Dublin City University Experiments in Connectivity Analysis for TREC-9

Cathal Gurrin & Alan F. Smeaton

School of Computer Applications
Dublin City University
Ireland
cgurrin@compapp.dcu.ie

Abstract

Dublin City University (DCU) took part in the Web Track (small task) in TREC-9. Our experiments were based on evaluating a number of connectivity analysis algorithms that we hoped would produce a marked improvement over a baseline Vector Space model system. Our connectivity experiments are all based on non-iterative post-query algorithms, which rerank a set of documents returned from content-only VSM queries. We feel that in order to implement a real-world system based on connectivity analysis the algorithms must have a low query-time processing overhead, hence our employment of non-iterative algorithms. Our results showed that we were unable to improve over a content-only run with our algorithms. We believe this to be mainly due to the nature of the link structure within the WT10g dataset.

1. Introduction

Dublin City University (DCU) took part in the Web Track (small task) for TREC-9. We wished to continue the experiments we carried out last year on the WT2g TREC-8 dataset. For additional information on our experiments for TREC-8 please see [1]. Our experiments were based on evaluating a number of connectivity analysis algorithms, which we hoped would produce a marked improvement over a baseline Vector Space model system. Our connectivity experiments are all based on non-iterative post-query algorithms, which rerank a set of documents returned from the content-only VSM queries. We feel that in order to implement a real-world system based on connectivity analysis the algorithms must have a low query-time processing overhead, or all required processing must be done at indexing time.

We outline, in this document, details of our content based and linkage-based runs, which were aimed at ranking the most relevant and useful documents at the top of the search results. In effect we are attempting to improve the precision (over the baseline result) of the top documents returned from a search because the vast majority of users only look at the first page of search results. Recall that almost 85% of users only look at the top 10 results. Our approach is based on the assumption that, by implementing a conventional text-based search on the dataset, a subset of documents that are relevant to the topic in question will be generated. The execution of various linkage-based formulae on this small subset of documents will then increase the ranking of the most popular/best documents contained therein.

We do this by developing three algorithms for generating a connectivity score (Sc'_n) for each document in a set of relevant documents. In so doing we must distinguish between the two semantically different types of links to be found on the WWW of today, discarding the less useful types from our processing. This paper assumes certain knowledge about connectivity analysis. For those requiring an introduction we recommend our own TREC-8 article, see [1] or Li's description of the Hyperlink Vector Voting method [2] which ranks a document on the basis of the number of hyperlinks pointing into it (in its immediate neighbourhood) and uses the hyperlink's anchor text as an indication of the semantic content of the target document.

2. System Overview

The WT10g dataset, which was used for the small web task required some pre-processing before we were able to execute queries against it. Each individual web page had to be extracted, given a name based on its document id and saved to disk. As these files were being generated, we extracted a small amount of information from each document, which would later be used to generate intuitive results for each query, thus giving us the ability to chart our progress as we were developing the software. The information we extracted consisted of:

- Document id
- Document Title
- Document Text (< 256 bytes, to the nearest word)

In a fashion similar to our TREC-8 small web task experiments, we used an 'off-the-shelf' search engine to generate content-only results for each query. We opted to use Microsoft Index Server [3] for this purpose, though in retrospect this created more problems than it solved and consequently we are developing our own search tools for use in future experiments. This utilisation of Index Server required the extraction of each individual web page to disk and the construction of large hub files that allowed the Index Server crawler to traverse the graph of web pages from just one root page. While extracting the files to disk, we removed any additional TREC mark-up from the beginning of the document, leaving only the raw HTML of the document. This whole process took about 48 hours to complete using two computers, a PIII server with 104GB-disk space available for data & index storage, and a PIII Workstation for processing the dataset source files.

The Connectivity Data was stored in a Microsoft SQL Server 7 [4] database running on a second PIII workstation, which acted as our Connectivity Server. For a detailed description of the components of a large Connectivity Server see [5]. We maintained separate tables for inLinks and outLinks, consisting of source node id and target node id pairs for each link. The Connectivity Server worked by accepting a document id and returning a set of all inLink and outLink document ids. This approach allowed us to send about 1,000 queries per second to the Connectivity Server, which although slow, proved sufficient for our purposes.

We used the first PIII workstation again to process the queries, and calculate the Connectivity Scores for each document and generate the results. All necessary code was written in JAVA (version 1.2) for Windows NT 4. We networked the three computers together using a dedicated 100 Mbit/s switch. For an overview of the System setup see Figure 1.

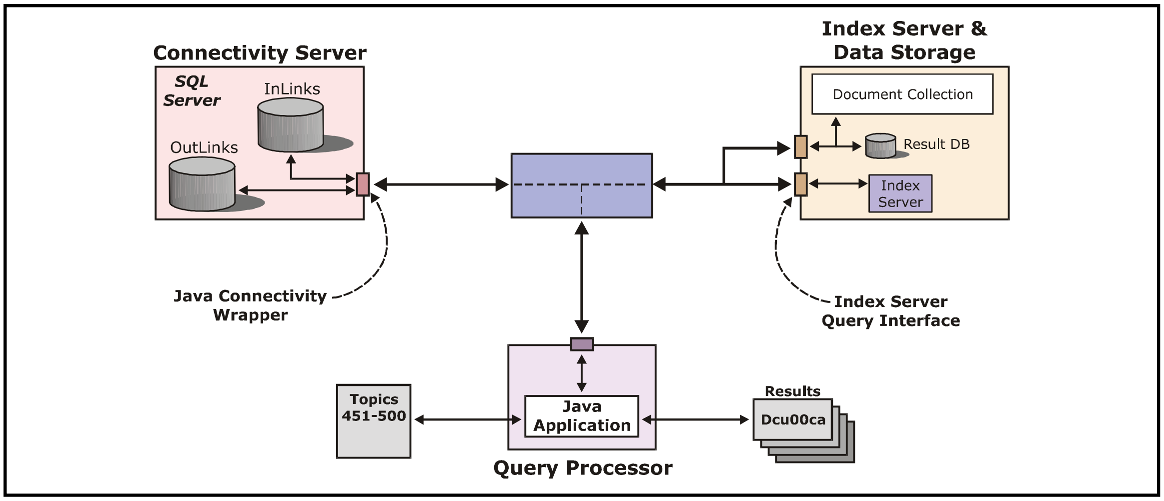

Figure 1: System overview

3. Experiments

Our experiments, as previously mentioned, were devised so that we could evaluate non-iterative approaches to connectivity analysis. We submitted four runs for evaluation purposes, one content-only run (*dcu00ca*), as provided by Index Server and three linkage-based runs (*dcu00la, dcu00lb, dcu00lc*). Only *dcu00lb* did not contribute documents to the assessment pools. The content run was executed before any of the linkage-based runs were executed as the basic output of Index Server was used as input into the three linkage runs.

3.1 Queries

Our queries were manually generated queries, a list of which, are included in the Appendix. In most cases the manual queries are simply the unmodified topic titles. A research student generated the queries after reading the title, description and narrative. In addition we generated a run based on automatic queries taken from the titles of each topic and have included details of where to get our results, as generated by trec_eval, in the Appendix. The automatic query run was not one of our official runs.

3.2 Content Experiment

As was previously mentioned, our experiments consisted of a two-stage process. The first stage was to generate results for a content-only run and the second was the linkage analysis stage, which reranked the set of documents returned by the content-only run. The content-only stage involved sending the query to Microsoft Index Server and extracting the results. We retrieved up to 2,000 result documents from Index Server using the Vector Space query model (there are a number of query models available). These 2,000 documents were ranked by Index Server according to their degree of relevance, but they were not scored, so we had to generate our own scores. If there were not 2,000 relevant documents returned, we processed the number of documents that were available. We assumed that these returned documents represented a large set of documents that could be considered relevant to the query, although this was not always the case as no query had more than 519 relevant documents (from results of the manual-runs). We refer to this ranked set of documents as the 'relevant-set'. In order to generate the content-based run, the top 1,000 results (where that number were available) were extracted for each query and submitted as run (*dcu00ca*) which was our *baseline* result. We generated a score for each document in the relevant set, which provided us with a content-only score for each document; it is this score that will be used in later linkage experiments.

Assuming N is the total number of documents in the result-set and R is the ranked position of that document the formula to generate the score Sc_n for each document in the 'relevant-set' is as follows:

$$Sc_n = \sqrt{\frac{N - R_n}{N}} \qquad for(R_{1\ldots N})$$

Thus far, our *baseline* result has been gathered using a similar approach to Kleinberg's when building a root-set in HITS (see [6]). However Kleinberg only takes the top 200 documents returned from a search engine (in his case AltaVista), which he calls the root-set and expands this root-set to include all neighbouring documents, which is referred to as the expanded-set. However the root-set expansion phase of HITS seems to lead to topic-drift problems as outlined in [7], where the documents that are ranked highest are often generalisations of the topic represented by the query. In order to avoid this problem it was decided to just retain the top 2,000 documents as representing a set of relevant documents and not incorporate the neighbourhood documents to generate an expanded set. We did experiment on using the Kleinberg style expanded-set method but found that even on comparing this to the top 1,000 documents (the basis of our content-only run), we lost on average 5.68 relevant documents per query. See Figure 2 for the total recall figures summed over all queries generated using three alternative methods of generating a set of relevant documents for linkage-based processing. The approach referred to as 'base 200' is simply the top 200 documents returned from a content-only run and the 'expanded set' is generated using the HITS technique mentioned above. The 'total possible' figure is the maximum summed recall over all queries.

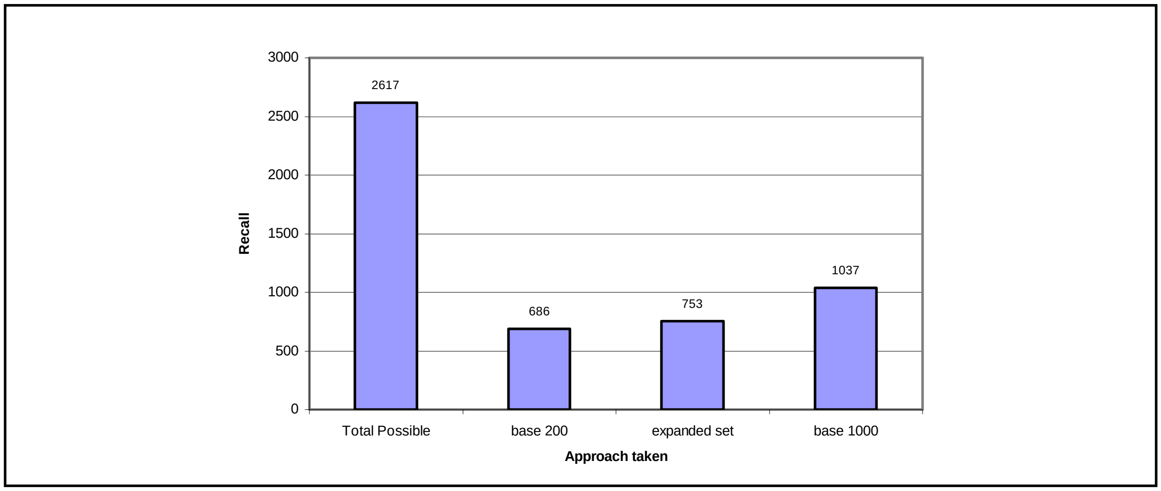

Figure 2: Recall at three different approaches to relevant-set generation

Notwithstanding any improvement in recall attained by using a large base set of documents generated using a content-only method, there are associated drawbacks. Expanding the root-set along the links does produce an expanded-set that naturally contains a high number of interconnected documents whereas selecting the top-ranked 1,000, or 2,000 documents (as we do) produces a set of documents having a much sparser set of interconnections. We may find that the use of an expanded-set is better for connectivity analysis as the expanded-set is guaranteed to have a denser set of links among the documents. This issue requires further research, but which needs to be accomplished on a new dataset.

3.3 Linkage Experiments

Our Linkage experiments were all executed at query time, and based on reranking the relevant-set of documents which were generated during the content-only stage outlined above. We developed three linkage-based approaches for our experiments, all of which adhered to the following requirements:

- Must provide a useful and accurate connectivity score for a document
- Must not require enough processing to adversely affect the performance of the search engine were it implemented in a real world system
- Must be scaleable to realistic sized datasets. We will explain below how we developed for the WT10g sized dataset but all algorithms must be capable of being implemented on more realistic datasets.

The first point is obvious, however the second point is rather more interesting. Looking at Google [8], which is visibly the most successful proponent of connectivity-based web search, it works by calculating a query independent connectivity score for each document in one processing run after all documents have been indexed by the system. This connectivity score, referred to as the PageRank [9] of a document, is then available for use by the system for all queries as part of the ranking formula, with no necessity to do any additional processing at query time.

The other widely known approach is the previously mentioned HITS, which generates linkage scores for documents at query-time. Currently the amount of processing involved in implementing HITS on a real world system would be prohibitive due to the iterative nature of the algorithm. While PageRank requires a similar iterative process, although on a vastly larger document-set (1,326,920,000 web pages as of February 2001), this is only done once per index update, but HITS requires it once per query. The algorithms outlined in this article do not have an iterative process involved and only one run through each document to be reranked is required, which helps our approaches to adhere to the second requirement above.

When discussing hyperlinks, we cannot assume all hyperlinks to be equal in value for our needs. An author writing a WWW document will create semantically different types of hyperlinks between documents, even though HTML supports only one syntactic type of hyperlink. In fact web page authors will most probably not be aware of the significance of the different link types that they are creating. In [10]

Spertus discusses hyperlinks and varieties of hyperlink information, based on information mined from identifying the target of each link. Generally speaking, on the WWW we can separate links into one of two broad types based on their intended function when being created:

- **Structural** links that link separate documents within a particular domain. They exist to aid the user in navigating within a domain, or web site and consequently cannot be seen as a source of authority judgements. See [10] for a more detailed discussion of structural links and their uses.
- **Functional** (content, or outward) links on the other hand link documents in different domains (across web site boundaries). They can be seen to mostly link from a source document to a target document that contains similar and, in the author's opinion, useful information, quite often this information is related to the concept explored in the source document.

When extracting information from hyperlinks on the WWW, we assume two properties inherent in hyperlinks from [7], these are:

- A link between two documents implies that the documents contain related content
- If the documents were authored by different people, then the first author found the second document valuable.

Because of this in the course of our research we are mainly interested in functional links as opposed to structural links and in describing our experiments we always extract only the functional links from the Connectivity Server, unless this is specifically described otherwise.

Our first connectivity analysis run (*dcu00la*) was a modification to basic citation or inLink counting. As mentioned above, we generally assume that the more popular a document is the more functional inLinks that document will have on the WWW. Letting φ be all documents in the domain of document n, the obvious choice for a basic popularity reranking formula would be as follows:

$$P_n = \sum_{m \to n} inLinks_n \qquad for(m \notin \varphi)$$

In this case the popularity of document$_n$ (P_n) is based on the number of functional inLinks into document$_n$. We implemented a similar approach last year as an unofficial run, see [1] for more details. It is notable that a system that implements only one iteration of unweighted HITS is similar to a system which ranks pages based purely on the number of functional inLinks into them.

This year we gave each document in the relevant-set a score based on its rank within the relevant-set and then added the log of the number of (inLinks + 1) multiplied by the original relevance rank of that document so that we could limit the ranking of a document of low relevance which had an unusually large indegree (number of inLinks). In this way a highly relevant document (as decided in the content analysis phase) will receive a higher rank from any inLinks than would a document that is considered less relevant in the content-only phase. This is a very simple approach and is intended as an alternative to basic inLink or citation counting.

Recall from the *dcu00ca* run that Sc_n is the content-only relevance score for document n, which was generated in the content-only stage. We generated Sc'_n as the new score for document n and ranked the documents by this score. Letting $\boldsymbol{\delta}$ be the set of documents generated in the content-only phase and utilising only functional inLinks we have:

$$Sc'_n = Sc_n + Sc_n \times Log\left(\sum_{m \to n} inlink_n + 1\right) \qquad for(m \in \partial)$$

This was calculated for the top 30 documents. During development of the software for the official runs, we had experimented with reranking a variety of cut-offs for the top-ranked documents and kept the value of 30 as we found that this value produced the best results. In fact this run produced the best results of all the linkage-based runs. This would concur with the findings of AT&T in [11] which found that simple indegree ranking performed at least as well as HITS and PageRank style algorithms. This was submitted as run *dcu00la*.

Our second connectivity-based run (*dcu00lc*) attempted to improve a document's score if it was pointed at by another document, which was in itself considered to be relevant to the topic represented by the query. This is a relatively simple approach based on the previously stated assumption that a link normally exists between two documents with related content. Taking this a step further, a link between two documents with tightly related content would be a ‘better’ link, or a link that we could weight higher than others. To this end we increased a document score (proportionally to its current score) if it contained a link from another document that exists in the relevant-set. More precisely we increased the document score by a value which is proportional to the score of the inLink associated document and in a manner similar to the way the PageRank algorithm spreads a document's rank evenly among its outLinks, we limit the score transferred from the inLink document to be proportional to the number of outLinks it has. If effect, if a document has inLinks from a number of relevant documents then its score is increased by an amount proportional to:

- its own relevance score
- the relevance score of the inLink document
- the number of outLinks originating from the inLink document

Recall that all documents have received a score in the content only phase. The formula for calculating each document score is shown below. Let **δ** be the relevant set of documents, therefore:

$$Sc'_n = Sc_n + \left(Sc_n \times \sum_{(m \to n)} \frac{Sc_m}{\sum outlinks_m + 1} \right) \qquad for(m \in \delta)$$

This was calculated for the top 250 documents in the result-set. Once again, this figure can be changed as seen fit, but 250 is our best parameter cut-off point as found when running our experiments.

An advancement on this approach was submitted as our third run (*dcu00lb*), once again calculated for a subset of the top documents in the result-set. This viewed the inLinks to a document as hub documents. Recall from [5] that Kleinberg describes documents in terms of hub documents and authority documents with hub documents acting as a source of links into similar documents while authority documents are seen as sources of authority on a topic and are gathered together into cohesive communities by groups of hub documents. This algorithm (for *dcu00lb*) worked with authority documents in the immediate neighbourhood of the inLinking documents to approximate the identification of documents from within a well-connected community. For example, a relevant document should have a number of hub documents linking into it. Since we assume that hub documents link together groups of related documents, this hub document should be a source of links into other documents that are considered relevant to the query, and consequently they should exist in the relevant-set generated in the content-only stage. Of course we know this not always to be the case and this can be seen by viewing Figure 5 which shows our recall figures for *dcu00ca*, compared to the best, median and worst.

We augment this theory by only allowing outLink documents to be considered if they are part of the relevant-set generated during the content-only phase. This being the case, if a document is considered to be part of this set of relevant documents, it would be more useful to the user making the query. It is, in effect, an attempt to rank highly documents that are related to other relevant documents by sharing a hub document. In this approach we did not exclude hub documents, which were not included in the root set, rather we wanted to reward an inLink from a content-relevant hub document more than from a non-content relevant hub document. To that end, if a hub document is included in the relevant-set, its score (multiplied by a constant α of value 0.45), generates a preliminary score for the hub document before the outLink score is generated. We propagate the relevancy score (multiplied by a constant β of value 0.35) of each relevant authority document linked to the hub document via a functional outLink back to the hub document. The consequence of this is that the hub document now has a score which reflects its own relevance as well as that of its outLink documents. Finally this hub score is divided by the total number of outLinks from it, as was the case with *dcu00lc*. It is this score that is added to the Sc'_n score of the document being reranked. The current values for α and β were best-parameter values that we arrived while running the experiments (we had tried numerous values in the range from 0.0 - 1.0) and we plan to look again at these values on a new web dataset.

Let β be a constant to limit the score being transferred from the target document of the link to the hub document and α be a constant to limit the score being transferred from the hub document to Sc'_n during calculation of the hub document score (HSc'_n), giving:

$$HSc_m = Sc_m \times \alpha + \sum_{m \rightarrow p} Sc_p \times \beta \qquad for(p \in \delta)$$

or if the hub document is not in the relevant-set:

$$HSc_m = \sum_{m \rightarrow p} Sc_p \times \beta \qquad for(p \in \delta)$$

Finally, the Sc'_n is generated from the original score Sc_n and all hub scores HSc_m:

$$Sc'_n = Sc_n + \sum_{m \rightarrow n} \frac{HSc_m}{\sum_{m \rightarrow p} p + 1}$$

The final Sc'_n score is used to rank the documents for the run. We had looked into implementing both HITS and PageRank to compare our experiment's effectiveness against, but felt that the connectivity data was too sparse to be of much benefit in these cases. This was shown to be correct by the results of other participating groups that took these approaches.

4. Results

Of the four approaches we submitted, *dcu00ca*, which is the content-only run, attained highest (or equal) precision across virtually all standard rank positions. Since all other results were dependent on the quality of *dcu00ca* and didn't involve the implementation of any hyperlink-based expansion measures on the result set we were not able to produce any improvement in overall recall at rank 1,000, except in theoretical cases where a document which was not in the top 1,000 and could possibly be reranked into the top 1,000. However due to the lack of useful linkage data which became apparent during development we found that limiting the number of documents reranked would produce relatively better results, so any improvement in recall turned out to be impossible due to these limitations.

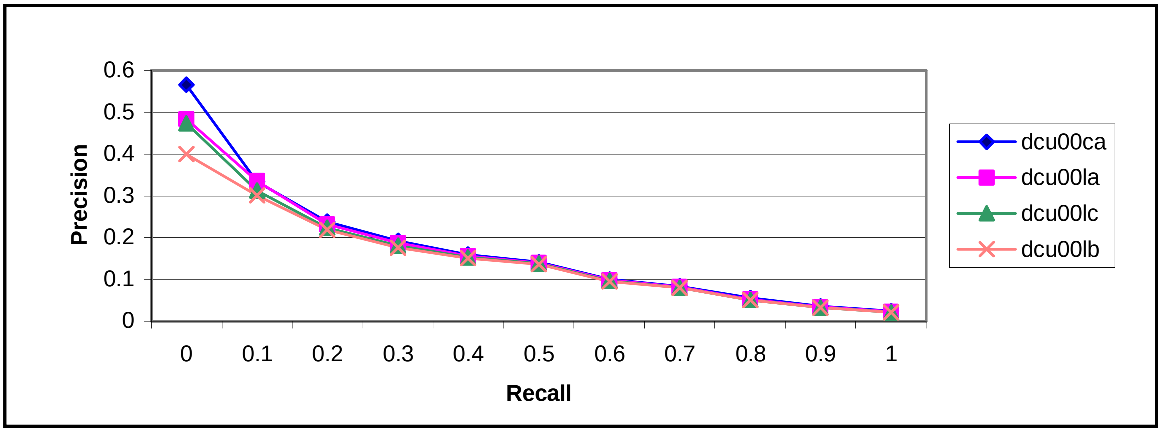

Figure 3: Precision vs. recall graph for all four runs

It would be a worthwhile exercise to see what kind of improvements could be gained by executing our experiments on a new dataset, perhaps one that has been generated with a view to conducting experiments into connectivity analysis. We may find it to be infeasible to limit our connectivity-based processing to only inLink or outLink documents that are considered directly relevant to the original query.

Another side effect of the sparse nature of the connectivity data, there was very little room for improvement over the values already in *dcu00ca* (see Figure 3 for the precision/recall graph of all four runs). With the exception of one query (486) *dcu00ca* performed equally as well, or better than all linkage based approaches. Quite often when *dcu00ca* was found to perform equally as well as the linkage approaches this was due to a lack of linkage data, which left no opportunity to rerank the documents. For details of our average precision results for each of the four runs as well as the best, median and worst overall see Figure 4 below. As you can see, the four runs have produced very similar average precision figures across all the queries. This we feel is as a result of the sparsity of connectivity data available and our best-parameter constants that limited the number of documents reranked by the linkage algorithms.

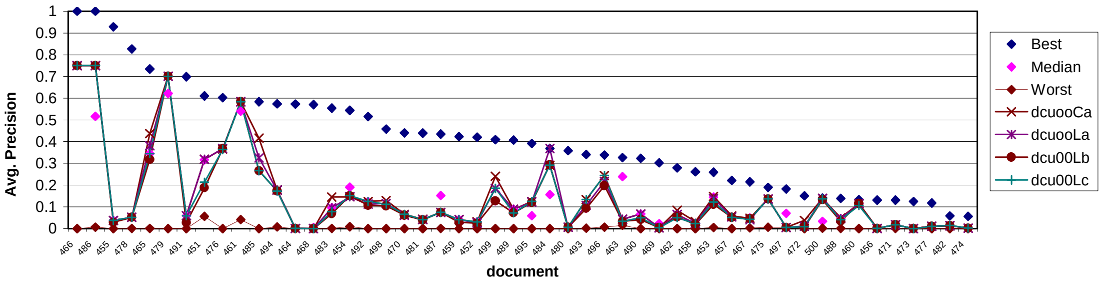

Figure 4: Average precision per topic (ordered by Best)

Our recall figures were dependent on the quality of the relevant-set generated in the content-only phase. Figure 5 shows recall at 1,000 documents for *dcu00ca* and the best, median and worst results. Recall that we never reranked even the top 1,000 documents from the relevant-set, so this recall at 1,000 figure never changed from *dcu00ca* for any of the connectivity-based runs. Hence *dcu00ca* is considered representative and is the only run plotted on the graph.

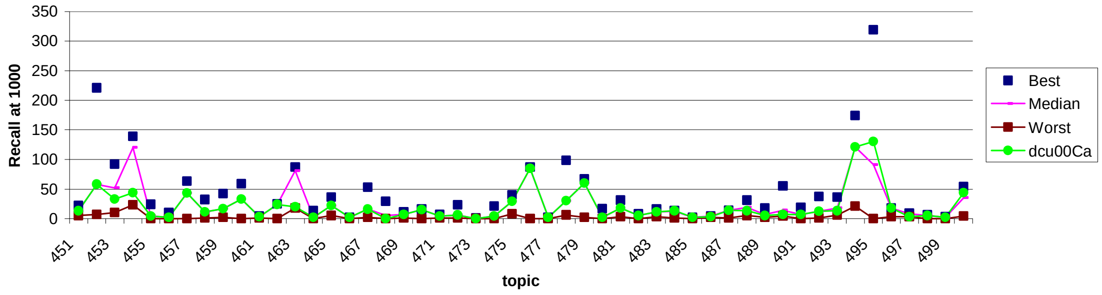

Figure 5: Content-only (*dcu00ca*) recall at 1,000 documents, including Best, Median and Worst

5. Conclusions & Future Work

It is very difficult to draw any concrete conclusions from our experiments. We feel that this is due to the fact that the WT10g dataset does not contain the density of inter-domain links as would be needed to draw these conclusions. We ran some simple experiments on the connectivity data to judge the sparseness, or otherwise, of the links within the connectivity data. We found that, in all approximately, 2% of the links were functional links while a large 98% were structural links. Recall that we were only working with functional links in our experiments. This lack of functional links seriously hampers our experiments, so much so that we decided not to implement HITS or PageRank on the dataset as additional runs. We still expect *dcu00la* would be the best reranking approach, unfortunately this is no improvement over *dcu00ca*. This does leave questions as to whether this would be best because it has the least effect on the content-result, or because the algorithm may be a more effective algorithm.

Perhaps HITS style expansion would be a better alternative than our approach to 'relevant-set' generation. The HITS approach would have the benefit of generating a set of documents that should have a higher density of functional links linking them together. Maybe a weighted HITS approach to generating a 'relevant-set' would be best. In [7] Henzinger & Bharat generate their expanded-set in the normal HITS way and then implemented a weighted algorithm over this set. This is open to additional experimentation. If we could generate a more focused 'relevant-set' of documents then we could perhaps defeat the problem of topic-drift.

In order to provide a framework within which we will be able to test out these concepts we have begun the development of our own crawler (and VSM-based IR system) to gather a dataset of language dependent documents for use in our future connectivity analysis experiments. We feel that a small minority language on the web may have an interesting link structure, in that we expect a large degree of connectivity between documents in the minority language. In developing the queuing function for selecting the candidate documents for crawling, we regard the following points as being of utmost importance:

- maintaining a weighted queue of known URLs to be indexed, favouring inter-domain links
- utilising a source of starter URLs from a source of functional outLinks (e.g. Yahoo! [12])
- documents from domains that have not yet been indexed and in the URL queue must be weighted with a highest possible weighting to increase the overall number of both functional links and domains indexed.

This would allow us to utilise a different dataset, which we hope will more accurately reflect the structure of the WWW, which is essential for us to be able to draw any concrete conclusions from our experimentation. We hope to present some of our work at TREC-2001.

6. Appendix

Due to the fact that our queries were manually generated we have made our queries available for downloading by interested parties. For the most part, the queries are unmodified from the actual topic titles. They can be found at the following URL:

http://www.compapp.dcu.ie/~cgurrin/trec9/queries.html

We are also making the results of our title-only unofficial run available for downloading. If you are interested in getting our results they can be found off the following URL:

http://www.compapp.dcu.ie/~cgurrin/trec9/titlerun.html

This page also contains links to the homepage of any third party software that we used during our experiments.

7. References

[1] C. Gurrin, A.F. Smeaton. - “A Connectivity Analysis Approach to Increasing Precision in Retrieval From Hyperlinked Documents”
Proc.8th Text Retrieval Conference (TREC-8), 1999

[2] Y. Li “Toward a more Qualitative Search Engine”
IEEE Internet Computing, Vol 2 No. 4, Jul-Aug 1998

[3] Microsoft Microsoft Index Server
http://www.microsoft.com/NTServer/web/exec/feature/ … IndexServerSummary.asp

[4] Microsoft Microsoft SQL Server 7
http://www.microsoft.com/sql/default.htm

[5] K. Bharat, A. Broder, M. Henzinger, P.Kumar, S. Venkatasubramanian
“The Connectivity Server: fast access to linkage information on the web”
Proc. 7th International WWW Conference, 1998

[6] J. Kleinberg “Authorative Sources in a Hyperlinked Environment”
Proc. 9th ACM-SIAM Symposium on Discrete Algorithms, 1998

[7] K. Bharat & M. Henzinger - "Improved Algorithms for Topic Distillation in a Hyperlinked Environment"
Proc. 23rd Annual International ACM SIGIR Conference on Research and Development in IR. 2000

[8] Google Google Search Engine
http://www.google.com

[9] L. Page “The PageRank Citation Ranking: Bringing Order to the Web”
Stanford Digital Libraries working paper, 1997-0072

[10] E. Spertus "Parasite: Mining Structural Information on the Web"
Proc. 6th International WWW Conference, 1997

[11] B. Amento, L. Terveen, W. Hill - "Does ‘Authority’ Mean Quality? Predicting Expert Quality Ratings of Web Documents"
Proc. 23rd Annual International ACM SIGIR Conference on Research and Development in IR. 2000

[12] Yahoo! Yahoo! Search Engine
http://www.yahoo.com