Project Number 001907

DELIS
Dynamically Evolving, Large-scale Information Systems

Integrated Project
Member of the FET Proactive Initiative Complex Systems

Deliverable D6.1.2

Final Report on
Models of Collaborative Peer Behavior and
Strategies for Global Search
Start date of the project: January 2004
Duration: 48 months
Project Coordinator: Prof. Dr. math. Friedhelm Meyer auf der Heide
Heinz Nixdorf Institute, University of Paderborn, Germany
Due date of deliverable: December 2005
Actual submission date: January 2006
Dissemination level: PU – public

Work Package 6.1: Models and Strategies for Collaborative Web Information Search

Participants: Max Planck Institute for Computer Science (MPII), Saarbrücken, Germany
Computer Science Institute (CTI), Patras, Greece
Telenor Communication AS (Telenor), Oslo, Norway

Authors of deliverable: Gerhard Weikum (weikum@mpi-sb.mpg.de)
Holger Bast (bast@mpi-sb.mpg.de)
Geoffrey Canright (geoffrey.canright@telenor.com)
Peter Triantafillou (peter@ceid.upatras.gr)
This report presents the final results on models for peer collaboration and search strategies in peer-to-peer (P2P) systems. It replaces the intermediate report, Deliverable D6.1.1, and extends this prior deliverable in the following ways:

- improvements and extensions of the work on efficient top-k search and concept-based retrieval, which led to publications in the prestigious conferences SIGIR 2005 and VLDB 2005 [4, 62, 48],
- new work on integrated quality-novelty-aware query routing, which led to publications in SIGIR 2005 and EDBT 2006 [9, 13] and early work on semantic overlay networks [55], and
- development of a novel set of techniques for distributed link analysis, and new work on decentralized computation of PageRank authority scores in P2P networks with overlapping peer content [53].

1 Introduction and Overview

One of the major goals of DELIS SP6 is to develop foundations for collaborative Web information search in an Internet-scale peer-to-peer (P2P) system. We are aiming at a P2P system where each peer has a full-fledged Web search engine, including a crawler and an index manager. The crawler may be thematically focused or crawl results may be postprocessed so that the local index content reflects the corresponding user's interest profile. With such a highly specialized and personalized “power search engine” most queries should be executed locally, but once in a while the user may not be satisfied with the local results and would then want to contact other peers. A “good” peer to which the user’s query should be forwarded would have thematically relevant index contents, which could be measured by statistical notions of similarity between peers. Both query routing and the formation of “semantic overlay networks” could greatly benefit from collective human inputs in addition to standard statistics about terms, links, etc.: knowing the bookmarks and query logs of thousands of users would be a great resource to build on. Note that this notion of Web search includes ranked retrieval and thus is fundamentally more difficult than Gnutella-style file sharing or simple key lookups via distributed hash tables. Further note that, although query routing in P2P Web search resembles earlier work on metasearch engines and distributed information retrieval, it is much more challenging because of the large scale and the high dynamics of the envisioned P2P system with thousands or millions of computers and users.

So far the following results have been achieved on the research issues outlined above:

- A novel family of query routing strategies for P2P Web search has been developed in joint work of MPII and CTI. The particular strength of these methods is that they consider both the content quality of peers and the overlap of the peers’ contents with the data that the query initiator already knows. Thus, we provide an integrated quality-novelty-aware strategy (IQN) [9, 13].

- Additional query routing methods have been developed jointly by MPII and CTI with the particular goal of exploiting user-profile information at the level of individual users or entire communities. These techniques harness user bookmarks or query logs [7, 12].

- An efficient algorithm for top-k query processing, coined Prob-sorted, has been developed at MPII. It utilizes score distribution statistics for score predictions to prune result candidates early, thus outperforming the so far best known methods by an order of magnitude with small and acceptable loss in precision and recall [61].
• New strategies for the scheduling of index accesses in top-k query processing have been developed at MPII. These methods experimentally showed near-optimal performance for a wide range of data and load characteristics [6].

• A distributed top-k algorithm, coined KLEE, has been developed in joint work of MPII and CTI. KLEE includes optimizations for network latency and bandwidth consumption and gains dramatic performance advantages over the prior work [48].

• Retrieval methods that make use of the concepts underlying large collections of text have been investigated and improved at MPII. One approach is to derive concepts from the raw data by spectral analysis or other matrix decomposition techniques, without training or dictionaries. Novel insights have been found into what makes such approaches work in practice. Various new methods are derived, which improve search results, have a better theoretical foundation, and are well-suited for a P2P setting [3, 4, 5]. In addition, a new method for query expansion based on thesauri and ontological relationships has been developed [62]. This method is particularly efficient and self-tuning in the sense that it automatically avoids over-expansion and thus reduces the danger of topic drifts.

• A new method, coined QRank, for incorporating query-log and click-stream information into Web page ranking has been developed at MPII. It is based on a Markov model that extends the traditional Page-Rank approach by query nodes and edges for query refinement, query-result clicking, and node-pair similarity [46].

• Link analysis techniques have been studied at Telenor, with new insights on the subtle but important differences between directed vs. undirected graphs and fully connected vs. multi-component graphs [24]. This work has led to several novel approaches to link analysis, which are distinct from the well-known PageRank and HITS methods [21]. These methods have shown very good results in limited tests so far.

• A fully distributed and asynchronous implementation of the Telenor link-analysis techniques (and also of PageRank) has been developed jointly by Telenor and UBologna [39]. The method has been shown to be highly robust against noise, such as message delay jitter and message loss.

• A new method, coined JXP, for decentralized link analysis on arbitrarily overlapping graph fragments stored at different peers has been developed at MPII for P2P-style computation of PageRank authority scores [53, 54].

2 Main Results

2.1 Query Routing

2.1.1 State of the Art

Query routing aims to select the most promising peers among a large set of candidates, for executing a given query. This problem is also known as database selection or resource selection in the information retrieval (IR) literature. However, collaborative P2P search is substantially more challenging than the traditional setup for distributed IR over a small federation of sources such as digital libraries, as these prior approaches mediate only a small and rather static set of underlying nodes, as opposed to the large scale and high dynamics of a P2P system.

The literature on P2P request routing has mostly focused on simple key lookups. Even techniques that consider multi-dimensional keyword queries and the formation of “semantic overlay networks” (e.g., [68, 28, 1]) disregards the need for ranked retrieval based on relevance scoring (as opposed to
Boolean retrieval where data items either satisfy search conditions or not but are not further discriminated regarding their relevance). On the other hand, prior research on distributed information retrieval and metasearch engines [23, 49] has addressed the ranking of data sources and also the reconciliation of search results from different sources. GLOSS [33] and CORI [22] are the most prominent distributed IR systems, but neither of them aimed at very-large-scale, highly dynamic, self-organizing P2P environments (which were not an issue at the time these systems were developed).

Recent approaches to query routing [45, 51, 29] consider larger federations of data sources, but they use computationally intensive techniques based on statistical language models for assessing a peer’s quality with regard to a given query. It is an open issue to what extent such techniques scale and can cope with high dynamics.

### 2.1.2 Integrated Quality-Novelty-Aware Query Routing

The rationale for the query routing strategies developed in DELIS is based on the following three observations:

1. The query initiator should prefer peers that have similar interest profiles and are thus likely to hold thematically relevant information in their indexes.

2. On the other hand, the query should be forwarded to peers that offer complementary results. If the remote peer returns more or less the same high-quality results that the query initiator already obtained from its own local index, then the whole approach of collaborative P2P search would be pointless.

3. Finally, all parties have to be cautious that the execution cost of communicating with other peers and involving them in query processing is tightly controlled and incurs acceptable overhead.

We address the first two points by defining the benefit that a remote peer offers for the given query to be proportional to the thematic similarity of that peer and the query initiator and inversely proportional to the overlap between the two peers in terms of their local index contents. As for the third point, we aim to estimate the load and performance behavior of different peers, and then consider the benefit/cost ratio as the main criterion for making query routing decisions. For the benefit and cost estimation we utilize statistical summaries and metadata about peers. This information is maintained in a decentralized directory implemented as a distributed hash table (DHT) [8]. The system architecture for this query routing framework is depicted in Figure 1.

![Figure 1: System Architecture for Query Routing Framework](image-url)
as the number of documents that the peer has for a given term, the average term frequency in these documents, and so on. Additionally, compact synopses about the identifiers (e.g., URLs), of the documents that each peer holds should be locally computed. These summaries are then posted to the distributed directory and conveniently accessible by every peer, with $O(\log N)$ communication costs where $N$ is the network size.

An integrated, quality-overlap-aware query routing method can now estimate the content richness of candidate target peers, in terms of the similarity of the peers’ contents to the given query (or the locally computed initial result), and the degree of novelty that a candidate peer would offer relative to the initial results that are already known to the query originator. We have developed an algorithm, coined IQN routing (for integrated quality and novelty) [13], that chooses target peers in an iterative manner, performing two steps in each iteration. First, the Select-Best-Peer step identifies the most promising peer regarding the product (or a weighted combination) of result quality and novelty. This step is driven by the statistical synopses that are obtained from the directory. Then, the Aggregate-Synopses step conceptually aggregates the selected peer’s content with the previously selected peers’ data collections (including the query originator’s own local collection). This aggregation is actually carried out on the compact synopses, not on the full data. The two-step selection procedure is iterated until some performance and/or quality goals are satisfied (e.g., a predefined number of peers is reached, or a desired recall is estimated to be achieved).

The efficiency and effectiveness of the IQN routing method crucially depends on appropriately designed compact synopses for the peer-collection statistics. The synopses should be small to keep network bandwidth consumption and storage costs low, yet they must offer low-error estimations of quality and novelty (or overlap) measures. Furthermore, to support the Aggregate-Synopses step, it must be possible to combine multiple synopses published by different peers in order to derive a synopsis for the aggregated collection. We have developed detailed methods for using Bloom filters [17], hash sketches [32], and min-wise independent permutations [19] as the basis of our peer-summary synopses. Extensive experiments have shown that our methods do indeed combine very low overhead with high accuracy for quality-novelty estimation, and the IQN query routing strategy outperforms standard approaches such as CORI by a significant margin [9, 13].

### 2.1.3 Query Routing Driven by User Information

In estimating the notion of a peer’s benefit for a given query, more advanced information about user behavior such as bookmarks or query logs may be utilized. We have developed two methods [7, 12] along these lines; experimental studies are in preparation.

Advanced information retrieval techniques based on statistical language models measure the similarity between text sources (e.g., between a document and a query or between two peers’ contents) by the Kullback-Leibler divergence of the underlying word distributions. This captures the likelihood of one source generating the other (e.g., a document’s language model generating the query). This approach has a potential for improving the peer-quality assessment of the query routing procedure. However, it is computationally more expensive than the term-frequency-oriented methods of the previous subsection. To reduce the overhead, we have developed a method [7] where the similarity computation is based on peer-specific bookmarks rather than the entire contents of peers. Here we view the index contents of a peer as being generated by the peer’s bookmarks, which served as seeds for the peer’s Web crawls and possibly also as training data for a thematically focused crawler [26, 57]. Bookmarks can also be used as a light-weight estimator for the overlap among peers. Finally, bookmarks reflect thematic user profiles in a compact manner, and thus provide a great potential for personalization and leveraging community recommendations.

Another approach that we have been investigating is to exploit query logs for identifying highly related words, and these word combinations can in turn trigger the computation of additional summaries that are posted to the P2P directory to improve future query routing decisions. For example,
consider the three-keyword query “native American music”. A standard approach would aim to identify the best peers for each of the three terms separately, and then combine the candidate peer lists to identify the most promising peers to which the query should be forwarded. The result may be mediocre as the best peers for the entire query are not necessarily among the top candidates for any of the individual terms. Our approach [12] monitors queries on the peers where they are initially submitted and posts them to the distributed directory, where they are aggregated so that interesting word combinations can be inferred (e.g., using frequent-itemset data mining algorithms). Then the responsible directory peers gather statistical summaries about the new word combinations and build up corresponding peer lists. When a query with the same or a sufficiently close set of terms is later asked by any peer, the query routing can draw on the extended directory information to select the best peers for term combinations. This way, the query routing procedure benefits from the knowledge of frequent queries. A salient property to emphasize is that this query-log-driven method does not incur any additional network traffic; all necessary information can be piggybacked on messages that need to be sent anyway (see [12] for details).

2.1.4 Semantic Overlay Networks

Query routing decisions are made at query run-time, so the cost of statistical comparisons is an issue that must carefully considered. For very low run-time overhead, various precomputations may be employed and leveraged. This leads to a form of semantic overlay network (SON) in addition or on top of the DHT-based overlay network that connects all peers. In a SON only peers are connected as immediate neighbors that have high likelihood of being beneficial to each other’s queries. Approaches for building and maintaining a SON include methods from the theory of random graphs [47] and methods inspired by social networks [35].

We have developed a new method, coined “p2pDating” [55], that combines elements from both of the above two paradigms. Peers meet with other peers by random selections, and exchange light-weight synopses. Each peer maintains a list of candidates in a ranked list of “interestingness” based on the synopses. A peer then has real “dates” with the most promising peers from the candidate list, and exchanges more detailed information for better quality and novelty assessment. Finally, a connection in the SON is established with a peer if the benefit from that peer is better than that of the lowest-ranked peer among the current neighbors (and the connection to the latter is removed). This form of p2pDating is implemented as a continuous, low-priority background process. It leads to continuous adaptation of the network connections in the SON to the evolution of the network. Preliminary experiments indicate the viability of this approach; more comprehensive studies are in progress.

2.1.5 Experimental Testbed Minerva

The strategies outlined above as well as various alternative strategies from the prior literature have been implemented in the Minerva testbed developed at MPII [10, 11]. Minerva uses a Chord-style distributed hash table (DHT) [58] as an overlay network. Each peer has its own search engine, along with a crawler and a local index. Peers post statistical summaries to the overlay network; they can also post bookmarks and information about their query logs and click streams at their discretion. All this information is managed as a decentralized directory, using the DHT, and is utilized by the query routing strategies. Minerva is implemented in Java and fully operational. A first release of the Minerva software has been made available as DELIS Deliverable D6.5.2 in July 2005.
2.2 Efficient Top-k Query Processing

2.2.1 State of the Art

Top-k query processing is a fundamental cornerstone for similarity search on multimedia data, ranked retrieval on text and semi-structured documents in digital libraries and on the Web, network and stream monitoring, collaborative recommendation and preference queries, and ranking of query results on structured data sources in general. It aggregates scores for different search terms or attribute values using a monotonic aggregation function such as weighted summation, and returns the top-ranked data items as the query result. Scores are usually precomputed features of different aspects of a data item, e.g., color distributions in images, access frequencies in Web server logs, or word occurrence statistics in text documents. The state-of-the-art algorithm for top-k queries on multiple index lists, each sorted in descending order of relevance scores, is the Threshold Algorithm, TA for short [31, 34, 50]. It is applicable to both structured data such as product catalogs and text documents such as Web data. In the latter case, the fact that TA performs random accesses on very long, disk-resident index lists (e.g., all URLs or document ids for a frequently occurring word), with only short prefixes of the lists in memory, makes TA much less attractive, however.

In such a situation, the TA variant with sorted access only, coined NRA (no random accesses), stream-combine, or TA-sorted in the literature, is the method of choice [31]. TA-sorted works by maintaining lower bounds and upper bounds for the scores of the top-k candidates that are kept in a priority queue in memory while scanning the index lists. The algorithm can safely stop when the lower bound for the score of the rank-k result is at least as high as the highest upper bound for the scores of the candidates that are not among the current top-k. Unfortunately, albeit theoretically instance-optimal for computing a precise top-k result [31], TA-sorted tends to degrade in performance when operating on a large number of index lists, which happens when user queries are automatically expanded based on ontologies, user profiles, or relevance feedback.

2.2.2 The Prob-sorted Algorithm

Statistics about the score distributions in the various index lists and some probabilistic reasoning help to overcome this efficiency problem and gain performance. In TA-sorted a top-k candidate \( d \) that has already been seen in the index lists in \( E(d) \subseteq [1..m] \), achieving score \( s_j(d) \) in list \( j \) (\( 0 < s_j(d) \leq 1 \)), and has unknown scores in the index lists \( [1..m] - E(d) \), satisfies:

\[
\text{lowerb}(d) = \sum_{j \in E(d)} s_j(d) \leq s(d) \leq \sum_{j \in E(d)} s_j(d) + \sum_{j \notin E(d)} \text{high}_j = \text{upperb}(d)
\]

where \( s(d) \) denotes the total, but not yet known, score that \( d \) achieves by summing up the scores from all index lists in which \( d \) occurs, \( \text{lowerb}(d) \) and \( \text{upperb}(d) \) are the lower and upper bounds of \( d \)'s score, and \( \text{high}_j \) is the score that was last seen in the scan of index list \( j \), upper-bounding the score that any candidate may obtain in list \( j \). A candidate \( d \) remains a candidate as long as \( \text{upperb}(d) > \text{lowerb}(\text{rank-k}) \) where rank-k is the candidate that currently has rank \( k \) with regard to the candidates' lower bounds (i.e., the worst one among the current top-k). Assuming that \( d \) can achieve a score \( \text{high}_j \) in all lists in which it has not yet been encountered is conservative and, almost always, overly conservative. Rather we could treat these unknown scores as random variables \( S_j \) (\( j \notin E(d) \)), and estimate the probability that \( d \)'s total score can exceed \( \text{lowerb}(\text{rank-k}) \). Then \( d \) is discarded from the candidate list if

\[
P[\text{lowerb}(d) + \sum_{j \notin E(d)} S_j > \text{lowerb}(\text{rank-k})] < \delta
\]

with some pruning threshold \( \delta \). Technically, this score prediction requires computing the convolution of the score distributions in the yet to be scanned parts of the index lists. This can be implemented, for example, using histograms, fitting appropriate parametric distributions such as Poisson mixes,
or using Laplace transforms of the underlying score distributions and Chernoff-Hoeffding bounds for the tail of the convolution. Figure 2 illustrates the probabilistic score predictor for early candidate pruning.

![Inverted Index](image1)

![Convolution](image2)

Figure 2: Probabilistic score predictor for early candidate pruning

This probabilistic interpretation makes some small, but precisely quantifiable, potential error in that it could dismiss some candidates too early. Thus, the top-k result computed this way is only approximate. However, the loss in precision and recall, relative to the exact top-k result using the same index lists, is stochastically bounded and can be set according to the application’s needs. A value of $\delta = 0.1$ seems to be acceptable in most situations. Details of this Prob-sorted method can be found in [61]. Experiments with the TREC-12 .Gov corpus and the TREC-13 Terabyte corpus (http://trec.nist.gov/), the INEX benchmark for XML information retrieval (http://inex.is.informatik.uni-duisburg.de:2003/), and the IMDB data collection (www.imdb.com) have shown that such a probabilistic top-k method gains about a factor of ten (and sometimes more) in run-time compared to a highly tuned version of TA-sorted.

The algorithm for approximate top-k queries with probabilistic guarantees is a versatile building block for ranked retrieval on text and semistructured data such as Web or XML documents. In combination with ontology-based query relaxation, for example, expanding a phrase query like "top-k query" into "top-k query or ranked retrieval or score aggregation", it can add index lists dynamically and incrementally, rather than having to expand the query upfront based on thresholds. To this end, the algorithm considers the ontological similarity $sim(i, j)$ between concept $i$ from the original query and concept $j$ in the relaxed query, and multiplies it with the $high_j$ value of index list $j$ to obtain an upper bound for the score (and characterize the score distribution) that a candidate can obtain from the relaxation $j$. This information is dynamically combined with the probabilistic prediction of the other unknown scores and their sum. The incremental algorithm outperforms the traditional techniques for query expansion by a factor of 3 to 50 in run-time; at the same time, it avoids the danger of topic drifts caused by over-expansion and eliminates the need for tuning expansion thresholds [62].

### 2.2.3 Distributed Top-k Algorithms for P2P Systems

In a P2P network a top-k query can be collaboratively processed by a set of peers each of which holds one or more index lists for the search terms or attribute values in a query. The query-initiating peer serves as a per-query coordinator and aggregates information about top-k candidates. While such algorithms are efficient in terms of the peers’ local resource consumption, they do not pay sufficient attention to the communication costs of the computation.

In joint work of CTI Patras and MPII Saarbrücken, a new family of algorithms, coined KLEE
has been developed to address the networking costs of top-k query algorithms. KLEE aims to minimize network latency, network bandwidth consumption, and the local work of the participating peers. To this end, it proceeds in a fixed number of phases to ensure bounded latency. In this regard, we follow the recent work of [25], but we differ significantly in the phases themselves and introduce various novel considerations.

The first phase of KLEE gathers an initial set of top-k candidates from the peers' index lists and derives a crude threshold for the final top-k result. Along with the candidates, peers send summary information in the form of score-distribution histograms and Bloom filters for data items that locally fall into high-score histogram cells. In the second phase the coordinator performs a benefit/cost estimation for a possible additional message round that would collect further Bloom-filter information to improve the knowledge about top-k candidates. In the optional third phase this information is sent to the coordinator, which in turn prepares a refined list of candidates. In the last phase the peers are requested to send the missing scores for all data items in the candidate list above some lower bound of relevant scores. Because of the approximate nature of histograms and Bloom filters, KLEE computes an approximate top-k results, but similarly to the Prob-sorted algorithm described above, the loss in precision and recall is small and controllable.

KLEE has been implemented in the Minerva testbed and intensively evaluated on various real-life datasets and query benchmarks. It outperforms both the distributed version of TA-sorted with batching and the TPUT method of [25] by one to two orders of magnitude when three or more peers participate in a query. This impressive performance gain is achieved by reducing both the network bandwidth consumption and the local work at the index-scanning peers. Precision and recall are 80 percent or higher, and the score and rank error measures indicate that the approximate top-k results are as good as the exact ones from a user acceptance viewpoint. This demonstrates the advantages of KLEE's design for flexible control over different cost and query-result quality metrics, with excellent performance in terms of the quality/cost ratio.

2.3 Concept-based Retrieval

2.3.1 State of the Art

Top-k queries usually operate on what is known as the bag-of-words or vector space model, where documents as well as queries are represented as vectors, with each dimension corresponding to a word or term that occurs in the document collection. This space may be enhanced by semantic information, as given, for example, by XML tags, but it typically stays on a term-centric basis.

One effective way to further boost search result quality is to switch to a representation of queries and documents in terms of the semantic concepts underlying a collection. Concept-based search is able to detect similarities that are beyond direct term matches, e.g., a query on “large-scale information systems” would also return a document entitled “DELIS research results”, even when it mentions none of the query words explicitly. The methods we consider could easily detect this similarity from the observation that in sufficiently many other documents the acronym DELIS will be mentioned together with its verb form. Our research is concerned mainly with such unsupervised concept-based search, where concepts are derived from the raw data itself, without training or dictionaries.

A major obstacle in integrating concept-based search in a P2P scenario is that its foundations are still little understood even in the centralized case. A confusing variety of methods exist, each of which works well on particular kinds of collections, but none of which can be called a clear winner. For example, latent semantic indexing (LSI), the grandmother of many concept-based retrieval schemes, has just recently been adapted for a P2P setting [59], although its usefulness for large document collections is questionable already in the centralized case.

Our research goal within DELIS is hence twofold [3]: (i) to develop practically useful algorithms for concept-based search that have a strong theoretical foundation, and (ii) to bring these algorithms
to the P2P world.

2.3.2 Concept-Based Retrieval Based on Spectral Analysis

One of the most prominent approaches to unsupervised concept-based retrieval is to derive the concepts from a spectral analysis of the term-document matrix, that is, from some sort of eigenvector computation. These include LSI and its many variants, and Google’s PageRank can be viewed as an eigenvector analysis, too. But even for these methods, despite their mathematical flavor, it it still little understood under which circumstances they work how well and why.

We have made major advances in this respect. In [4], we work out the essence of spectral retrieval with a number of surprising insights. One result is a simple parameterless scheme that outperforms all of the previous spectral retrieval schemes that commit themselves to a fixed number of dimensions (the concepts). We achieve this result, even when giving the fixed-dimension schemes the unfair advantage that they may try every possible number of concepts and every one from a broad selection of standard term or document normalizations, and then pick the one which gives their best result.

Our work is built on the observation that all spectral retrieval schemes, no matter which variant, essentially work by performing a document expansion: each term pair is (implicitly) assigned a relatedness score and a document will match a query well to the extent that the document contains terms which are, by these scores, related to terms in the query. Note that if each term were related only to itself, this would result in retrieval according to the standard vector space model.

Our main trick is the following: instead of looking at the relatedness scores for all term pairs for a fixed number of concepts — this is what all previous methods implicitly do — look at the relatedness score of a fixed term pair for all possible number of concepts. Plotting these scores gives us, for each term pair, what we call its synonymy graph. These synonymy graphs turn out to be key to understanding how and why spectral retrieval works.

![Figure 3: The synonymy graph for two pairs of related terms (a) + (b), and for a pair of unrelated terms (c), all from a collection of computer science abstracts. On the x-axis the number of concepts is varied, the y-axis gives the relatedness score.](image)

We find that what distinguishes intuitively related from intuitively unrelated terms is not any relatedness score for a particular fixed dimension, but the shape of the synonymy graph. Graphs for related terms have a positive gradient up to some point, then drop, and finally go up again. Graphs for unrelated terms fluctuate around zero, with most (but not necessarily all) of their absolute values at least an order of magnitude below those reached by graphs for related terms. We are able to derive these properties from a mathematical model, and these insights are confirmed by extensive experiments.

The synonymy graphs reveal that no fixed choice for the number of concepts can be good for all term pairs. An extreme, but not untypical example is given in Figure 3(a) and (b). Both term pairs are highly related — indeed, one pair being the singular and plural of the same word and the other pair forming a common phrase, they are related in the strongest sense possible — and this relatedness
indeed shows in the form of the synonymy graphs. But any fixed choice of dimension can give a high relatedness score to only one of the two term pairs.

The synonymy graphs give us new insights into the effects of the various (heuristic) data normalizations, which are known to be crucial for good retrieval performance. From all these insights, we finally derive the aforementioned parameterless algorithm that is superior to all of the previous fixed-dimensions schemes even at their best parameter settings. Together with our theoretical findings this gives strong evidence that we indeed identified the major principle that makes spectral retrieval work.

2.3.3 Insights from Viewing Ranked Retrieval as Rank Aggregation

In [5], we take a broader perspective and study the common principles behind ranked retrieval in general. We view various methods in a unifying manner, namely as processes of combining query-independent rankings that were precomputed for certain attributes. In particular, we apply this view to various established concept-based retrieval schemes. While the research described in the previous subsection assumes linear mappings from term to concept space, here also non-linear methods are included. We pay particular attention to one of the most prominent and promising non-linear concept-based retrieval schemes, named probabilistic latent semantic indexing (PLSI) [36].

Non-linear methods have fundamental advantages over linear schemes; for example, they have the ability to deal with polysems, which schemes based on a linear mapping from term to concept space can only do to a limited extent. This has been vaguely commented on by various authors; in [5], we turn it into a precise statement.

It seems that the extended capabilities of non-linear methods do not come for free. Indeed, and not surprisingly, the underlying optimization problems are significantly harder than for linear methods — for a characterization of the differences see our research within SP3, the subproject on large-scale optimization. The preprocessing time required for a method like PLSI is an order of magnitude higher than for a method like LSI. For both approaches, the preprocessing is an inherently global computation, which on one hand gives these methods their power, but on the other hand makes an integration into a distributed P2P setting difficult.

In [5], we replace PLSI's compute-intensive part by a simple set of document rankings, which need not even be precomputed. The trick lies in an unusual choice for the attributes. On a number of test collections, we find the retrieval performance for this new method to be at least as good and sometimes even better than that of PLSI and a number of other non-linear matrix decomposition schemes. Another advantage of the new method is that it is much more suitable for a distributed P2P setting. We are currently working on the integration of our method into the Minerva testbed, which is being developed in DELIS WP6.5.

2.3.4 Efficient and Self-tuning Query Expansion

Query expansion is an alternative to spectral-analysis techniques: keyword queries are automatically expanded to include semantically related words and phrases. State-of-the-art approaches use one or a combination of the following sources to generate additional query terms: thesauri such as WordNet with concept relationships and some form of similarity measures, explicit user feedback or pseudo relevance feedback, query associations derived from query logs, document summaries such as Google top-10 snippets, or other sources of term correlations. In all cases, the additional expansion terms are chosen based on similarity, correlation, or relative entropy measures and a corresponding threshold. For difficult retrieval tasks, query expansion can improve precision@top-k, recall, as well as uninterpolated mean average precision (MAP) by a significant margin (see, e.g., [41, 44]). However, it also faces major problems: 1) The threshold for selecting expansion terms needs to be carefully hand-tuned, and this is highly dependent on the application’s corpus and query workload. 2) An inappropriate choice of the sensitive expansion threshold may result in either not achieving the desired improvement
in recall (if the threshold is set too conservatively) or in high danger of topic dilution (if the query is expanded too aggressively). 3) The expansion may often result in queries with 50 or 100 terms, which in turn leads to very high computational costs in evaluating the expanded queries over inverted index lists.

We have developed a novel algorithm that addresses the above three issues and provides an efficient, scalable, and largely self-tuning solution [62]. The algorithm builds on our top-k query processing algorithms (see above) and uses an incremental-merge technique for inverted index lists. The query expansion itself uses a thesaurus with statistically weighted similarities between concepts. The mapping of words and phrases onto concepts also uses similarity measures and EM-style statistical learning procedures [60, 38].

Our key techniques for robustness and efficiency are to avoid aggregating scores for multiple expansion terms of the same original query term and to avoid scanning the index lists for all expansion terms. For example, when the term “disaster” in a query “transportation tunnel disaster” is expanded into “fire”, “earthquake”, “flood”, etc., we do not count occurrences of several of these terms as additional evidence of relevance. Rather we use a score aggregation function that counts only the best score of a document for all expansion terms of the same original query term, weighted by the similarity (or correlation) of the expansion term to the original term. Furthermore, and most importantly for efficiency, we open scans on the index lists for expansion terms as late as possible, namely, only when the best possible candidate document from a list can achieve a score contribution that is higher than the score contributions from the original term’s list at the current scan position or any list of expansion terms with ongoing scans at their current positions. The algorithm conceptually merges the index lists of the expansion terms with the list of the original query term, in an incremental on-demand manner during the run-time of the query. For further speed-up probabilistic score estimation can be used, considering both score distributions and selectivities of terms. The algorithm has achieved excellent performance results with good query result quality on advanced benchmarks like the TREC Robust benchmark, TREC Terabyte benchmark, and the XML INEX benchmark.

2.4 Exploiting Query Logs and Click Streams

Information about user behavior is a rich source to build on. This could include relatively static properties like bookmarks or embedded hyperlinks pointing to high-quality Web pages, but also dynamic properties inferred from query logs and click streams. For example, Google’s PageRank views a Web page as more important if it has many incoming links and the sources of these links are themselves high authorities. This rationale can be carried over to analyzing and exploiting entire surf trails and query logs of individual users or an entire user community. These trails, which can be gathered from browser histories, local proxies, or Web servers, capture implicit user judgements. For example, suppose a user clicks on a specific subset of the top 10 results returned by a search engine for a query with several keywords, based on having seen the summaries of these pages. This implicit form of relevance feedback establishes a strong correlation between the query and the clicked-on pages. Further suppose that the user refines a query by adding or replacing keywords, e.g., to eliminate ambiguities in the previous query. Again, this establishes correlations between the new keywords and the subsequently clicked-on pages, but also, albeit possibly to a lesser extent, between the original query and the eventually relevant pages.

Observing and exploiting such user behavior could be a key element in adding more “semantic” or “cognitive” quality to a search engine. The literature contains interesting work in this direction (e.g., [30, 67]), but is rather preliminary at this point. Perhaps, the difficulties in obtaining comprehensive query logs and surf trails outside of big service providers is a limiting factor in this line of experimental research. Our own recent work generalizes the notion of a “random surfer” into a “random expert user” by enhancing the underlying Markov chain to incorporate also query nodes and transitions from queries to query refinements as well as clicked-on documents. Transition probabilities are derived
from the statistical analysis of query logs and click streams. The resulting Markov chain converges to stationary authority scores that reflect not only the link structure but also the implicit feedback and collective human input of a search engine’s users [46].

2.5 Decentralized Link Analysis

PageRank [18] and HITS [40] are the seminal methods for link analysis on the Web graph, which can be used to derive authority scores for Web pages that reflect query-independent importance and community endorsements. [42] provides a good in-depth survey of follow-up and related work. Recently, various distributed versions of link analysis methods have been proposed (e.g., [20, 63, 66]), but they all assume that the global graph is partitioned into disjoint fragments across Web hosts or peers.

JXP [53, 54] is a new algorithm, developed at MPII, for dynamically computing, in a decentralized P2P manner, global authority scores when the Web graph is spread across many autonomous peers. In contrast to the above related work, the peers’ graph fragments may overlap arbitrarily, and peers are (a priori) unaware of other peers’ fragments. The main idea of JXP is as follows. Each peer computes the authority scores of the pages that it has in its local index, by locally running the standard PageRank algorithm. A page may be known and indexed by multiple peers, and these may have different scores for that same page. A peer gradually increases its knowledge about the rest of the network by meeting with other, randomly chosen, peers and exchanging information. To improve the initial authority scores and approximating the global authority of pages, a peer uses what it learns from the other, randomly met, peers, combined with its own local information, for recomputing the scores. The local computations are very space-efficient (as they require only the local graph and the authority-score vector), and fast (as they operate on much smaller graph fragments than a server-side global PageRank computation). They leverage results on Markov-chain state lumpability (i.e., state aggregation into superstates). Experiments have indicated that the resulting JXP scores quickly converge to the true, global PageRank scores.

Complementary to the above kind of decentralized approaches are fundamental studies of link analysis techniques, particularly: (i) the subtle but important differences between authority and hub measures (PageRank is authority); (ii) the nontrivial effects of normalizing (or not) the matrix which gives the importance measure via its eigenvector; and (iii) the effects of various types of “sink remedies”, which are needed for hyperlinked graphs such as the Web to correct the problems which arise in link analysis from sets of pages which collectively have no outlinks (and hence are, collectively, a sink).

Telenor has started investigating this area systematically [21], building on earlier work on undirected graphs [24]. For undirected graphs, one finds that matrix normalization gives the rather trivial result that each node’s importance (authority) score is directly proportional to its degree. In contrast, for directed graphs (such as the Web), matrix normalization which is used in the PageRank method gives useful and nontrivial results. However, the non-normalized case has received little attention. The work by Telenor [21] has shown that importance scores from non-normalized matrices can give very good results, equal to or better than those obtained from PageRank.

Furthermore, for directed graphs, one has the choice of computing authority scores, i.e., pages which are linked to by good pages, or hub scores, which give pages which point to good pages [40]. The HITS method [40] gives both, while PageRank [18] gives essentially an authority score. Telenor [21] has explored both types of scores for non-normalized matrices; this approach is however distinct from HITS. The HITS method uses symmetric, quadratic products (authority = $A^TA$ and hub = $AA^T$) of the hyperlink adjacency matrix $A$, while the Telenor work uses simply the asymmetric, non-normalized matrices $A^T$ (for authority scores) and $A$ (for hub scores).

A final aspect of link analysis for Web graphs is the problem of sinks. The PageRank method uses a “random surfer” operator, which amounts to laying in symmetric links between all pairs of pages,
each link with a small weight. The Telenor work [21] proposes two novel methods for removing sinks from graphs such as the Web graph; each of these methods uses vastly fewer ‘extra’ links than the PageRank method.

Limited tests of these new ideas have given the result that one of the new “sink-remedy” methods, in combination with a non-normalized matrix, gives the best results in terms of giving high ranking to “good” pages. PageRank gave intermediate results, while the HITS method gave the poorest results.

Finally, in collaboration with UBologna, Telenor has developed fully distributed versions [39] for many of these methods of link analysis: for PageRank and also for the non-normalized but non-quadratic matrices $A$ and $A^T$. These distributed versions so far have only implemented the random surfer sink remedy, since the other, new sink remedy methods are more challenging to implement in a distributed fashion. The resulting methods require no global coordination among peers, other than an agreed updating frequency. In particular, global synchronization of updates is not needed. The method has been shown to be robust against high levels of noise, such as message delay, message delay jitter, and message loss. Hence we now see the possibility of implementing new approaches to page ranking — approaches which may give better results than PageRank — in a fully distributed search engine.

3 Result Dissemination

The research in this WP has led to 16 publications [3, 4, 5, 7, 8, 9, 13, 21, 24, 38, 46, 48, 52, 53, 61, 62] in international conferences and refereed workshops in the years 2004 and 2005, and another 5 papers [6, 12, 21, 39, 54] are under submission for publication. The publication list includes 6 full papers in the prestigious, highly selective first-rate conferences VLDB [61, 48], SIGIR [4, 9, 62], and EDBT [13], all of which have acceptance ratios of 1:5 or higher. An overview of the results, jointly co-authored by the five partners MPII, CTI, UPB, Telenor, and UBologna, has also been presented at the European Conference on Complex Systems in November 2005 in Paris [64, 65]. All referenced papers are available on request from the coordinating site UPB.

MPII organized a summer school on information retrieval (ADFOCS 2004, September 6–10 in Saarbrücken, Germany); this was attended by 70 participants including several researchers from DELIS partners.

Selected results are implemented in the Minerva software, a prototype testbed for P2P Web search. A first release of Minerva has been made available as open source software as Deliverable D.6.5.2 in July 2005. Minerva has been demonstrated at two major conferences, VLDB 2005 in Trondheim and Middleware 2005 in Grenoble, and received great attention and very positive feedback [10, 11].

References


[4] Holger Bast, Debapriyo Majumdar. Understanding Spectral Retrieval via the Synonymy Graph,

Holger Bast, Debapriyo Majumdar, Ralf Schenkel, Martin Theobald, Gerhard Weikum: IO-Top-k: Index-access Optimized Top-k Query Processing, submitted for publication.


Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, Christian Zimmer: P2P Content Search: Give the Web Back to the People, submitted for publication.

Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum: IQN Routing: Integrating Quality and Novelty in P2P Querying and Ranking, accepted for International Conference on Extending Database Technology (EDBT), Munich, Germany, 2006.


Klaus Berberich, Michalis Vazirgiannis, Gerhard Weikum: Time-aware Authority Ranking, accepted for publication in the Internet Mathematics Journal.


[34] Ulrich Giintzer, Wolf-Tilo Balke, Werner Kiefling: Optimizing Multi-Feature Queries for Image Databases, International Conference on Very Large Data Bases (VLDB), Cairo, Egypt, 2000.


[61] Martin Theobald, Gerhard Weikum, Ralf Schenkel: Top-k Query Evaluation with Probabilistic Guarantees, International Conference on Very Large Data Bases (VLDB), Toronto, Canada, 2004


