Text Based Information Retrieval (H02C8A)
Link-based algorithms

Karl Gyllstrom, Marie-Francine Moens

1 Introduction

For the 4 study points: Choose paper assignment OR programming assignment.
The assignment is due April 28, 2011.
Graded for 1/3 of points.
For the 6 study points: There is only a programming assignment.
The first part of the assignment is due April 28, 2011, the second part May 26, 2011.
Graded for 1/3 of points.

2 Paper assignment description

Paper of ca. 5 p. can be sent by e-mail to: `mailto:karl.gyllstrom+tbir@gmail.com`

The paper answers the following problems with regard to Web information retrieval and more specifically the PageRank algorithm.

1. What is the distribution of PageRank values of the pages as the number of iterations approaches infinity? Are there problems with using no damping factor?

2. What is the trade-off for the damping factor? Consider the effect of the damping factor on variability of PageRank scores, and that on massive datasets, PageRank iterations take much longer to compute.

3. Describe a method to personalize the computation of PageRank? What would be the advantages and disadvantages of the proposed approach?

It is important that you include for each question:

- Description of the problem
- Description of the solution(s), illustrate with examples
- Critical discussion of the solution(s) (refer to the material we have seen in the course, give your personal opinion)
- Conclusions
- References to literature, URLs,...

3 Programming assignment description

Source code (in C, Java or Perl) can be sent by e-mail to: `mailto:karl.gyllstrom+tbir@gmail.com`

Important:

- Add commentary
- Add description of how to run your software
- Add your test examples

In this assignment, you are going to implement some link-based algorithms.

- For the 4 study points variant of the course, you implement only problems 3.1 and 3.2. You may implement them in any way you choose (due April 28, 2011).
- For the 6 study points variant of the course, you answer all problems, and use the MapReduce framework for your implementations. We expect that you submit already solutions for parts 3.1 and 3.2 by April 28, 2011; the answers to the other parts are due May 26, 2011.

3.1 Calculating PageRank

View the dataset provided with this assignment, called *sample-tiny.txt*

The file is tab-delimited and represents a web graph, a list of pages and the outgoing links from the pages. If a node has no out-going links, only the node ID is listed.

Calculate the PageRank vector for this graph, using no *damping factor*. Run the algorithm iteratively and view the PageRank values after each iteration.

1. What is the distribution of PageRank values of the pages as the number of iterations approaches infinity? Are there problems with using no damping factor?

Next, implement the damping factor in PageRank. This time, use the larger graph called *sample-large.txt*. Execute PageRank with various damping values from 0 to 1 in increments of 0.05. Create the following plots:

Calculate the PageRank of this graph.

2. Damping value and standard deviation of PageRank scores.

3. Damping value and number of iterations required for PageRank to converge. We measure convergence by comparing the sum of values in the PageRank vector between two successive iterations. If the difference is less than 0.0001, we say it is converged. i.e., $\sum_{i=0}^{N} PR_j - \sum_{i=0}^{N} PR_{j-1} < 0.0001$.

Answer the following question:

4. What is the trade-off for the damping factor? Consider the effect of the damping factor on variability of PageRank scores, and that on massive datasets, PageRank iterations take much longer to compute.

3.2 Topical PageRank

View the dataset provided with this assignment, called *sample-large.txt*

Topical PageRank is an approach where pages pertaining to a given topic are promoted to receive higher relative PageRank than pages not pertaining to that topic. Let's create a topical PageRank for topic X, including pages 8614504, 10936880, and 8848271. Assume that a random surfer has a 50% of appearing at these pages via a random jump (e.g., via a bookmark).

3.2.1 Simple approach

One simple approach to implementing this is to populate the original PageRank vector with higher values for these topical nodes. Run your implementation of PageRank on an initial PageRank vector where the initial values are 0.02 for normal nodes, and .5 for the topic X nodes listed above (run until convergence).

5. What do you notice about the new PageRank values? Explain the reason behind the new values.

3.2.2 Advanced approach

Rather than modify the original PageRank vector, this time you will modify the damping factor.

Calculate the new PageRank with this topic approach. Describe your approach.

6. What do you notice about the new PageRank values? How are they different from those in the previous approach? Explain.

3.3 Crawling

Due to the large size of the web, it is not practical to index every page within it. Crawlers must be designed intelligently to decide which pages to crawl and which to ignore.

7. Which page should this engine crawl next, and why?

3.4 Spam

Calculate the PageRank for the graph in *sample-large-spam.txt*. We have discovered page 8712790 is a spam page that has embedded a link to itself on various message boards from pages with high PageRank values to boost its own score. One simple way to handle this is to remove it from our graph. However, this is not ideal.

One quality that is common among spam sites is that they link to other spam pages. In practice, spam pages often have a large number of outgoing pages pertaining to spam pages, relative to non-spam pages. Thus, rather than remove this page from the graph, we can use it to discover other pages that are likely to be spam. Here we will define a spam page as one that only (or primarily) links to other spam pages.

8. Design an algorithm that can identify other likely spam pages and submit it.

9. Which other pages are likely to be spam?