Towards Self-Organizing Query Routing and Processing for Peer-to-Peer Web Search

Gerhard Weikum, Holger Bast, Geoffrey Canright, David Hales,
Christian Schindelhauer, Peter Triantafillou *

Contact author: Gerhard Weikum, Max-Planck Institute for Computer Science,
Stuhlsatzenhausweg 85, Saarbruecken, Germany, Email: weikum@mpi-sb.mpg.de

Keywords:

peer-to-peer systems, probabilistic and statistical methods, semantic overlay networks, self-organization, Web search.

Abstract

The peer-to-peer computing paradigm is an intriguing alternative to Google-style search engines for querying and ranking Web content. In a network with many thousands or millions of peers the storage and access load requirements per peer are much lighter than for a centralized Google-like server farm; thus more powerful techniques from information retrieval, statistical learning, computational linguistics, and ontological reasoning can be employed on each peer's local search engine for boosting the quality of search results. In addition, peers can dynamically collaborate on advanced and particularly difficult queries. Moroever, a peer-to-peer setting is ideally suited to capture local user behavior, like query logs and click streams, and disseminate and aggregate this information in the network, at the discretion of the corresponding user, in order to incorporate richer cognitive models.

This paper gives an overview of ongoing work in the EU Integrated Project DELIS that aims to develop foundations for a peer-to-peer search engine with Google-or-better scale, functionality, and quality, which will operate in a completely decentralized and self-organizing manner. The paper presents the architecture of such a system and the Minerva prototype testbed, and it discusses various core pieces of the approach: efficient execution of top-k ranking queries, strategies for query routing when a search request needs to be forwarded to other peers, maintaining a self-organizing semantic overlay network, and exploiting and coping with user and community behavior.

* The authors are with the Max-Planck Institute for Computer Science in Saarbruecken, Telenor in Oslo, the University of Bologna, the Heinz-Nixdorf Institute in Paderborn, and the University of Patras. The work presented in this paper is partially supported by the EU within the 6th Framework Programme under contract 001907 "Dynamically Evolving, Large Scale Information Systems" (DELIS).

1 Motivation

The age of information explosion poses tremendous challenges regarding the intelligent organization of data and the effective search of relevant information in business and industry (e.g., market analyses, logistic chains), society (e.g., health care), and virtually all sciences that are more and more data-driven (e.g., gene expression data analyses and other areas of bioinformatics). The problems arise in intranets of large organizations, in federations of digital libraries and other information sources, and in the most humongous and amorphous of all data collections, the World Wide Web and its underlying numerous databases that reside behind portal pages. The Web bears the potential of being the world's largest encyclopedia and knowledge base, that would be of great value to all kinds of "knowledge workers" from students to Nobel laureates, but we are very far from being able to exploit this potential.

Search-engine technologies provide support for organizing and querying information; for simple mass-user queries that aim to find popular Web pages on pop stars, soccer clubs, or the latest Hollywood movies, existing engines like Google are probably the best solution. But for advanced information demands search engines all too often require excessive manual preprocessing, such as manually classifying documents into a taxonomy for a good Web portal, or manual postprocessing such as browsing through large result lists with too many irrelevant items or surfing in the vicinity of promising but not truly satisfactory approximate matches. The following are a few example queries where current Web and intranet search engines fall short:

Q1: Which professors from Saarbruecken in Germany teach information retrieval and participate in EU projects?
Q2: What are the most important research results on percolation theory?
Q3: Which drama has a scene in which a woman makes a prophecy to a Scottish nobleman that he will become king?

Why are these queries difficult (too difficult for Google-style keyword search unless one invests a huge amount of time to manually explore large result lists with mostly irrelevant and some mediocre matches)? For Q1 no single Web site is a good match; rather one has to look at several pages together within some bounded context: the homepage of a professor with his address, a page with course information linked to by the homepage, and a research project page that is a few hyperlinks away from the homepage. Q2 is not a query in the traditional sense, but requires gathering a substantial number of key resources with valuable information on the given topic; it would be best served by looking up a well maintained Yahoo-style topic directory, but highly specific expert topics are not covered there. Q3 cannot be easily answered because a good match does not necessarily contain the keywords "woman", "prophecy", "nobleman", etc., but may rather say something like "Third witch: All hail, Macbeth, thou shalt be king hereafter!" and the same document may contain the text "All hail, Macbeth! hail to thee, thane of Glamis!". So this query requires some background knowledge to recognize that a witch is usually female in the literature, "shalt be" refers to a prophecy, and "thane" is a title for a Scottish nobleman.

One of the major goals of the *EU Integrated Project DELIS*[⋆] is to develop foundations for collaborative Web information search in an Internet-scale *peer-to-peer (P2P) system* [33]. This approach bears the potential of overcoming the shortcomings of today's Google-style search engine technology and successfully tackling the above kinds of killer queries. In the DELIS approach every peer, e.g., the home PC of a scientist or student, has a full-fledged search engine that indexes a small portion of the Web, according to the interest profile of the user. Such an architecture has four major advantages over a centralized server farm like Google:

1. As the data volume and the query load per peer are much lighter, the peer's search engine can employ much more advanced techniques for concept-based rather than keyword-based search, leveraging background knowledge in the form of thesauri and ontologies [23, 57, 62] and powerful mathematical and linguistic techniques such as spectral analysis and named entity recognition [4, 5, 13, 17, 30, 34, 64].
2. Peers can collaborate for finding better answers to difficult queries: if one peer does not have a good result locally it can contact a small number of judiciously chosen peers who are considered "knowledgeable" on the query topic [6, 38, 48]. This approach should often be able to exploit the small-world phenomenon on the Web: knowledgable peers are only a short distance away.
3. A P2P system can gather and analyze bookmarks, query histories, user click streams, and other data about user and community behavior; the implicit and explicit assessments and recommendations derived from this input can be leveraged for better search results [20, 39, 65]. In contrast to a central server, the P2P approach provides each user with direct, fine-grained control over which aspects of her behavior may be collected and forwarded to other peers.
4. A politically important issue is that a P2P search engine is less susceptible to manipulation, censorship, and the bias induced by purchased advertisements.

This paper addresses some of the foundational issues that need to be solved in order to achieve the elusive goal of intelligent, efficient, and self-organizing P2P Web search. It gives an overview of various research contributions made by the DELIS project: models, algorithms, strategies, and experimental analyses, from a complex systems viewpoint. The paper is organized as follows. Section 2 presents a reference architecture, discusses its complex systems aspects, and outlines a prototype implementation coined *Minerva*. Section 3 presents new methods for efficiently processing top-k queries, that is, search requests that produce ranked result lists, in descending order of statistically estimated relevance, and may stop early after the top-k result items. Section 4 discusses various approaches to the problem of query routing, that is, the decision about selecting a small number of peers to which a user's query should be forwarded for obtaining richer search results with good benefit/cost ratio. Section 5 deals with the diversity of peer behaviors: how to exploit user profiling information for better search results and how to handle altruistic vs. selfish peers. We conclude with an outlook on future challenges.

[⋆] Dynamically Evolving Large-Scale Information Systems

2 P2P Web Search as a Complex System

The DELIS project is 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 contents 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 "statistically 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 much 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 [44], 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.

A system architecture for the envisioned solution is depicted in Figure 1. This architecture is currently prototyped, as an experimental platform within the DELIS project, under the name *Minerva* [6, 7].** Throughout this paper, we will often to refer to the Minerva system whenever it is necessary to refer to a concrete architecture.

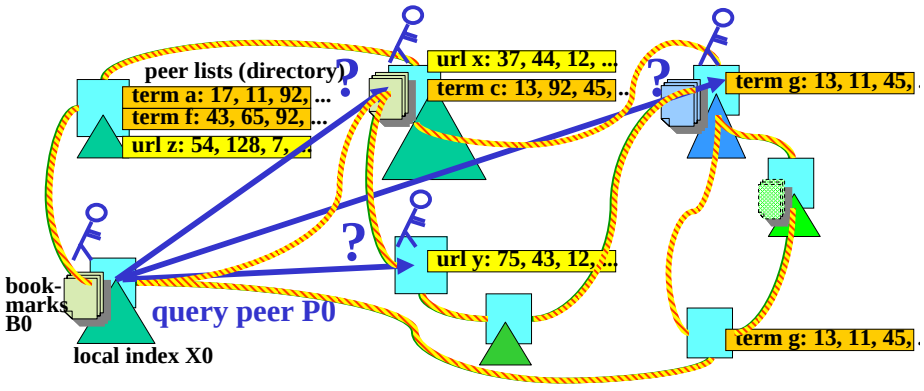

Fig. 1. System Architecture for P2P Web Search

**Minerva is the Roman goddess of science, wisdom, and learning, and is also the icon of the Max Planck Society.

In this setting, each peer runs a full-fledged, possibly personalized, search engine, fully equipped with a crawler, index manager, and a query processor to search the local index and information content. Peers are connected by an overlay network, for example, a distributed hash table (DHT); in Minerva a Chord-based DHT is used [59]. Such overlay networks and the resulting neighborhood structures ensure [37, 53, 59]

1. short routing paths (by a very small graph diameter and appropriate routing tables), usually O(log n) where n is the number of peers, and
2. low memory overhead per peer, usually by bounding the size of the routing tables,
3. unlimited scalability, so that the aggregated performance capacity increases linearly with the number of peers in the network,
4. self-organization in the presence of churn, i.e., the high dynamics of peers that join and leave the network without notice and at possibly very high rates, and
5. self-healing capabilities in the presence of peer failures.

Peers analyze their local information content and prepare compact statistical synopses that capture the relevance for specific query terms (i.e., keywords, stemmed words, or concepts onto which words are mapped), the richness, authority, and freshness of the content, the behavioral characteristics of the peer including the corresponding user's thematic interests, the peer's quality-of-service properties, etc. These synopses are posted into the overlay network: disseminated to specifically chosen (e.g., by the DHT hash function) peers, often in a redundant manner with judiciously chosen replicas on different peers, such that the overall network forms a conceptually integrated but physically massively distributed directory for metadata and statistical summaries. In the Minerva architecture shown in Figure 1, each peer posts the terms that are statistically most characteristic for its local content and the URLs of its bookmarks that reflect the user's interests. The peer or the peers that are responsible for maintaining the directory entry for a given term or URL maintain ranked lists of peers that have good information about the term or have bookmarked the URL, respectively. The decentralized directory can be efficiently queried by all peers.

A user query is normally executed on the local index first, thus avoiding network costs unless involving other peers is justified by unsatisfactory local results. In the latter case, a query routing decision is made about which other peers should be contacted in order to evaluate the query. Clearly this is the technically interesting case, which would typically arise with advanced information demands of "power users" like scientists, students, or journalists. For dynamically selecting target peers, the originating peer can inquire the directory and base its decision on the statistical summaries that have previously been posted. The originator may additionally use further information about the candidate peers' content, trustworthiness, and behavior that it has locally cached from previous interactions and observations. Once the target peers for executing the query are determined, the query is processed using a top-k algorithm, either by a) running the complete query on each selected peer and merging the search results, or by b) decomposing the query into individual subqueries, like one subquery per term, and using a network-conscious distributed top-k algorithm, where the extra difficulty is to reconcile the network costs (i.e., bandwidth consumption and latency) and the processing costs of the involved peers (i.e., CPU time, disk accesses, memory consumption).

We see that this system architecture has all the characteristics and poses the challenges of a complex system [2, 3]. Its individual components, the local search engines or the DHT in isolation, are reasonably well understood, but the interplay of all components within a network of many thousands or millions of peers creates extra complexity that we cannot yet master. The autonomy of peers and the diversity of different behavioral patterns can be understood only by analyzing and controlling the system at different levels, ranging from the underlying physical network and the virtual overlay network layers to the level of intelligent search, query routing, and collaboration strategies of the individual peers. For cost-efficient solutions it is crucial to consider benefit and cost factors at all these levels. Finally, a deep understanding mandates studying such complex systems at different scales in terms of time and space, for example, the short-term interactions of a peer with its immediate neighborhood, triggered by query routing and query execution, on one hand, and the long-term, long-range evolution of the entire system, to organize itself into effective and robust semantic overlay structures, on the other hand.

3 Efficient Top-k Query Processing

3.1 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 [40, 12]. 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 [21, 26]. 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 situtation, 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 [21]. 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. TA-sorted has been shown to be theoretically instance-optimal when the number of index lists is a (small) constant [21, 26, 47]. Unfortunately, the performance of TA-sorted tends to degrade 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.

3.2 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:

$$lowerb(d) = \sum_{j \in E(d)} s_j(d) \leq s(d) \leq \sum_{j \in E(d)} s_j(d) + \sum_{j \notin E(d)} high_j = upperb(d)$$

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, $lowerb(d)$ and $upperb(d)$ are the lower and upper bounds of d's score, and $high_j$ 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 $upperb(d) > lowerb(rank\text{-}k)$ where $rank\text{-}k$ 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 $high_j$ 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 $lowerb(rank\text{-}k)$. Then d is discarded from the candidate list if

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

with some pruning threshold δ. 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.

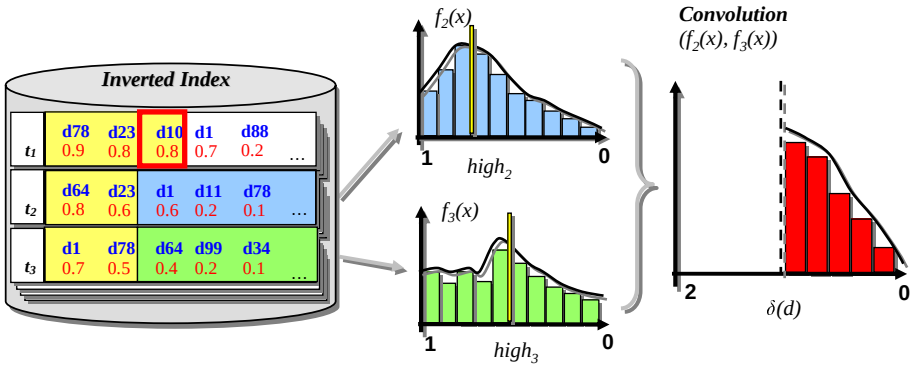

Fig. 2. Probabilistic score predictor for early candidate pruning

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 various benchmark and other test corpora (see, e.g., http://trec.nist.gov) 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 [61, 62].

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")*, the Prob-sorted method can add index lists dynamically and incrementally, rather than having to expand the query upfront based on thresholds. To this end, the algorithm considers a statistically computed 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]. The Prob-sorted algorithm has been implemented in the Minerva testbed for P2P Web search that is being developed within DELIS [6, 7].

3.3 Distributed Top-k Algorithms for P2P Systems

Minerva can run distributed versions of both TA-sorted and the Prob-sorted algorithm. A query is 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.

A new family of algorithms, coined KLEE [43], 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 [11], 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 describe 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 [11] 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 resuls 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.

4 Self-Organizing Query Routing

4.1 State of the Art

Query routing is a core piece of a P2P search engine. It aims to select the most promising peers among a large set of candidates, for executing a given query posed by one of the peers. 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 prior work on the formation of "semantic overlay networks" (e.g., [1, 19, 16]) disregard 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 [44] has addressed the ranking of data sources and also the reconciliation of search results from different sources. relevant, too. GlOSS [24] and CORI [10] 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 [38, 48] 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 copy with high dynamics.

4.2 Dynamic Peer Selection

The rationale for the query routing strategy developed in DELIS is based on the following three observations [6, 7]:

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. To limit the overhead of estimating these measures, we use the Kullback-Leibler divergence [18, 63] between the bookmark documents of the two peers as a basis for estimating benefit. 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 [12, 56]. For overlap we consider a large sample of a peer's locally indexed documents, and we represent the set of these documents using a compact Bloom filter for information exchange among peers [7]. Then benefit is defined as the quotient of similarity and overlap. An additional aspect of benefit that we are currently exploring is to incorporate also measures of reputation (authority) and trust, both of which could be implemented using PageRank-style Eigenvector computations [9, 25, 32, 35]. The difficulty is to compute such a measures in a fully decentralized way, without having to build a global graph structure, capturing links, recommendations, or opinions, on a central site. Promising techniques along these lines have been developed in [49].

We reconcile the notion of benefit with the third of the above observations by considering the *benefit/cost ratio* of peers, where cost is estimated based on tracking the utilization and resulting response time of different peers.

The outlined strategy as well as various alternative strategies from the prior literature have been implemented in the *Minerva testbed* [6, 7]. Experimental studies on benchmark datasets show that the estimation of overlap makes query routing very cost-efficient, reducing the number of peers that need to be contacted while at the same time ensuring very high recall and precision.

4.3 Semantic Overlay Networks

The query routing decisions are made at the run-time of queries, so the overhead for statistical similarity and overlap comparisons may adversely affect the user-perceived query response times. To ensure that routing decisions have acceptably low overhead, various kinds of precomputations can be employed and leveraged. In essence they all

lead to some form of "semantic overlay network" in addition or on top of the DHT-based overlay network that connects all peers. In the semantic overlay network, SON for short, only peers are directly connected as neighbors that have high likelihood of being beneficial to each other's queries. In principle, one could think of the SON as the result of running a content-similarity-based clustering algorithm on the peers, but this approach would be static and would not self-adjust the resulting neighborhood structure to the dynamics of a P2P system. In DELIS we rather pursue three other approaches, which are better geared for a decentralized large-scale setting with high dynamics:

- benefit-based ranking of postings in the DHT-based directory,
- opportunistic caching at every peer,
- proactive dissemination based one epidemic spreading methods, and
- dynamically self-organizing SONs based on random graphs with benefit-driven bias.

The *directory-based approach* [59, 6] is closest to the architecture outline before. In addition to posting terms, term-frequency and other statistical measures, and bookmarked or otherwise important URLs to the directory, the peer that is responsible for some of a term or URL simply computes a global ranking of the best peers for the term or URL. At query-routing time, when the directory peer is inquired by the query originator, only a small number of best candidate peers is returned, thus ensuring low overhead. The directory should be continuously updated, at an acceptable background-activity rate, to avoid degradation by stale statistics.

In the *caching approach* [50] every peer maintains a history of its interactions with other peers. This includes statistical similarity, overlap, and execution-cost information inquired for query routing, actual query results for peers that were involved in query execution, additional content statistics that such peers may have piggybacked on their replies, and also information about queries posed by other peers when the given peer conversely processes remotely issued queries. All this information would simply be collected in a local cache, with appropriate invalidation, update, and replacement strategies when cached entries become stale or the cache becomes full. This enables a peer to build up its own, possibly personalized or otherwise biased, statistical view of interesting other peers. On this basis, a peer could effectively determine a short list of most interesting peers and consider them as virtual neighbors in an addition overlay structure. Note that piggybacking, adding a small amount of extra data to the payload of a network message that needs to be sent anyway, is a well established technique in distributed computing and particularly intriguing in the P2P setting. It may even include peers that are involved in lower-level network routing, e.g., the small number of peers that lie on the IP-layer path from a sending to a receiving peer in the DHT overlay. This way, peers may learn about frequent queries by other peers even if they never become involved in the actual query execution. Obviously, these techniques should be controlled by the degree to which peers are willing to share such information.

In addition to merely caching useful information when the opportunity arises, we can employ a *proactive dissemination* scheme. Typically, this will be based on the epidemic spreading paradigm [8, 31]. A peer chooses a small number of random peers and sends them a compact synopsis of statistical information about itself, and possibly also about its subjective view of the global-network statistics. Such a dissemination step is

transitively repeated. The rate of these background messages can be controlled so as to limit its network bandwidth consumption. Note that this approach resembles the request forwarding scheme in unstructured Gnutella-style P2P networks, which is widely considered inefficient in terms of networking costs. A major difference in the DELIS approach, however, is that we advocate such techniques only for statistical summaries and other metadata, as opposed to actual queries, query results, or primary data. Thus, we can much more easily throttle the message rate for this kind of network traffic and limit the overhead to an acceptable level, whereas Gnutella-style approaches rely on message flooding as its only mechanism for on-demand request processing.

Finally, the last approach that we investigate, and currently favor, for the Minerva system is to dynamically maintain a *SON based on a random but biased graph*. The maintenance of this graph can actually exploit any of the above approaches for propagating statistical summaries in the network. We assume that a peer maintains a list of other interesting peers, its current *"friends"* (in addition to the DHT-provided connectivity). For simplicity, we assume that this is a fixed-size list (alternatively, a variable but bounded number of friends may be maintained). The friends should always be those peers, known to the given peer, that are predicted to provide the highest benefit to the given peer's (future) queries. As mentioned before, benefit is measured as a quotient of estimated thematic similarity and estimated overlap.*** When the given peer interacts with other peers and learns about their potential benefit, it can consider replacing one of its current friends by the newly met peers. In addition, the peer may consider "blind dates" with randomly chosen peers, to obtain a broader perspective and avoid getting stuck in a local or thematically narrow neighborhood. When a friend is replaced, this amounts to re-wiring the graph on which the SON is based. The whole approach is fully self-organizing as it does not require any centralized coordination or explicit control by humans, and every peer can use its own strategy and bias for meeting other peers and considering them as new friends.

The DELIS project investigates various specific methods along the above lines, and aims to compare the current DHT-based implementation of Minerva with alternative P2P networks. One of the novel approaches is called *Peanuts* [42] and combines a random-graph-based, robust backbone structure with search trees.

The backbone network is simple and very reliable under churn. A random network has several advantages compared to deterministic structured networks. Above all, the costs for maintaining the network are low since there is no predetermined neighborhood. In [41] we introduce a simple scheme to maintain such random networks. The main component of this scheme is a periodically performed local link exchange procedure. This so-called 1-Flipper operation continuously and randomly permutes the d-regular undirected network. Unless nodes disappear from the network this so-called 1-Flipper operation guarantees connectivity and upholds an infinite series of truly random networks chosen uniformly from the set of all d-regular connected graphs. Such a random graph guarantees the expansion property asymptotically almost surely, i.e. with probability converging to 1. Networks with high expansion have small (logarithmic) diameter and high connectivity.

*** It is straightforward to incorporate also further measures like estimated execution cost, reputation, trust, etc.

Upon the backbone we superimpose a search tree, supporting range queries, neighborhood search, and prefix lookups. We aim to keep the maintenance of the network structure as local as possible, relying on periodic handshakes as non-local operations. The search keys can capture semantic information like topics in a topic directory. Therefore the trees may be highly unbalanced, and load balancing is necessary. For the problem of weight balancing, [55] introduced weighted consistent hashing. Peers are recursively assigned to branches of the search tree according to the load information available. When the load changes because new data arrives, old data is deleted, tree branches are addedm or branches are pruned, each peer can determine whether it needs to be rebalanced. One of the many benefits of this approach is that if the original load is restored then all peers return to their previous positions within the search tree. Thus, weighted consistent hashing nicely supports local caching of distributed data.

Another interesting task is to detect social and unsocial behavior in P2P networks. This is inspired by "social tagging", mimicking evolution in social and biological networks [28]. Here the metaphor is that a peer that meets another "good" peer should be interested in adopting some of the good peers' friends. In the implementation, this behavior amounts to re-wiring the SON by replacing peers in a peer's friends list by friends of a beneficial peer, with some degree of randomization.

These methods are currently being implemented in the Minerva testbed for future experimentation with self-organizing SONs, as a boosting technique for highly efficient query routing.

5 Coping with User and Community Behavior

5.1 Exploiting Query-Log and Click-Stream Information

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., [20, 65]), 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 recent work [39] 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.

5.2 Tracking Egoistic versus Altruistic Peers

A Real-world Perspective. Our starting position is that our world consists of altruists, selfish people, and of others with behavior ranging in between, with a non-negligible percentage of the last category showing altruistic behavior if given the incentives to do so. Within the world of P2P networks this fact has been clearly manifested and documented: take as example the "free riders" phenomenon [54], first measured in the Gnutella network.

The bad news is that the great majority of Gnutella peers were proven to be free riders (more than 70%). And this is indeed very bad news for DHT-style overlays, since the great majority of peers may be joining the network and leaving very soon thereafter. The good news are that a non-negligible percentage of the peers were proven to be altruistic. In Mojonation more than 1-2% of all users stayed connected almost all the time. In Gnutella, 1% (10%) of peers served about 40% (90%) of the total requests [54], and the longer a node has been up, the more likely it is to remain up. Note that the flooding-based routing algorithm in Gnutella-like networks acts as a counter-incentive to acting altruistically, by swamping good peers with requests. We conjecture that, by giving incentives and taking away such counter-incentives, more network nodes will be willing to act altruistically.

A Research Perspective. In DHT-structured P2P networks with extremely high dynamics (i.e., a large fraction of peers joining and leaving at very high rates), routing performance is susceptible to degradation. [36, 51] have studied how to still guarantee in highly-dynamic cases $O(\log N)$ routing performance by increasing redundancy and running more sophisticated forms of stabilization "rounds". However, these approaches come with considerable overhead, and they largely ignore an optimization potential that lies in the widely varying reliability and performance capacity of the peers. On the other hand, when considering the unstructured P2P research efforts, one also notices a lack of considerable attention on research exploiting the heterogeneities among peer nodes [54].

But heterogeneity means more than a mere distinction between powerful and weak nodes; there is also heterogeneity with respect to their behavior, being altruistic or selfish. For example, there will be powerful nodes that will not be acting altruistically. It is reasonable to expect that altruistic nodes will tend to have greater (processing, memory,

and bandwidth) capabilities, willing to share them (when not in use) with others (practically at very small extra costs, given the flat-rate resource pricing) . This expectation has been validated in [66].

Our Approach. The aforementioned related work has led to a more critical attitude to DHT-based structured overlay networks and a preference towards unstructured Gnutella-style overlays [14]. Conversely, we follow a different path; we add further structure to DHTs, leveraging altruistic peers. In this way, we can deliver performance and robustness guarantees for the steady-state case and, more importantly, also for the high-churn cases. Our paradigm is based on monitoring and identifying behavioral patterns of various peers. At the base of this approach there is an accounting and auditing layer, coined *SeAl*, that identifies and validates selfish versus altruistic peers [45], giving peers incentives to behave altruistically, while guarding against misbehaving peers. This infrastructure can in turn serve as a basis upon which a number of key services can be provided over complex systems, most importantly enhanced routing methods that combine short latency and high throughput with robustness regarding churn [46].

5.3 Emerging Incentives

Recent work on P2P systems has indicated the potential value of incentive mechanisms [15, 22], particularly link-based incentive approaches for promoting cooperation and reducing the impact of malicious or selfish nodes [60].

Essentially, nodes in the network dynamically change their neighbor lists - the nodes they link to directly - based on some evaluation of the performance they receive from those neighbors. The idea is that bad nodes get pushed to the periphery of the network as links are broken and hence have only a small impact on the functioning of the network as a whole. However, this kind of approach imposes some node-level overheads, requires the design of an appropriate incentive structure for each application domain, and requires interactions to occur over an extended period with the same neighbors.

More recently, new work within the DELIS project has applied ideas derived from Computational Social Science [27, 52, 58] in order to circumvent some of these problems by producing light-weight algorithms that dynamically emerge incentive structures through a form of evolution in the network. This novel approach, called the SLAC algorithm (Selfish Link-based Algorithm for Cooperation) [28], is simple yet powerful. Nodes move in the network by copying the neighbor lists and behaviors of nodes that are obtaining better levels of performance, the idea being that nodes want to move in the network to locations that support better levels of performance than their current position.

Experiments with this approach show that initially random or disconnected networks can be made to quickly self-organize into clusters or “tribes” that have desirable properties. Firstly, selfish and malicious nodes are quickly isolated because they destabilize the tribes they inhabit; secondly, tribes that have complementary skills or capacities can be made to emerge.

The SLAC algorithm appears particularly suitable for P2P applications because it is light-weight (making few computational or communication demands), it is scalable

(in fact the larger the population the better it works), and it is robust and decentralized (requiring no central control or servers). In the context of a P2P search engine, the desirable properties of SLAC can be applied to reduce the effect of malicious and selfish nodes and to structure the dynamics of a semantic overlay network.

6 Conclusion

Aiming for a post-Google Web search engine based on the P2P paradigm is a very ambitious goal. How can complex systems research contribute to this endeavor that the DELIS project has embarked on? Collaboration among peers requires strategies for routing queries to other peers and for exchanging metadata, statistics, and background knowledge to form an evolving "semantic overlay network". Understanding the dynamics and behavior of such a network requires analyses at different levels and scales of the overall network, and thus falls right in the core of complex systems methodology. To be practically viable, a P2P approach needs good incentive mechanisms to limit the influence of egoistic or malicious peers. Successfully addressing this difficult issue requires combining expertise and methods from multiple scientific fields like game theory, sociology and evolutionary biology, statistical dynamics, and algorithmics, in order to master both the descriptional and the computational complexity. Finally, constructing and maintaining the rich statistical, ontological, and cognitive models that should drive the information dissemination and query processing strategies of a P2P search engine requires a deep and unified understanding of the interrelationships and dynamic interactions among these different building blocks.

The DELIS project is pursuing these challenging research avenues for laying the foundations of next-generation information search spread across a self-organizing network of millions of peers.

References

1. Karl Aberer, Philippe Cudr-Mauroux, Manfred Hauswirth, Tim Van Pelt: GridVine: Building Internet-Scale Semantic Overlay Networks. International Semantic Web Conference (ISWC), Hiroshima, Japan, 2004.
2. Albert-Lszl Barabsi Linked: The New Science of Networks, Persus Publishing, 2002
3. Mark Buchanan Nexus: Small Worlds and the Groundbreaking Science of Networks, Norton Publishing, 2002
4. Holger Bast, Debapriyo Majumdar: Understanding Spectral Retrieval via the Synonymy Graph, ACM SIGIR International Conference on Research and Development in Information Retrieval, Salvador, Brazil, 2005
5. Holger Bast and Ingmar Weber: Insights from Viewing Ranked Retrieval as Rank Aggregation, International Workshop on Challenges in Web Information Retrieval and Integration (WIRI), Tokyo, Japan, 2005
6. Matthias Bender, Sebastian Michel, Gerhard Weikum, Christian Zimmer: Bookmark-driven Query Routing in Peer-to-Peer Web Search, ACM SIGIR Workshop on Peer-to-Peer Information Retrieval, Sheffield, UK, 2004
7. Matthias Bender, Sebastian Michel, Peter Triantafillou, Gerhard Weikum, Christian Zimmer: Improving Collection Selection with Overlap Awareness, ACM SIGIR International Conference on Research and Development in Information Retrieval, Salvador, Brazil, 2005

8. Kenneth P. Birman: The Surprising Power of Epidemic Communication. In: Future Directions in Distributed Computing, Lecture Notes in Computer Science 2584, Springer, 2003.
9. Sergey Brin, Lawrence Page: The Anatomy of a Large-Scale Hypertextual Web Search Engine, International World Wide Web Conference (WWW), Brisbane, Australia, 1998
10. James P. Callan, Zhihong Lu, W. Bruce Croft: Searching Distributed Collections with Inference Networks. ACM SIGIR International Conference on Research and Development in Information Retrieval, Seattle, USA, 1995
11. Pei Cao, Zhe Wang: Efficient Top-K Query Calculation in Distributed Networks, ACM International Symposium on Principles of Distributed Computing (PODC), St. John's, Canada, 2004
12. Soumen Chakrabarti: Mining the Web: Discovering Knowledge from Hypertext Data, Morgan Kaufmann Publishers, 2002
13. Soumen Chakrabarti: Breaking Through the Syntax Barrier: Searching with Entities and Relations. European Conference on Machine Learning (ECML), Pisa, Italy, 2004
14. Yatin Chawathe, Sylvia Ratnasamy, Lee Breslau, Nick Lanham, Scott Shenker: Making gnutella-like P2P systems scalable. ACM SIGCOMM International Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication, Karlsruhe, Germany, 2003
15. Bram Cohen: Incentives Build Robustness in BitTorrent, Workshop on the Economics of Peer-to-Peer Systems, Berkeley, USA, 2003
16. Edith Cohen, Amos Fiat, Haim Kaplan: Associative Search in Peer to Peer Networks: Harnessing Latent Semantics. Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM), San Francisco, 2003
17. William W. Cohen, Sunita Sarawagi: Exploiting dictionaries in named entity extraction: combining semi-Markov extraction processes and data integration methods. ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Seattle, USA, 2004.
18. Thomas Cover, Joy Thomas: Elements of Information Theory, Wiley & Sons, 1991.
19. Arturo Crespo, Hector Garcia-Molina, Semantic Overlay Networks for P2P Systems, Technical Report 2003-75, Stanford University, 2003
20. Hang Cui, Ji-Rong Wen, Jian-Yun Nie, Wei-Ying Ma: Query Expansion by Mining User Logs. IEEE Transactions on Knowledge and Data Engineering 15(4): 829-839, 2003
21. Ronald Fagin, Amnon Lotem, Moni Naor: Optimal Aggregation Algorithms for Middleware, Journal of Computer and System Sciences 66(4): 614-656, 2003
22. Joan Feigenbaum, Scott Shenker: Distributed Algorithmic Mechanism Design: Recent Results and Future Directions, International Workshop on Discrete Algorithms and Methods for Mobile Computing and Communications, Atlanta, USA, 2002.
23. Christiane Fellbaum (Editor): WordNet: An Electronic Lexical Database, MIT Press, 1998
24. Luis Gravano, Hector Garcia-Molina, Anthony Tomasic: GlOSS: Text-Source Discovery over the Internet. ACM Transactions on Database Systems 24(2): 229-264, 1999
25. Zoltan Gyngyi, Hector Garcia-Molina, Jan Pedersen: Combating Web Spam with TrustRank. International Conference on Very Large Data Bases (VLDB), Toronto, Canada, 2004
26. Ulrich Güntzer, Wolf-Tilo Balke, Werner Kießling: Optimizing Multi-Feature Queries for Image Databases, International Conference on Very Large Data Bases (VLDB), Cairo, Egypt, 2000
27. David Hales: Cooperation without Space or Memory: Tags, Groups and the Prisoner's Dilemma. International Workshop on Multi-Agent-Based Simulation (MABS), Lecture Notes in Artificial Intelligence 1979, Springer, 2000.
28. David Hales. From Selfish Nodes to Cooperative Networks Emergent Link-based Incentives in Peer-to-Peer Networks. IEEE International Conference on Peer-to-Peer Computing, Zurich, Switzerland, 2004

29. Thomas Hofmann: Unsupervised Learning by Probabilistic Latent Semantic Analysis. Machine Learning 42(1/2): 177-196, 2001
30. Thomas Hofmann: Latent semantic models for collaborative filtering. ACM Transactions on Information Systems 22(1): 89-115, 2004
31. Richard M. Karp, Christian Schindelhauer, Scott Shenker, Berthold Vcking: Randomized Rumor Spreading. IEEE Symposium on Foundations of Computer Science (FOCS), Redondo Beach, USA 2000
32. Jon M. Kleinberg: Authoritative Sources in a Hyperlinked Environment. Journal of the ACM 46(5): 604-632, 1999
33. John Kubiatowicz: Extracting guarantees from chaos. Communications of the ACM 46(2): 33-38, 2003
34. Amy N. Langville and Carl D. Meyer. A Survey of Eigenvector Methods of Web Information Retrieval. The SIAM Review, 47(1):135-161, 2005
35. Amy N. Langville and Carl D. Meyer Deeper Inside PageRank, Internet Mathematics 1(3): 335-380, 2005
36. David Liben-Nowell, Hari Balakrishnan, David R. Karger: Observations on the Dynamic Evolution of Peer-to-Peer Networks. International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, USA, 2002.
37. David Liben-Nowell, Hari Balakrishnan, David R. Karger: Analysis of the evolution of peer-to-peer systems. ACM International Symposium on Principles of Distributed Computing (PODS), Monterey, USA, 2002.
38. Jie Lu, James P. Callan: Content-based Retrieval in Hybrid Peer-to-peer Networks, International Conference on Information and Knowledge Management, New Orleans, USA, 2003
39. Julia Luxenburger, Gerhard Weikum: Query-log based Authority Analysis for Web Information Search, International Conference on Web Information System Engineering (WISE), Brisbane, Australia, 2004
40. Christopher D. Manning, Hinrich Schütze: Foundations of Statistical Natural Language Processing, MIT Press, 1999
41. Peter Mahlmann, Christian Schindelhauer: Peer-to-Peer Networks based on Random Transformations of Connected Regular Undirected Graphs, ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), Las Vegas, 2005
42. Peter Mahlmann, Christian Schindelhauer: Peanuts — A Top-Down Peer-to-Peer Network, submitted as short paper to ECCS 2005.
43. Sebastian Michel, Peter Triantafillou, Gerhard Weikum: KLEE: Internet-scale Distributed Top-k Query Algorithms, submitted for publication
44. Weiyi Meng, Clement T. Yu, King-Lup Liu: Building Efficient and Effective Metasearch Engines, ACM Computing Surveys 34(1): 48-89, 2002
45. Nikolai Ntarmos, and Peter Triantafillou. SeAl: Managing Accesses and Data in Peer-to-Peer Data Sharing Networks. IEEE International Conference in Peer-to-Peer Computing, Zurich, Switzerland, 2004
46. Nikolai Ntarmos, Peter Triantafillou. AESOP: Altruism-Endowed Self Organizing Peers. 2nd International Workshop on Databases, Information Systems and Peer-to-Peer Computing, Toronto, Canada, 2004
47. Surya Nepal, M. V. Ramakrishna: Query Processing Issues in Image (Multimedia) Databases, IEEE International Conferenced on Data Engineering (ICDE), Sydney, Australia, 1999
48. Henrik Nottelmann, Norbert Fuhr: Combining CORI and the Decision-Theoretic Approach for Advanced Resource Selection, European Conference on Information Retrieval (ECIR), Sunderland, UK, 2004
49. Josiane Xavier Parreira, Gerhard Weikum: JXP: Global Authority Scores in a P2P Network, International Workshop on Web and Databases, Baltimore, USA, 2005

50. Michael Rabinovich, Oliver Spatscheck: Web Caching and Replication, Addison-Wesley, 2001
51. Sean C. Rhea, Dennis Geels, Timothy Roscoe, John Kubiatowicz: Handling Churn in a DHT, USENIX Annual Technical Conference, Boston, USA, 2004
52. Rick Riolo, Michael D. Cohen, Robert Axelrod: Cooperation without Reciprocity, Nature 414: 441-443, 2001.
53. Antony I. T. Rowstron, Peter Druschel: Pastry: Scalable, Decentralized Object Location, and Routing for Large-Scale Peer-to-Peer Systems. Middleware: International Conference on Distributed Systems Platforms, Heidelberg, Germany, 2001
54. Stefan Saroiu, P. Krishna Gummadi, Steven D. Gribble: Measuring and analyzing the characteristics of Napster and Gnutella hosts. Multimedia Systems 9(2): 170-184, 2003
55. Christian Schindelhauer, Gunnar Schomaker: Weighted Consistent Hashing, ACM Symposium on Parallelism in Algorithms and Architectures (SPAA), Las Vegas, USA, 2005
56. Sergej Sizov, Martin Theobald, Stefan Siersdorfer, Gerhard Weikum, Jens Graupmann, Michael Biwer, Patrick Zimmer: The BINGO! System for Information Portal Generation and Expert Web Search. International Conference on Innovative Data Systems Research (CIDR), Asilomar, USA, 2003
57. Steffen Staab, Rudi Studer (Editors): Handbook on Ontologies, Springer 2004
58. Luc Steels: The Evolution of Communication Systems by Adaptive Agents, Adaptive Agents and Multi-Agent Systems, LNCS2636, Springer, 2002.
59. Ion Stoica, Robert Morris, David Liben-Nowell, David R. Karger, M. Frans Kaashoek, Frank Dabek, Hari Balakrishnan: Chord: a scalable peer-to-peer lookup protocol for internet applications. IEEE/ACM Transactions on Networking 11(1): 17-32, 2003
60. Qixiang Sun, Hector Garcia-Molina: SLIC: A Selfish Link-based Incentive Mechanism for Unstructured Peer-to-Peer Networks. IEEE International Conference on Distributed Computing Systems (ICDCS), Tokyo, Japan, 2004.
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
62. Martin Theobald, Ralf Schenkel, Gerhard Weikum: Efficient and Self-Tuning Incremental Query Expansion for Top-k Query Processing, ACM SIGIR International Conference on Research and Development in Information Retrieval, Salvador, Brazil, 2005
63. Larry Wasserman, All of Statistics, Springer, 2004
64. Gerhard Weikum, Jens Graupmann, Ralf Schenkel, Martin Theobald: Towards a Statistically Semantic Web, International Conference on Conceptual Modeling, Shanghai, China, 2004
65. Ji-Rong Wen, Jian-Yun Nie, Hong-Jiang Zhang: Query Clustering Using User Logs, ACM Transactions on Information Systems 20(1): 59-81, 2002
66. Bryce Wilcox-O'Hearn: Experiences Deploying a Large-Scale Emergent Network. International Workshop on Peer-to-Peer Systems (IPTPS), Cambridge, USA, 2002.