = 1 * \begin{pmatrix} 3 \\ 2 \\ 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} + 0,75 * \begin{pmatrix} 2 \\ 2 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} - 0,25 * \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}$$

Also gilt:

$$\overrightarrow{q_m} = \begin{pmatrix} 4,25 \\ 3,5 \\ 1 \\ 0,75 \\ 0,75 \\ -0,25 \end{pmatrix}$$

Normalerweise werden negative Terme im Rocchio Algorithmus ignoriert.

2. Heutige Web-Suchmaschinen benutzen in der Regel kein klassisches Relevance Feedback oder Anfrageerweiterung, z. B. weil Anwender nicht bereit sind, aufwendige Formulare zur Anfrage auszufüllen.

   Welche Ansätze von Relevance Feedback oder Anfrageveränderungen können in Web-Suchmaschinen heutiger Machart umgesetzt werden, ohne ein kompliziertes Interface zu benutzen?

   - Verfolgen des Sucherfolgs: oft wird nicht der Link auf die Zieldokumente zurückgeliefert, sondern eine Umleitungsseite. Damit bekommt der Suchmaschinenbetreiber eine Rückmeldung darüber, welche Links zu einer gegebenen Anfrage angeklickt wurden oder nicht.
   - Bei erfolglosen Anfragen bietet beispielsweise Google eine Tippfehlerkorrektur an. So wird z. B. “Dirk Nowitzki Würzbrg” nach Rückfrage beim Benutzer umgeschrieben zu “Dirk Nowitzki Würzburg”.
   - Suchmaschinen wie clusty.com gruppieren (clustern) Suchergebnisse und bieten dann gezielte Suche mit erweiterten Suchtermen an.

3. Anfrageerweiterung zielt darauf ab, durch Veränderung der Anfrage mehr relevante Dokumente aus einem gegebenen Korpus zu finden.

   Sehen Sie eine Parallele (oder Symmetrie) zwischen dieser Idee und der Arbeit von Suchmaschinenoptimierern?

Während bei der Anfrageerweiterung die Anfrage so verändert wird, dass sie die gewünschten Dokumente erreicht, geht die Suchmaschinenoptimierung (zumindest die klassischen On-Site-Maßnahmen) genau umgekehrt vor: die Dokumente werden dahingehend verändert, z. B. mit den entsprechenden Suchworten "gespickt", dass sie für möglichst viele der gewünschten Queries relevant sind.

5 Praxisübung; Abgabe am 12.12.2006

Implementieren Sie mit Hilfe eines HTML-Parsers (z. B. JTidy oder NekoHTML) eine Unterklasse `HTMLDocument` von `Document`. Ein `HTMLDocument` soll anstelle eines Textfiles ein HTML-Dokument lesen und den Text extrahieren können. Benutzen Sie dazu den DOM-Parser von JTidy!

Dabei sollen die Sonderfälle vom letzten Übungsblatt (ALT-Tags usw.) berücksichtigt werden.

Bereinigen Sie den Text um nichtalphabetische Zeichen, bauen Sie einen Porter-Stemmer ein, und entfernen Sie die Stopwords aus der unten genannten Liste!

JTidy: `http://jtidy.sourceforge.net`

NekoHTML: `http://people.apache.org/~andyc/neko/doc/html/`

Stemmer: `http://www.tartarus.org/~martin/PorterStemmer/`

Stopwords: `http://www.unine.ch/info/clef/englishST.txt`