**ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ**

Τμήμα Επιστήμης Υπολογιστών

ΗΥ 463 – Συστήματα Ανάκτησης Πληροφοριών


Εαρινό εξάμηνο 2008- 009

# Εργασία: Ranker

# Εισαγωγή

Σκοπός της εργασίας αυτής είναι η επέκταση του **υποσυστήματος διαβάθμισης** (Ranker) της μηχανής αναζήτησης Μίτος. Μελέτες έχουν δείξει ότι το 58% των χρηστών παρατηρούν τα αποτελέσματα της πρώτης σελίδας αποτελεσμάτων ενώ μόνο το 19% συνεχίζει και στη δεύτερη. Εκ τούτου η διαβάθμιση των στοιχείων της απάντησης είναι εξαιρετικά σημαντική και για το λόγο αυτό οι μηχανές αναζήτησης διαβαθμίζουν τα στοιχεία της απάντησης λαμβάνοντας υπόψη και τη *συνάφεια* τους με την επερώτηση αλλά και την *εγκυρότητά* τους όπως αυτή μπορεί να εκτιμηθεί από τις τεχνικές **ανάλυσης συνδέσμων** (Link Analysis Techniques) που έχουν προκύψει τα τελευταία χρόνια.

Ο Μίτος υποστηρίζει τον αλγόριθμο PageRank τον οποίο έχετε ήδη διδαχτεί. Για τις ανάγκες της εργασίας καλείστε (α) να υλοποιήσετε **JUNIT** tests, προκειμένου να αυτοματοποιηθεί ο έλεγχος ορθότητας του ήδη υλοποιημένου Ranker, (β) να επεκτείνετε τον Ranker ώστε να υποστηρίζει και τον αλγόριθμο **HITS** (Hyperlink-Induced Topic Search) και (γ) να υλοποιήσετε μια λειτουργία σύστασης παρόμοιων σελίδων (similar pages) βάσει ανάλυσης συνδέσμων.

# Υπάρχουσα Υλοποίηση

Ο Μίτος υποστηρίζει ήδη τον αλγόριθμο PageRank. Το σχετικό εξάρτημα (Ranker) βλέπει τον Ιστό ως ένα διευθυνόμενο γράφο G=(V,E) όπου οι κόμβοι αντιπροσωπεύουν τις σελίδες που έχουμε ευρετηριάσει και οι ακμές τους συνδέσμους που υπάρχουν μεταξύ τους. Την πληροφορία που χρειάζεται ο Ranker για να κατασκευάσει αυτόν το γράφο τη λαμβάνει διαβάζοντας το αρχείο index το οποίο δημιουργείται από τον Crawler. Η υπάρχουσα υλοποίηση υποστηρίζει τη διαδικασία ανάγνωσης του αρχείου και της δημιουργίας του γράφου Ιστού. Για το σκοπό αυτό έχουν δημιουργηθεί δύο κλάσεις σε Java, η WebGraph και η groogleURL. Η κλάση WebGraph δημιουργεί το γράφο Ιστού ο οποίος αποτελείται από αντικείμενα της κλάσης groogleURL. Τα αντικείμενα groogleURL αναπαριστούν τις σελίδες και κρατούν πληροφορία για το url της σελίδας και των εξωτερικών συνδέσμων αυτής (τις ακμές). Τέλος, στην κλάση WebGraph υλοποιούνται οι μέθοδοι που υπολογίζουν τη διαβάθμιση της κάθε σελίδας σύμφωνα με τον υλοποιημένο γράφο (ήτοι computePageRank()).

Μετά την εκτέλεση του αλγόριθμου υπολογισμού του PageRank, ολόκληρη η κλάση WebGraph αποθηκεύεται σε ένα αρχείο για μελλοντική χρήση. Παρακάτω θα σας παρουσιαστεί ο τρόπος που μπορείτε να διαβάσετε το αρχείο και να το φορτώσετε στη κύρια μνήμη.

# Α) Έλεγχος Ορθότητας Ranker

Καλείστε να υλοποιήσετε JUNIT tests για τον έλεγχο ορθότητας των μεθόδων της κλάσης WebGraph. Τα τεστ θα χρησιμοποιούν αρχεία index (με λίγες εγγραφές) για τα οποία θα έχετε υπολογίσει εκ των προτέρων την αναμενόμενη διάταξη (των σελίδων που θα καταγράφονται στα αρχεία αυτά) τα οποία και θα συγκρίνετε με τα αποτελέσματα που επιστρέφουν οι μέθοδοι της WebGraph. Εάν παρατηρήσετε λάθη στα αποτελέσματα κάποιας μεθόδου, καλείστε να τα διορθώσετε (και να τα καταγράψετε στην γραπτή αναφορά).

Για την εκμάθηση δημιουργίας JUNIT tests, θα σας δοθεί το απαραίτητο υλικό. Περισσότερες πληροφορίες για την υλοποίηση του Ranker υπάρχουν στην:
http://groogle.csd.uoc.gr/apache2-default/index.php/Ranker.
Ένα πρότυπο αρχείο index περιγράφεται στην:
http://groogle.csd.uoc.gr/apache2-default/index.php/Crawler ,
συγκεκριμένα στην ενότητα Έξοδος Ερπυστή - Αρχείο index

# Β) Υλοποίηση HITS

Καλείστε να επεκτείνετε τον Ranker υλοποιώντας την τεχνική διαβάθμισης HITS (Hypertext-Induced Topic Search). Ο αλγόριθμος HITS περιγράφεται λεπτομερώς στη διάλεξη Web Searching II του μαθήματος. Μια σημαντική διαφορά του HITS από τον PageRank είναι ότι εκτελείται κατά τη διάρκεια της αποτίμησης της επερώτησης. Για το λόγο αυτό θα πρέπει να παίρνετε τα αποτελέσματα της επερώτησης του χρήστη από τον Query Evaluator. Για τον έλεγχο ορθότητας της υλοποίησης υλοποιήστε μερικά JUNIT tests.

Ένα παράδειγμα εκτέλεσης του HITS θα ήταν το ακόλουθο:

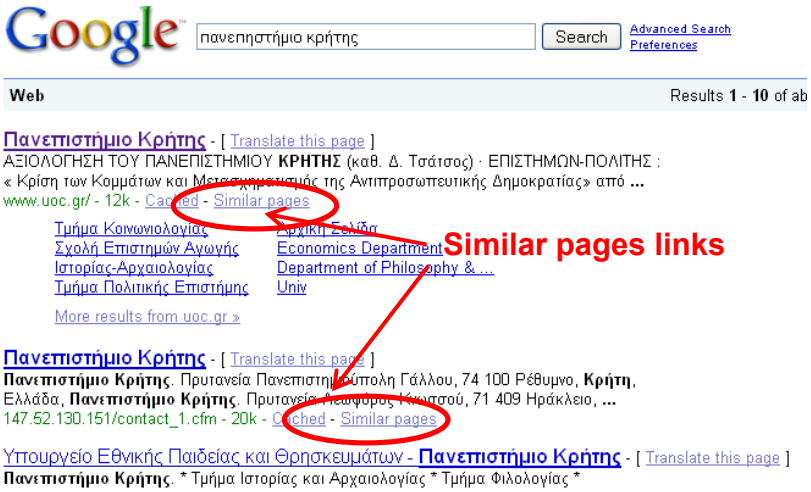
```
WebGraph wGraph = new WebGraph(indexfile);  // initialize the WebGraph with the indexed pages found
                                            // in the file that was made by the Crawler.
    ……..  ……  ……  …….
    ……..  ……  ……  …….      // computes the PageRank for each indexed page

// the wGraph has been stored into the specified file

// now we want to compute the HITS for a specified answer set
WebGraph wg = new WebGraph();                          //create a new WebGraph object
File rankerObject = new File(Resources.RANKER_DIR);
rankerObject.mkdirs();                                 // Create pathname
rankerObject = new File(Resources.RANKER_FILE);        // Open file
wg = wg.read(rankerObject);  //reads the specified file and loads the WebGraph object to the main memory
HashSet<Integer> answerSet  = …  // where answerSet is a list with all ids of the documents that ϵ Ans(q).
wg.setRootBase(answerSet,200);       // defines the root base for HITS as the top 200 elements of Ans(q)
wg.createBase();                     // computes the base S
wg.runHITS();                        // computes and normalizes the scores of the hubs and authorities
int topK = 10;
HashSet<Integer> topAuthorities = wg.getTopAuthorities(topK); // returns the ids of the  topK
                                                              // authorities according the scores
HashSet<Integer> topHubs = wg.getTopHubs(topK);               // returns the ids of the  topK
                                                              // hubs according the scores
```

# Γ) Σύσταση Παρόμοιων Σελίδων (Similar Pages)

Καλείστε να επεκτείνετε τη διεπαφή του Mitos όπως φαίνεται παρακάτω. Εάν ο χρήστης επιλέξει ένα από τα (similar pages)-links θα πρέπει εκτελείται ένας αλγόριθμος που θα επιστρέφει τις σχετικές σελίδες. Ο τρόπος υλοποίησης θα πρέπει να βασίζεται στις τεχνικές ανάλυσης συνδέσμων που είδαμε στο μάθημα. Ένας από αυτούς τους τρόπους βασίζεται στον HITS. Καλείστε να τους υλοποιήσετε και να δοκιμάσετε τέτοιες μεθόδους και να επιλέξετε εκείνη που συμπεριφέρεται καλύτερα (πιο αποτελεσματικά και γρήγορα). Καταγράψτε τα πειράματα που κάνατε και την επιλογή σας στη γραπτή αναφορά.




Για τις ανάγκες ενσωμάτωσης του link “similar pages” στο UI της μηχανής θα πρέπει να ακολουθήσετε τα παρακάτω βήματα:

1. Στο αρχείο inc/jsp-bin/search.jsp στο div της γραμμής 378 θα πρέπει να εισάγετε ένα ακόμα span για τα similar pages το οποίο θα έχει το ακόλουθο url:

   "<%=url%>/index.jsp?related=<%=(result.getDocID().toString())%>”

   Το url αυτό υποδηλώνει ότι πατήθηκε το Similar pages του εγγράφου με id: result.getDocID()

2. Στην αρχή του αρχείου index.jsp θα πρέπει να ελέγχετε αν ο χρήστης έχει επιλέξει να δει τα similar pages κάποιου εγγράφου. Για να το ελέγξετε θα πρέπει να γράψετε το παρακάτω κώδικα:

```
<% String related = request.getParameter("related");
     if (related != null ) {%>
        <jsp:include page = "inc/jsp-bin/related.jsp?related=<%=related%>" flush="true"/>
<%  } %>
```

   Αυτό που ουσιαστικά θα κάνει το index.jsp είναι να ελέγξει αν στο request του χρήστη υπάρχει η πληροφορία για εύρεση των similar pages κάποιου εγγράφου και αν ναι, τότε να

ενσωματώσει το αρχείο related.jsp και να του δώσει ως παράμετρο το id του εγγράφου. Το αρχείο αυτό καλείστε να το κατασκευάσετε στο τρίτο και τελευταίο βήμα.

3. Μέσα στο φάκελο inc/jsp-bin/ θα πρέπει να δημιουργήσετε το αρχείο similarPages.jsp. Στο αρχείο αυτό θα πρέπει να παίρνετε από το request το id του εγγράφου που θέλετε να βρείτε τα similar pages και να καλέσετε τον WebGraph. Για να δημιουργήσετε το WebGraph θα πρέπει να ενσωματώσετε στο αρχείο το παρακάτω κώδικα:

```
<%@ page import = "mitos.ranker.WebGraph" %>

<%    ServletContext context = this.getServletContext();
      WebGraph wg = (WebGraph)context.getAttribute("storedWebGraph");
%>
```

Η κλήση της κατάλληλης μεθόδου του WebGraph θα σας επιστρέψει τα id των similar pages κι έπειτα θα πρέπει να καλέσετε τον Index για να πάρετε όλες τις υπόλοιπες λεπτομέρειες των εγγράφων και να τα τυπώσετε όπως ακριβώς φαίνεται και στο αρχείο search.jsp στις γραμμές 324 – 392. Για να δημιουργήσετε το root set R του HITS πρέπει να φορτώσετε τα ids των κορυφαίων στοιχείων της απάντησης της επερώτησης του χρήστη. Για να τα πάρετε απλώς καλείτε:

```
ArrayList results = (ArrayList)session.getAttribute("storedResults");
```

* Η διαδικασία είναι παρόμοια με αυτήν της υλοποίησης των cached pages, την οποία μπορείτε να δείτε στα jsp's.

# Χρονοδιάγραμμα

Προτείνεται το εξής χρονοδιάγραμμα:

- 1η εβδομάδα (3-10 Απριλίου)
  - Εξοικείωση με τον υπάρχοντα κώδικα του Μίτου και τη δημιουργία JUnit tests μέσω των γνωστών IDE (Eclipse, NetBeans).
- 2η εβδομάδα (27 Απριλίου-3 Μαΐου)
  - Υλοποίηση ελέγχων Junit για τον διαβαθμιστή, υλοποίηση του HITS
- 3η εβδομάδα (4 Μαΐου -10 Μαΐου)
  - Υλοποίηση του similar pages
- 4η εβδομάδα (11 Μαΐου -17 Μαΐου )
  - Πειράματα, δοκιμές και σύνταξη αναφοράς εργασίας (με μετρήσεις)

# Παρατήρηση

Η καλύτερη υλοποίηση θα ενσωματωθεί στη μηχανή Μίτος και η ομάδα θα πάρει 5% bonus (στο βαθμό εργασίας).