Improved Algorithms for Topic Distillation in a Hyperlinked Environment

To appear at the 21st ACM SIGIR Conference on Research and Development in Information Retrieval 1998

Krishna Bharat
Digital Equipment Corporation
Systems Research Center
130 Lytton Avenue
Palo Alto, CA 94301
bharat@pa.dec.com

Monika R. Henzinger
Digital Equipment Corporation
Systems Research Center
130 Lytton Avenue
Palo Alto, CA 94301
monika@pa.dec.com

Abstract This paper addresses the problem of *topic distillation* on the World Wide Web, namely, given a typical user query to find quality documents related to the query topic. Connectivity analysis has been shown to be useful in identifying high quality pages within a topic specific graph of hyperlinked documents. The essence of our approach is to augment a previous connectivity analysis based algorithm with content analysis. We identify three problems with the existing approach and devise algorithms to tackle them. The results of a user evaluation are reported that show an improvement of precision at 10 documents by at least 45% over pure connectivity analysis.

1 Introduction

Search services on the World Wide Web are the information retrieval systems that most people are familiar with. As argued by Marchionini [23] "end users want to achieve their goals with a minimum of cognitive load and a maximum of enjoyment." Correspondingly, in the context of Web searches we observe that users tend to type short queries (one to three words) [2, 9], without giving much thought to query formulation. Additionally, it is often the case that users themselves are unclear about their information need [12] when framing the query. Since determining relevance accurately under these circumstances is hard, most search services are content to return exact query matches – which may or may not satisfy the user's actual information need.

In this paper we describe a system that takes a somewhat different approach in the same context. Given typical user queries on the World Wide Web (i.e., short queries), our system attempts to find quality documents related to the *topic* of the query. Note that this is more general than finding a precise query match and not as ambitious as trying to exactly satisfy the user's information need. The latter is often hard to do since most short queries do not express the need unambiguously. In cases where the query is ambiguous, i.e. there is more than one possible query topic, our goal is to return relevant documents for (some of) the main query topics. This excludes minor interpretations of the query and encourages users to type in queries that are representative of the topic they seek to explore. We call the process of finding quality documents on a query topic, *topic distillation*.

The situation on the World Wide Web is different from the setting of conventional information retrieval systems for several reasons. The main reasons are:

- Users tend to use very short queries (1 to 3 words per query [2, 9]) and are very reluctant to give feedback.
- The collection changes continuously.
- The quality and usefulness of documents varies widely. Some documents are very focused; others involve a patchwork of subjects. Many are not intended to be sources of information.
- Preprocessing all the documents in the corpus requires a massive effort and is usually not feasible.

However, there is an additional source of information that an information retrieval system on the World Wide Web can harness: namely, the opinions of people who create hyperlinks. A simple approach to finding quality documents is to assume that if document A has a hyperlink to document B, then the author of document A thinks that document B contains valuable information. Thus, using the in-degree of a document as a measure of its quality is a first heuristic. However, transitivity is worth exploiting as well. If A is seen to point to a lot of good documents, then A's opinion becomes more valuable, and the fact that A points to B would suggest that B is a good document as well.

Using this basic idea, Kleinberg [21] developed a *connectivity analysis algorithm* for hyperlinked environments. Given an initial set of results from a search service, the algorithm extracts a subgraph from the Web containing the result set and its neighboring documents. This is used as a basis for an iterative computation that estimates the value of each document as a source of relevant links and as a source of useful content.

While this algorithm works well for some queries, it performed poorly in several of our test cases. To better understand its behavior we built a visualization tool. This enabled us to discover three problems with connectivity analysis as suggested by Kleinberg, i.e. a "links-only" approach: *Mutually Reinforcing Relationships Between Hosts* (where certain arrangements of documents "conspire" to dominate the computation), *Automatically Generated Links* (where no human's opinion is expressed by the link), and *Non-relevant Documents* (where the graph contains documents not relevant to the query topic). In this paper we present several techniques for tackling these three scenarios. The last problem is by far the most common, and our general solution is to use content analysis to help keep the connectivity-based computation "on the topic."

We compare the performance of 10 algorithms with the basic Kleinberg algorithm on 28 topics that were used previously in [6]. The best approach increases the precision over basic Kleinberg by at least 45% and takes less than 3 minutes. This running time is dominated by the time to fetch 130 documents from the World Wide Web and can be reduced considerably when term vectors for the documents are available.

The paper is structured as follows. Section 2 describes the connectivity analysis algorithm, its implementation, and the problems we encountered. Section 3 shows how be address the first problem, Section 4 gives algorithms addressing the other two problems. In Section 5 we evaluate the different algorithms. Section 6 presents considerably faster algorithms that additionally improve precision. In Section 7 we discuss related work.

2 Connectivity Analysis

The goal of connectivity analysis is to exploit linkage information between documents, based on the assumption that a link between two documents implies that the documents contain related content (*Assumption i*), and that if the documents were authored by different people then the first author found the second document valuable (*Assumption ii*). In 1997 Kleinberg [21] published an algorithm for connectivity analysis on the World Wide Web which we describe next.

2.1 Kleinberg's Algorithm

The algorithm computes two scores for each document: a *hub* score and an *authority* score. Documents that have high authority scores are expected to have relevant content, whereas documents with high hub scores are expected to contain *links* to relevant content. The intuition is as follows. A document which points to many others is a good hub, and a document that many documents point to is a good authority. Transitively, a document that points to many good authorities is an even better hub, and similarly a document pointed to by many good hubs is an even better authority.

In the context of a user query the algorithm first constructs a query specific graph whose nodes are documents. Then it iteratively computes the hub and authority scores for the nodes. The graph is constructed as follows. A *start set* of documents matching the query is fetched from a search engine (say the top 200 matches). This set is augmented by its *neighborhood*, which is the set of documents that either point to or are pointed to by documents in the start set. In practice, since the indegree of nodes can be very large, Kleinberg recommends considering at most 50 predecessors of a document. The documents in the start set and its neighborhood together form the nodes of the *neighborhood graph*. Hyperlinks between documents *not on the same host* form the directed edges. Links within the same host[1] are assumed to be by the same author and hence are not indicators of value. The computation of hub and authority scores is done as follows.

(1) Let N be the set of nodes in the neighborhood graph.
(2) For every node n in N, let $H[n]$ be its hub score and $A[n]$ its authority score.
(3) Initialize $H[n]$ and $A[n]$ to 1 for all n in N.
(4) While the vectors H and A have not converged:
(5) For all n in N, $A[n] := \sum_{(n',n)\in N} H[n']$
(6) For all n in N, $H[n] := \sum_{(n,n')\in N} A[n']$
(7) Normalize the H and A vectors.

Kleinberg [21] proved that the H and A vectors will eventually converge, i.e., that termination is guaranteed. In practice we found the vectors to converge in about 10 iterations. The documents are then ranked by hub and authority scores respectively.

Note that the algorithm does not claim to find *all* relevant pages, since there may be some that have good content but have not been linked to by many authors. In our evaluation of different algorithms we use Kleinberg's algorithm [21] as our baseline, which we call *base*.

2.2 Implementation

To determine the neighborhood of the start set the algorithm needs to follow links that point in and out of these documents. Outlinks are easily obtained by fetching the document. One way of obtaining inlinks is to use AltaVista queries of the form $link : u$, which returns a list of documents that point to the URL u. This was the implementation used by [21].

In our queries, the neighborhood graph contained on the order of 2000 nodes. The running time is completely dominated by the time it takes to fetch the documents. With a download rate of 1 document per second queries takes about 30 minutes.

To get fast access to linkage information within the World Wide Web, we built a *Connectivity Server* [4] that provides linkage information for all pages indexed by the AltaVista search engine. The server provides a specialized interface to compute the neighborhood graph for a set of URLs. This speeds up the graph construction to under half a minute and enables us to handle queries in almost real time.

We ran the computation for 150 iterations in each case, although the system seemed to converge after 10.

2.3 Problems Encountered

We found that the algorithm as described above did not work well in all cases. Obviously, if there are very few edges in the neighborhood graph not much can be inferred from the connectivity. We built a neighborhood visualization tool which allowed us to trace the computation and discover three other reasons why the algorithm tends to fail:

1. *Mutually Reinforcing Relationships Between Hosts*: Sometimes a set of documents on one host point to a single document on a second host. This drives up the hub scores of the documents on the first host

[1] We assume throughout the paper that the *host* can be determined from the URL-string.

and the authority score of the document on the second host. The reverse case, where there is one document on a first host pointing to multiple documents on a second host, creates the same problem. Since we make the (simplifying) assumption that the set of documents on each host was authored by a single author or organization, these situations give undue weight to the opinion of one "person."

2. *Automatically Generated Links*: Web documents generated by tools (e.g., Web authoring tools, database conversion tools) often have links that were inserted by the tool. For example, the Hypernews system which turns USENET News articles into Web pages, automatically inserts a link to the Hypernews Web site. In such cases Assumption *ii*, namely that a human's opinion is represented by the link, does not apply.

3. *Non-relevant Nodes*: We often find that the neighborhood graph contains documents not relevant to the query topic. If these nodes are well connected, the *topic drift* problem arises: the most-highly ranked authorities and hubs tend not to be about the original topic. For example, when running the algorithm on the query "jaguar *and* car" the computation drifted to the general topic "car" and returned the home pages of different car manufacturers as top authorities, and lists of car manufacturers as the best hubs.

3 Improved Connectivity Analysis

As discussed in the previous section we identified three problems with Kleinberg's algorithm. In this section we address problem 1, mutually reinforcing relationships between hosts. The next section addresses problems 2 and 3.

Mutually reinforcing relationships between hosts give undue weight to the opinion of a single person. Ideally we would like all the documents on a single host to have the same influence on the document they are connected to as a single document would. To achieve this we give fractional weights to edges in such cases:

If there are k edges from documents on a first host to a single document on a second host we give each edge an *authority weight* of $1/k$. This weight is used when computing the authority score of the document on the second host. If there are l edges from a single document on a first host to a set of documents on a second host, we give each edge a *hub weight* of $1/l$. Additionally, we discard isolated nodes from the graph. This leads to the following modified algorithm:

(4) While the vectors H and A have not converged:
(5) For all n in N,
$A[n] := \sum_{(n',n)\in N} H[n'] \times auth\_wt(n',n)$
(6) For all n in N,
$H[n] := \sum_{(n,n')\in N} A[n'] \times hub\_wt(n,n')$
(7) Normalize the H and A vectors.

In the appendix we prove that the H and A vectors converge, i.e., that the algorithm terminates.

This modified algorithm was effective in eliminating the mutually reinforcing relationship problem in all the cases where we had encountered it. In our evaluation we call this improved algorithm, *imp*.

4 Combining Connectivity and Content Analysis

In this section we combine content analysis using traditional Information Retrieval techniques with improved connectivity analysis to tackle topic drift. There are two basic approaches both assuming we can determine the relevance of a node to the query topic: (i) eliminating non-relevant nodes from the graph, and (ii) regulating the influence of a node based on its relevance. We have also experimented with combinations of these techniques. These mostly address problem 3 since they discard or penalize nodes that do not belong to the topic. However, in practice they also seem to solve problem 2, since automatically generated links often point to pages outside the topic.

4.1 Computing Relevance Weights for Nodes

The relevance weight of a node equals the similarity of its document to the query topic. We describe next how to compute the similarity score of a document D. As mentioned before, the query topic is broader than the query itself. Thus matching the query against the document is usually not sufficient. Instead we use the documents in the start set to define a broader query and match every document in the graph against this query. Specifically, we consider the concatenation of the first 1000 words from each document to be the query, Q and compute $similarity(Q, D)$.

In our implementation, since queries are long and the document vocabulary tends to be varied we use term frequency weighting. We use cosine normalization in weighting both the query and the documents since the deviation in term vector lengths is large. See Salton and Buckley [28] for a discussion of weighting options. Specifically,

$$similarity(Q, D_j) = \frac{\sum_{i=1}^{t}(w_{iq} \times w_{ij})}{\sqrt{\sum_{i=1}^{t}(w_{iq})^2 \times \sum_{i=1}^{t}(w_{ij})^2}}$$

where
$w_{iq} = freq_{iq} \times IDF_i$,
$w_{ij} = freq_{ij} \times IDF_i$,
$freq_{iq}$ = the frequency of the term i in query Q,
$freq_{ij}$ = the frequency of the term i in document D_j,
IDF_i = an estimate of the inverse document frequency of term i on the World Wide Web.

4.2 Pruning Nodes from the Neighborhood Graph

There are many approaches one can take to use the relevance weight of a node to decide if it should be eliminated from the graph. We investigated approaches based on thresholding the relevance weight. All nodes whose weights are below a threshold are pruned. Thresholds are picked in one of three ways:

1. *Median Weight*: The threshold is the median of all the relevance weights.

2. *Start Set Median Weight*: The threshold is the median of the relevance weights of the nodes in the start set.

3. *Fraction of Maximum Weight*: The threshold is a fixed fraction of the maximum weight. We used $max/10$ in our experiments.

On the pruned graph we run the *imp* algorithm. We call the corresponding algorithms: *med*, *startmed*, and *maxby*10.

4.3 Regulating the Influence of a Node

This approach seeks to modulate how much a node influences its neighbors based on its relevance weight. If $W[n]$ is the relevance weight of a node n and $A[n]$ the authority score of the node we use $W[n] \times A[n]$ instead of $A[n]$ in computing the hub scores of nodes that point to it. Similarly, if $H[n]$ is its hub score we use $W[n] \times H[n]$ instead of $H[n]$ in computing the authority score of nodes it points to. This reduces the influence of less relevant nodes on the scores of their neighbors.

Combining the previous four approaches with the above strategy gives us four more algorithms, which we call: *impr*, *medr*, *startmedr*, and *maxby*10*r*.

4.4 Implementation

Unlike the previous implementation where it sufficed to get the graph from the Connectivity Server, in this case we need to fetch all the documents to do content analysis. To build term vectors we eliminate stop words and use Porter stemming [27]. For IDF weights, since we know of no source of IDF weights for the Web and of no official representative collection, we had to build our own collection. Hence we used term frequencies measured in a crawl of 400,000 Yahoo! [30] documents in January 1997.

5 Evaluation

Traditionally, ranking schemes are evaluated by computing precision and recall on a pre-labeled corpus, such as the TREC [17] collection. We compare our algorithms based on precision and relative recall at 5 and 10 documents. We used relative recall instead of recall since we do not know the number of relevant documents for a topic on the Web, or even in the Neighborhood Graph. We used a set of 28 queries previously used by [6] in comparing the rankings from their version of Kleinberg's algorithm with category listings on the Web. Table 1 gives a listing of the queries ordered by the number of results returned by AltaVista in December 1997 for each query, which can be taken as a measure of the topic's popularity on the Web.

We ran our 8 algorithms and *base* on each of the queries and considered documents with the top 14 hub and authority scores. The set of top authority documents from all the algorithms were pooled together randomly and independently rated for relevance by 3 volunteers. The ratings were then combined and the final relevance rating for each document was decided by majority vote. A similar rating was done for the top hub documents. In each case the subjects were instructed to determine whether the document was not relevant to the topic (case i), relevant to the the topic (case ii), or both relevant to the topic and a good example of a hub or an authority as the case may be (case iii). Only documents classified under case iii by a majority of reviewers (i.e., 2 out of 3) were considered relevant for the purposes of computing precision and relative recall.

The subjects were encouraged to follow links and browse the document's neighborhood before deciding on a rating. Specifically, the subjects were told:

Abb.	*Query*	*AV ct.*	t_a	t_h
VC	"vintage car"	2159	13	15
RE	+recycling +cans	2811	13	10
ZB	"Zen Buddhism"	4617	21	22
TH	+Thailand +tourism	4642	20	18
PA	"parallel architecture"	4709	4	11
SC	"stamp collecting"	5581	20	21
TE	telecommuting	7436	20	22
SU	sushi	8082	8	8
AL	alcoholism	9596	8	17
CG	"classical guitar"	11404	19	31
LD	"lyme disease"	12123	16	12
BI	bicycling	16956	26	24
FH	"field hockey"	20410	33	22
AP	"amusement park"	25202	19	19
TT	"table tennis"	27409	12	20
RC	"rock climbing"	31286	27	30
CV	"computer vision"	35762	26	23
SH	shakespeare	41885	13	15
CR	cruises	46820	17	24
GW	"Gulf war"	49055	16	21
GA	gardening	65009	20	20
CH	cheese	65782	9	10
HI	HIV	82218	20	28
AA	"affirmative action"	104280	10	9
MF	"mutual funds"	110064	22	25
BL	blues	126971	23	16
GD	"graphic design"	158847	11	9
AR	architecture	306720	12	19

Table 1: Queries used in sorted order of AltaVista result set size in December 1997. The table also lists for each topic the total number of relevant documents that appeared in the top-10 ranking of at least one algorithm. For authority rankings this is listed as t_a and for hub rankings as t_h.

"You have some latitude in deciding what constitutes a good hub or authority. A good hub generally has useful links. A good authority is generally a document with useful content. If a document with little content has links to relevant content-rich documents on the same site (e.g., if it is a 'Table of Contents' page), it may still count as an authoritative page. You might instead choose to regard all good hubs as good authorities. Whatever policy you adopt please be consistent."

Two issues came up: (i) Sometimes queries had more than one interpretation. For instance, some reviewers restricted *architecture* to building related topics, whereas others included computer architecture as well. (ii) There was disagreement among the reviewers on whether to include pages on the topic containing very localized information, e.g., pages on bicycling trails in New Jersey for the query "bicycling."

No rating was given in cases where documents were not accessible or were in a language that the subjects did not understand. To compensate for this we obtained ratings for the top 14 documents in each ranking, and omitted the unrated documents. This gave us a list of at least 10 documents for each algorithm–topic pair with 3 ratings for each. We computed precision and relative recall for this list using the combined relevance measure described previously (relevant if placed in class iii by a majority of the reviewers). We computed precision at 5 and at 10 documents for each algorithm–topic pair, as well as average precision for specific sets of documents and all the documents combined. To compute relative recall in the context of a topic, we first determined t, the total number of relevant documents for the topic occurring in the top-10 ranking of at least one of the algorithms. Table 1 lists values of t for the various topics (t_a for authorities and t_h for hubs). For each algorithm, relative recall at 5 (similarly 10) documents was computed as the number of relevant documents in the top 5 (similarly 10) ranked documents expressed as a fraction of t.

Table 2 shows the precision after the top 5 and 10 ranked authority documents. We classified the five queries with the smallest AltaVista result set size as *rare*, and the five the with largest result set size as *popular*. We also give precision values for the sets of rare and popular queries. Similarly, Table 3 gives precision values for hub documents.

First, we discuss the performance in the context of authority ranking. We observe that in all cases *imp*, which eliminates mutually reinforcing relationships between hosts, provides an appreciable improvement over *base*, the algorithm described by Kleinberg. Adding content analysis either by pruning nodes or regulating the influence of nodes improves on *imp*, especially in the case of rare topics. *Med*, *startmed*, and *maxby*10 all perform roughly the same and improve precision by about 10% over *imp*. Regulation helps *imp* in all cases, about as much as pruning. For the algorithms that use pruning, adding regulation does not seem to affect precision.

On both popular and rare topics the algorithms performed, in general, worse than on all topics. Precision for rare topics is in general lower than for popular topics. We conjecture that rare topics do not have enough connectivity for the algorithms to exploit, while for popular topics that threshold based pruning is too simplistic. In the next section we present algorithms that prune more selectively. One of them performs significantly better on popular topics.

To summarize authority rankings, *imp* improves precision by at least 26% over *base*; regulation and pruning each improve precision further by about 10%, but combining them does not seem to give any additional improvement.

Considering precision in the ranking for hubs we find as before that *imp* improves on *base* (by 23% or more), and *med* improves on *imp* by a further 10%. Regulation slightly improves *imp* and *maxby*10 but not the others.

Overall hub precisions are better than authority precisions, even for *base*, but *medr* still improves precision by 45% over *base*. In general at 10 precision averaged over all topics is higher than on *rare* and *popular* topics.

Due to the distribution of the t_a and t_h (see Table 1) no algorithm can have a better relative recall at 10 than 0.65 for authorities and 0.6 for hubs. *Base* achieved a relative recall at 10 of 0.27 for authorities and 0.29 for hubs. Our best algorithm for authorities gave a relative recall of 0.41; similarly for hubs it was 0.46 (see Table 4), i.e., we achieved roughly half the potential improvement by this measure.

6 Partial Content Analysis

Although the content analysis based algorithms described in the previous section improve precision – they do so at the expense of response time. Query response times with *imp* are about half a minute, whereas content analysis of all nodes in the graph requires downloading roughly 2000 documents from the Web which can take about 30 minutes. Ideally, we would like to use the advantage that content analysis provides – i.e., reduction of the effect of non-relevant nodes, without paying the high cost of a full graph download. In this section we describe two algorithms that involve content pruning but only analyze a part of the graph (less than 10% of the nodes). This makes them a factor of 10 faster than previous content analysis based algorithms, supporting query response times of around 3 minutes, which are more tolerable.

Our two algorithms are motivated by the observation that not all nodes are equally influential in deciding the outcome of the improved connectivity analysis. Some are better connected than others and hence likely to dominate the computation. The new algorithms attempt to selectively analyze and prune if needed, the nodes that are most influential in the outcome. Since the act of pruning itself alters the course of the computation selecting the best candidates for pruning is problematic. We use two heuristics, *degree based pruning* and *iterative pruning*, to select the nodes to be analyzed. These are described in the subsections below.

In both cases, as before, an expanded query, Q, is needed to compute the relevance weights of nodes. Previously the entire start set was used to compute Q. With partial content analysis only a subset of the start set (30 documents in our implementation) is used for this purpose. These are selected by another heuristic, based solely on the information the Connectivity Server can provide – namely the URL and connectivity of each document. With some experimentation we arrived at a heuristic that selects nodes based on in-degree, out-degree, and match of the URL string with the original query. Specifically, we select the 30 start set documents that maximize the value of $in\_degree + 2 \times num\_query\_matches + has\_out\_links$, where $num\_query\_matches$ is the number of unique substrings of the URL that exactly match a term in the user's query, and has_out_links is 1 if the node has at least one out-edge and otherwise 0.

The documents selected from the start set are fetched and their initial 1000 words are concatenated to give Q. Each of them is then scored against Q and the 25th percentile relevance weight is selected as the *pruning threshold*. The pruning threshold is used in the next phase (the pruning phase) to eliminate some of the influential but non-relevant nodes in the graph. In computing similarity between the query, Q, and a document, D, a slightly modified formula is used from before. The weight of terms in the original query is boosted by a factor of three. Specifically, w_{iq} is computed as $freq_{iq} \times IDF_i \times 3$, whenever term i is a (stemmed form) of a term in the user's query. This is done in the pruning phase as well.

In the pruning phase a hundred nodes are selected from the graph by one of two heuristics, which we describe next. They are matched with Q, and pruned if their relevance weight is below the pruning threshold. In all at most 130 documents are fetched and analyzed.

We experimented with two partial pruning approaches: (i) *Degree Based Pruing* and (ii) *Iterative Pruning.*

6.1 Degree Based Pruning

In degree based pruning, the in and out degrees of the nodes are used to select nodes that might be influential. Specifically, we use $4 \times in\_degree + out\_degree$ as a measure of influence. The top 100 nodes by this measure are fetched, scored against Q and pruned if their

		Without Regulation					*With Regulation*				*Partial*	
		base	*imp*	*med*	*startmed*	*maxby10*	*impr*	*medr*	*startmedr*	*maxby10r*	*pca0*	*pca1*
All	*At 5*	0.52	0.66	0.73	0.65	0.69	0.67	0.72	0.65	0.7	0.72	0.75
	At 10	0.46	0.58	0.65	0.66	0.62	0.62	0.65	0.65	0.64	0.64	0.67
Rare	*At 5*	0.24	0.36	0.64	0.48	0.55	0.6	0.6	0.48	0.6	0.48	0.6
	At 10	0.18	0.24	0.5	0.5	0.43	0.44	0.48	0.54	0.48	0.44	0.64
Popular	*At 5*	0.36	0.55	0.6	0.68	0.64	0.55	0.6	0.6	0.6	0.68	0.88
	At 10	0.4	0.54	0.57	0.7	0.6	0.58	0.6	0.62	0.64	0.68	0.8

Table 2: Average Precision at Top 5 and 10 ranked authority documents

		Without Regulation					*With Regulation*				*Partial*	
		base	*imp*	*med*	*startmed*	*maxby10*	*impr*	*medr*	*startmedr*	*maxby10r*	*pca0*	*pca1*
All	*At 5*	0.6	0.74	0.87	0.78	0.75	0.8	0.87	0.77	0.81	0.8	0.8
	At 10	0.56	0.73	0.79	0.7	0.73	0.76	0.81	0.69	0.76	0.74	0.71
Rare	*At 5*	0.44	0.64	0.88	0.72	0.6	0.8	0.88	0.8	0.8	0.56	0.72
	At 10	0.46	0.6	0.76	0.6	0.64	0.76	0.8	0.66	0.76	0.53	0.63
Popular	*At 5*	0.48	0.8	0.8	0.88	0.8	0.8	0.8	0.72	0.8	1.0	0.68
	At 10	0.42	0.68	0.74	0.72	0.68	0.7	0.74	0.6	0.7	0.76	0.54

Table 3: Average Precision at Top 5 and 10 ranked hub documents

score falls below the pruning threshold. After this, connectivity analysis as in *imp* is run for 10 iterations on the pruned graph. The ranking for hubs and authorities computed by *imp* is returned as the final ranking. This algorithm is called *pca*0.

6.2 Iterative Pruning

For iterative pruning we use connectivity analysis itself (specifically the *imp* algorithm) to select nodes to prune.

Pruning happens over a sequence of rounds. In each round *imp* is run for 10 iterations to get a listing of the (currently) best hubs and authorities. The top documents by these rankings are examined in decreasing order of rank, alternating between the hub and the authority ranking. When examining a document, we fetch it and compute its relevance (if it is not already fetched) until either 5 documents have been fetched in the round or enough top ranked documents have been found to be relevant (15 in our experiments). In the latter case the algorithm terminates. In the former case the algorithm terminates the round and starts a new round on the pruned graph, until an allotted quota of documents has been fetched (100 in our implementation). The rankings computed in the last round are returned as the best hubs and authorities overall.

The motivation for stopping each round when 5 documents have been fetched is that when combating topic drift by pruning, it is usually sufficient if the top ranked documents are pruned, since they tend to be high degree nodes that support others in the ranking. After this point we think it is more profitable to execute another round than to continue with the pruning.

This algorithm is called *pca*1.

6.3 Comparison with Previous Techniques

In Table 2 we show precision for authority ranking by the new algorithms (*pca*0 and *pca*1) as well. Even though our main goal was to speed up the computation, *pca*0 performs comparably with the best previous algorithm and *pca*1 improves precision. We believe that the *pca*1 improvement comes from the fact that partial content analysis avoids pruning non-influential documents that are below the threshold in terms of relevance but are connected to and support good hubs and authorities on the topic.

Table 3 show precisions for hub ranked documents. For all topics, *pca*0 and *pca*1 perform 10% worse than *medr*, the best of the previous algorithms. For the topics where *pca*1 performs poorly we found that it uses up its whole quota of 100 documents, suggesting that a larger quota allowing for more pruning would be more successful. For example, in the case of "graphic design" *pca*1 used up its quota before it could eliminate a set of irrelevant documents containing automatically generated links. These links pointed to a very good authority which placed the irrelevant documents at the top of the hub list.

In terms of relative recall, compared with the best previous algorithm, selective pruning performed comparably for authority documents, and about 10% worse for hub documents.

7 Related Work

The *ARC* algorithm of Chakrabarti et al [6] also extends Kleinberg's algorithm with textual analysis. ARC computes a distance-2 neighborhood graph and weights edges. The weight of each edge is based on the match between the query terms and the text surrounding the hyperlink in the source document. Regulation is similar to their approach but there are three differences: (i) We use an expanded query instead of the original query. (ii) The relevance is computed using the whole document, not just a window surrounding the hyperlink. (iii) The weight of an edge is either the relevance of the source document or the target document depending on whether authority or hub scores are being computed.

Connectivity analysis of Web hyperlinks resembles the work on citation and co-citation analysis in the area of bibliometrics. This is used to discover influential publications and authors with similar interests within the articles of a certain field of study. See [22] for a discussion on applying bibliometrics to the World Wide Web. Citation analysis has been criticized (see [8]) as a source of systematic bias, since members of cliquish communities tend to

		Without Regulation					With Regulation				Partial	
		base	*imp*	*med*	*startmed*	*maxby10*	*impr*	*medr*	*startmedr*	*maxby10r*	*pca0*	*pca1*
Authorities	*At 5*	0.15	0.19	0.22	0.2	0.21	0.2	0.22	0.19	0.21	0.22	0.23
	At 10	0.27	0.35	0.39	0.41	0.37	0.38	0.39	0.4	0.38	0.38	0.41
Hubs	*At 5*	0.16	0.21	0.26	0.22	0.21	0.24	0.26	0.22	0.24	0.24	0.23
	At 10	0.29	0.41	0.46	0.38	0.4	0.43	0.46	0.38	0.43	0.41	0.4

Table 4: Relative Recall

cite each other preferentially, and some authors are cited out of deference rather than relevance. On the Web this is less of a problem since the community is diverse and distributed, and the right to publish cannot be restricted by cliques. Indeed, the importance of considering referential statistics in document selection is increased since there is no quality control on the Web.

Others have used inter-document linkage to compute useful data on the Web as well. Pirolli et al [26] run a computation on a inter-document matrix, with weights derived from linkage, content similarity and usage data, to identify usable structures. *PageRank* [25] is a ranking algorithm for Web documents that uses connectivity to compute a topic-independent score for each document.

There has been much work in IR on supporting topic exploration. This is typically done by letting users browse topic hierarchies that are either predetermined (e.g., *Cat-a-Cone* [19]), or dynamically constructed by clustering based on user selection (e.g., *Scatter/Gather* [10], *Paraphrase* [3]). Another approach to topic exploration is interactive query expansion where new terms are suggested to help focus the query (e.g., [24, 15]). On the Web there are examples of topic hierarchies (e.g., Yahoo! [30, 16]), dynamic clustering (AltaVista's *LiveTopics* [5]) and query expansion (as in Excite [13]). The goal of topic exploration is to locate a set of documents dealing with the user's topic of interest, whereas topic distillation assumes such a set and finds quality documents within it. Hence, topic exploration may be viewed as a powerful preliminary step to topic distillation. This was suggested by Hearst in [18], who observed that Kleinberg's algorithm does not bring forth documents that deal with less popular interpretations of the query. She suggests first clustering the documents to separate out the subtopics and then analyzing the induced subgraphs individually. Another option would be to modify the algorithm so that within-cluster edges have a higher weight than cross-cluster edges. This would allow nodes belonging to smaller, less developed topics to be supported by nodes belonging to other related topics.

Finally, our approach to evaluating precision at a fixed number of result documents based on user relevance ratings seems typical of ranking evaluations done on the Web (e.g., search service comparisons [7, 11]).

8 Conclusions

In this paper we showed that Kleinberg's connectivity analysis has three problems. We presented various algorithms to address them. The simple modification suggested in algorithm *imp* achieved a considerable improvement in precision. Precision was further improved by adding content analysis, with algorithms *medr*, *pca*0 and *pca*1 being the most promising. In our current implementation *pca*0 and *pca*1 compute ranking with a relatively fast turnaround (about 3 minutes) when using the Connectivity Server to compute the graph.

For authorities, *pca*1 seems to be the best algorithm overall. It provides enough of an improvement over *imp* to justify the overhead of analyzing a small set of documents. For hubs, *medr* is the best general-purpose algorithm, but if term vectors are not available for the documents in the collection, we suggest using *imp*. In each case the best algorithm improves precision over baseline Kleinberg by at least 45%.

This approach is limited to topics that are well represented and well connected on the Web. Additionally, this work assumes that the results of a search service query defines a good start set, which is debatable. It would be interesting to apply query expansion and clustering to produce a better start set.

Hypertext encourages documents to be split up into pieces. One could argue that what users are looking for on the Web are good sites, containing a set of connected documents on the topic, rather than individual documents. Connectivity based ranking schemes might serve this purpose well since they have a tendency to return the root document within a site, which is a good starting point for exploration. This happens because external hyperlinks most often link to the root document, even if it does not have much content.

References

[1] AltaVista, `www.altavista.digital.com/`

[2] Anick, P.G. 1994 "Adapting a Full-text Information Retrieval System to Computer the Troubleshooting Domain." *Proc. of ACM SIGIR '94* pp. 349–358.

[3] Anick, P.G. and Vaithyanathan, S. 1997. "Exploiting Clustering and Phrases for Context-Based Information Retrieval." *Proc. of ACM SIGIR '97* pp. 314–323.

[4] Bharat, K., Broder, A., Henzinger, M., Kumar, P., and Venkatasubramian, S. 1998. "The Connectivity Server: Fast Access to Linkage Information on the Web.", *Proc. of 7th World Wide Web Conference*, pp. 469–477, available as `www7.conf.au/programme/fullpapers/1938/com1938.htm`

[5] Bourdoncle, F. 1997 "LiveTopics: Recherche Visuelle d'Information sur l'Internet." *Dossiers de l'Audiovisuel, La Documentation Francaise* No. 74 (July-Aug 1997), pp. 36–38.

[6] Chakrabarti, S., Dom, B., Gibson, D., Kleinberg, J., Raghavan P., and Rajagopalan, S. 1998 "Automatic Resource Compilation by Analyzing Hyperlink Structure and Associated Text" *Proc. of 7th World Wide Web Conference*, pp. 65–74.

[7] Chu, H. and Rosenthal, M. 1996 "Search Engines for the World Wide Web: A Comparative Study and Evaluation Methodology." *Proc. of ASIS 1996 Annual Conference.*

[8] Cronin, B. and Snyder, B. 1996 "Citation Indexing's Achilles Heel? Evaluative Bibliometrics and Non Coverage of the Monographic Literature." `www.slis.indiana.edu/Research/cronin-achilles.html`

[9] Croft, W.B., Cook, R., and Wilder, D. 1995. "Providing Government Information on the Internet: Experience with 'THOMAS'." *U. of Mass. Technical Report 95–45.*

[10] Cutting, D.R., Karger, D.R., Pedersen, J., and Tukey, J.W. 1992. "Scatter/Gather: A Cluster-Based Approach to Browsing Large Document Collections." *Proc. of ACM SIGIR '92.*

[11] Ding, W. and Marchionini. G. 1996 "Search Engines for the World Wide Web: A Comparative Study and Evaluation Methodology." *Proc. of ASIS 1996 Annual Conference.*

[12] Efthimiadis, E.N. 1993 "A User-Centered Evaluation of Ranking Algorithms for Interactive Query Expansion", *Proc. of ACM SIGIR '93*, pp. 146–159.

[13] Excite, `www.excite.com/`

[14] Golub, G., Van Loan, C. F., "Matrix Computations", Johns Hopkins University Press, Baltimore, 1989.

[15] Harman, D.K. 1988 "Towards Interactive Query Expansion." *Proc. of ACM SIGIR '88* pp. 321–331.

[16] Infoseek, `www.infoseek.com/`

[17] Harman, D.K. 1995 "The TREC Conferences" R. Kuhnlen and M. Rittberger (Eds.) *Hypertext – Information Retrieval – Multimedia: Synergieeffekte Elektronischer Informationssysteme, Proc. of HIM '95* pp. 9–28.

[18] Hearst, M. 1997 "Distinguishing between Web Data Mining and Information Access: Position Statement." *KDD '97 Panel on Web Data Mining.*

[19] Hearst, M. and Karadi, C. 1997 "Cat-a-Cone: An Interactive Interface for Specifying Searches and Viewing Retrieval Results using a Large Category Hierarchy." *Proc. of ACM SIGIR '97*, pp. 246–255.

[20] Karlin, S., Taylor, H. M., "A first course in stochastic processes", Academic Press, London, 1975.

[21] Kleinberg, J. 1998. "Authoritative sources in a hyperlinked environment." *Proc. of 9th ACM-SIAM Symposium on Discrete Algorithms.* Also appeared as IBM Research Report RJ 10076, May 1997.

[22] Larson, R.R. 1996 "Bibliometrics of the World Wide Web: An Exploratory Analysis of the Intellectual Structure of Cyberspace." *Proc. of ASIS '96 Annual Conference.*

[23] Marchionini, G. 1992. "Interfaces for End-User Information Seeking." *Journal of the American Society for Information Science*, 43(2):156–163.

[24] Magennis, M. and van Rijsbergen, C.J. 1997 "The Potential and Actual Effectiveness of Interactive Query Expansion." *Proc. of ACM SIGIR '97*, pp. 324–332.

[25] Page, L. 1997 "PageRank: Bringing Order to the Web." *Stanford Digital Libraries Working Paper*, 1997–0072.

[26] Pirolli, P., Pitkow, J., and Rao, R. 1996 "Silk from a sow's ear: Extracting usable structures from the Web." *Proc. of ACM SIGCHI '96*, pp. 118–125.

[27] Porter, M.F. 1980 "An Algorithm for Suffix Stripping." *Program*, 14, 130–137.

[28] Salton, G. and Buckley, C. 1988. "Term-Weighting Approaches in Automatic Text Retrieval." *Information Processing and Management*, 24(5), 513–23.

[29] Vélex, Weiss R., Sheldon M. A., Gifford, D. K. 1997. "Fast and Effective Query Refinement." *Proc. of ACM SIGIR '97*, pp. 6–15.

[30] Yahoo!. `www.yahoo.com/`

9 Appendix

We prove that the connectivity analysis aglrotihm terminates.

Lemma 1 *The improved connectivity analysis algorithm terminates, i.e., the H and A vectors eventually converge.*

Proof. Let $|N|$ be the size of the neighborhood graph. Let $B = (b_{nm})$ be a matrix such that for all $1 \leq n \leq |N|$ and $1 \leq m \leq |N|$, $b_{nm} = authority\_weight(n, m)$ and let $C = (c_{nm})$ be a matrix with $c_{nm} = hub\_weight(m, n)$ for all $1 \leq n \leq |N|$ and $1 \leq m \leq |N|$. Then steps (5) and (6) can be rewritten as:

(5) $A := BH$
(6) $H := CA$

Let $D = CB$. Every entry of D is non-negative. Since every node n is incident to an edge, $d_{nn} > 0$ for every n. Note that $authority\_weight(n, m) > 0$ if and only if $hub\_weight(n, m) > 0$, i.e., $b_{nm} > 0$ if and only if $c_{mn} > 0$. Thus, $d_{mn} > 0$ if and only if $d_{nm} > 0$.

Consider $D^{|N|}$. There exists a permutation of the rows of $D^{|N|}$ such that the resulting matrix has the following block-diagonal shape for some $l > 0$:

$$\begin{array}{l} \boxed{D_1}\ 0 \cdots\cdots\cdots 0 \\ 0 \cdots 0\ \boxed{D_2}\ 0 \cdots\cdots 0 \\ \qquad\qquad\qquad \ddots \\ 0 \cdots\cdots\cdots 0\ \boxed{D_l} \end{array}$$

Each block, i.e., matrix D_i with $1 \leq i \leq l$ is square and all its entries are positive. Thus, by the Frobenius theory of positive matrices (see e.g. [20]), it follows that the first eigenvalue $\lambda_0(D_i) > |\lambda(D_i)|$, where $\lambda(D_i)$ is any other eigenvalue of D_i and there exists an eigenvector X_0 for $\lambda_0(D_i)$ with positive entries. If this condition is fulfilled then $D_i^k \cdot Z$ converges [14], where Z is a vector with each coordinate equal to 1. Since the value of H after the k-th iteration equals $D^k \cdot Z$, it follows that the H vector and thus also the A vector converges. $\blacksquare$