Semantic Social Overlay Networks

Alexander Löser, Steffen Staab, and Christoph Tempich

***Abstract*— Peer selection for query routing is a core task in peer-to-peer networks. Unstructured peer-to-peer systems (like Gnutella) ignore this problem, leading to an abundance of network traffic. Structured peer-to-peer systems (like Chord) enforce a particular, global way of distributing data among the peers in order to solve this problem, but then encounter problems of network volatility and conflicts with the autonomy of the peer data management. In this paper, we propose a new mechanism, INGA, which is based on the observation that query routing in social networks is made possible by locally available knowledge about the expertise of neighbors and a semantics-based peer selection function. We validate INGA by simulation experiments with different data sets. We compare INGA with competing peer selection mechanisms on resulting parameters like recall, message gain or number of messages produced.**

***Index Terms*— Social networks, Communication system routing, Cooperative systems, Distributed database searching, Indexes, Information retrieval, Semantic networks**

I. Introduction

Finding relevant information from a heterogeneous set of information resources is a longstanding problem in computing. Studies of social networks suggest that the challenge of finding relevant information may be reduced to asking the 'right' people. 'The right people' either have the desired piece of information and can directly provide the relevant content or can recommend 'the right people'. Milgram's [4] and Kleinbergs [5] experiments illustrated that individuals with only local knowledge of the network (i.e. their immediate acquaintances) may successfully construct acquaintance chains of short length, leading to networks with 'small world' characteristics. In such a network, a query is forwarded along out going links taking it 'closer' to the destination. Such mechanisms work in social networks although

- people may not respond to requests,
- people may shift their interests and attention,
- people may not have exactly the 'right' knowledge, but knowledge which is *semantically close* to the request.

Rephrased in a peer-to-peer (P2P) terminology, peers may find relevant information in a real-world social network which is *highly dynamic* w.r.t. peer availability and expertise on topics using their *local knowledge* enriched by means of *semantic similarity* measures. In this context local knowledge refers to locally available information – structured and unstructured – and to knowledge about remote peers.

Manuscript received ; revised .
Alexander Löser is with University of Technology Berlin.
Christoph Tempich is with University of Karlsruhe.
Steffen Staab is with University of Koblenz Landau.
This paper is largely based on [1], [2], [3].
Digital Object Identifier 10.1109/JSAC.2006.070102.

A. Searching Dynamic Communities with Personal Indexes

Inspired by these observations this paper presents INGA a P2P routing algorithm for pure and unstructured P2P networks [3], [1]. Routing algorithms for pure and unstructured P2P networks do not require the active dissemination of index information describing a peer. They are therefore better suited for highly dynamic environments than routing algorithms based on structured or partially centralized network organizations [6]. INGA regards each peer in the network as a person in a social network; facts are stored and managed locally on each peer constituting the 'topical knowledge' of the peer. A peer responds to a query by providing an answer matching the query or by forwarding the query to relevant remote peers. The local peer determines the relevance of a remote peer based on a *personal semantic shortcut index*. The index is created and maintained in a lazy manner, i.e., by analyzing the queries initiated by the local peer and by analyzing the queries that are routed through the local peer. INGA creates shortcuts on four layers: The *content provider layer* contains shortcuts to remote peers which have successfully answered a query; the *recommender layer* stores information about remote peers who have issued a query; the *bootstrapping layer* maintains shortcuts to well connected remote peers; and the *network layer* connects to peers on an underlying default network.

This paper proposes a shortcut selection strategy able to identify and efficiently group peers with similar interests in a dynamic setting by semantic means. In order to adapt to the dynamics of the networks and to bound the local index this paper presents an index update policy combining temporal, semantic and community locality. The algorithm has been evaluated in simulations and real world case studies. This work presents the results of the simulation studies, while [7] introduces the case studies. Results show that INGA outperforms other state-of-the-art approaches significantly in terms of recall and messages per query.

The paper is organized as follows: Section II describes the infrastructure to maintain the index and the semantic similarity function to select peers. Section III elaborates on the index structure and update strategy for each type of shortcut. Section IV presents the dynamic routing model. Section V discusses the evaluation results and Section VII concludes the paper.

B. Related Work

Research in routing for P2P networks has devised an inventory of methods to collect index information about peers in the network. The first approaches for efficient indexing in P2P architectures were central indices, that have to transmit either meta data about the available content to central indexing peers, like e.g. GlOSS [8] or *Napster*. Current techniques for indexing P2P systems use structured overlays in the form of

distributed hash tables (DHTs), e.g. [9] or see [10] for a survey. Structured overlays allow for routing keyword queries to particular peers containing the desired data without the need for a central index. In structured overlays (a reference to) new content in the network is published at the node for the respective key, if new data on a peer arrives or a new peer joins the network. When a peer leaves the network the information about its content is unpublished. Recent research in [6] shows that due to the publishing/unpublishing overhead, DHTs lack efficiency when highly replicated items are requested, thus perform worse than flooding approaches in practical settings degrading further if network churn is introduced.

While the management of keys and objects in the same name space used in structured overlays provides an elegant and clean solution to routing within logarithmical bounds, it comes at significant costs destroying the locality of the content: Content at a user's desktop is co-located with other relevant items. Opportunities for browsing and pre-fetching are lost in structured overlays [11]. Routing approaches for pure unstructured networks, such as Gnutella, retain locality, but may produce many messages per query. In order to improve the routing efficiency in pure unstructured P2P networks current routing approaches use local index information (cf. [12] for an overview). They store information related to the success of past queries locally and utilize this information to route queries. The approaches may be distinguished, among other dimensions, according to their information gathering strategy, their information evaluation strategy and their index update strategy. The routing approach presented in [13] exploits interest-based locality and builds up indexing information about remote peers only at the querying peer. This information is not shared with remote peers, nor is it utilized for queries routed through the local peer. To update the index it employs a LRU strategy. The routing approaches of [14] and [15] return answers on the query path enabling remote peers on that path to profit from other peers queries. This information is also used to route queries from remote peers. Additionally, in [14] a peer disseminates obtained information to its neighbors and considers the number of responses of a remote peer for ranking purposes. In contrast, [15] uses a probability based evaluation function. Alternative routing algorithms like [16] introduce improvements on the evaluation of conjunctive queries or the dissemination procedure [17].

INGA differs from existing routing algorithms, because it uses queries routed through a peer to build up indexing information in contrast to responses, it uses semantic information to evaluate the index information for forwarding purposes and it uses semantic information to update the index.

II. System Architecture

The INGA peer selection strategies described in section III may be implemented for any unstructured P2P network. For evaluation purposes, though, we use the SWAP infrastructure [18]. It provides all standard peer-to-peer functionality such as information sharing, searching and publishing of resources.

A. Building Blocks

The building blocks of an INGA node responsible for the content and index information are the *local content database*, the *content management* component, the *local shortcut index* and the *shortcut management* component. The user interacts with the system by means of the *query/result interface*. The *routing logic* selects remote peers to send and forward queries to. Finally, the *network management* handles the physical network traffic assuming that each peer provides a unique peer identifer (PID). The components are described with more detail in the following.

Similar to file sharing networks each peer may publish all resources from its *local content database*, thus remote peers can discover them on request (this also applies to resources downloaded from other peers). All information is wrapped as RDF statements and stored in an RDF repository [19][1]. RDF is a light-weight ontology representation language offering, among others, constructs to type information ("isInstanceOf") and to relate information items ("subTopicOf"). In this format each resource representing the local data (*Nick isExecutiveEditorOf JSAC2005*) is assigned a topic *(JSAC isInstanceOf IEEEJournal)* and hierarchical information about the topic *(IEEEJournal subTopicOf Journal)*. The topics a peer stores resources for are subsequently referred to as the peer's own topics. INGA supports exact match queries which come in two flavours: single predicate queries and conjunctive queries. The *shortcut management* extracts information about answering and forwarding peers from queries to create, update or remove shortcuts in the *local shortcut index*. Contrary to related approaches, such as DHTs, INGA peers only index 'egoistically', i.e. they decide based on local information which shortcuts to store. The *routing logic* selects 'most suitable' peers to forward a query to, for all local queries or queries forwarded from remote peers. The selection depends on the knowledge a peer has already acquired and the similarity between the query and locally stored shortcuts.

B. Query and Result Messages

INGA uses a query message model which is similar to the structure of a Gnutella query message. Each query message is a quadruple: $QM(q, b, mp, qid)$ where q is a SERQL query[2]). INGA supports conjunctive SERQL queries. From a query INGA uses for routing purposes only information, which is shared among all peers. From a query for all *IEEEJournal* with editor-in-chief *Nick*, only *IEEEJournal* is utilized for routing. b is the bootstrapping capability of the querying peer to allow the creation of bootstrapping shortcuts. mp is the message path of a query message containing the unique PIDs of the peers, which have forwarded the query to the receiving peer. Finally, qid refers to a unique query ID to ensure that a peer does not respond to a query it has already answered. INGA computes unique query IDs using a random number generator that has sufficiently high probability of generating unique numbers. A result message is a tuple: $RM(r, mp, qid)$ where r represents the answer to the query. The message path mp is copied to the answer message to allow the creation of recommender and content provider shortcuts.

[1] `http://www.openrdf.org/`

[2] SERQL is a SQL like query language for RDF.

C. Similarity Function

INGA analyzes the locally stored shortcuts to determine remote peers to route a query to. If no shortcuts exist which exactly match a query, INGA evaluates the shortcuts based on their similarity to the query. Depending on the query type INGA chooses from the following similarity functions:

Single predicate query using a common topic hierarchy. If the peers in the network share a common topic hierarchy INGA exploits the semantic similarity between a query and a shortcut. In this case a query consists of a single predicate, which represents a topic in a common topic hierarchy, INGA utilizes the similarity function $sim_{Topic}(qt, st)$ presented in [20]. The similarity between the extracted topic qt from a SERQL query q and the extracted topic st from a shortcut, which are both given by query topics in the same topic hierarchy is calculated according to equation 1.

$$sim_{Topic}(qt, st) = \begin{cases} e^{-\alpha l} \cdot \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}} & \text{if } q \neq sc \\ 1 & \text{otherwise} \end{cases} \quad (1)$$

In equation (1), l is the length of the shortest path between qt and st in the graph spanned by the sub topic relation and h is the minimal level in the topic hierarchy of either qt or st. α and β are parameters scaling the contribution of shortest path length l and depth h, respectively. Based on the benchmark data set given in [20], we chose $\alpha = 0.2$ and $\beta = 0.6$.

Conjunctive queries. Each query may include several predicates, e.g. *Select all resources that belong to the topic semantic web and to the topic p2p*. Using a common topic hierarchy this query can be rewritten as *Find any resource having topics /computer/web/semanticweb and /computer/distributed/TourismTechnology*. An exact match approach routes a query only to a peer that matches *all* predicates of the query using a simple exact match paradigm. Too specific query predicates under the exact match paradigm often lead to empty result sets and do not appropriately consider negation. The notion of best matches and relative importance of predicates can be a good alternative to satisfy a user's information needs independently of the individual peer instances. In [2] we investigated metrics to determine the best peers to route a query using multi predicate queries in shortcut networks. We observed satisfying results using the selection function described in [16] which uses an equation similar to equation (2) to combine query hits for distributed document retrieval. We refer to this strategy as *Multiply*.

$$R_p(q) = \prod_{i=1}^{\#t} q_i^p \quad (2)$$

INGA calculates the relevance R for a peer p for a query q using equation (2), where $\#t$ represents the number of topics in the query, q_i^p represents the query hits per topic i of each peer matching at least one of the topics of the query. INGA selects the remote peers with the highest relevance.

III. Building and Maintenance of the Index

Each peer is connected to remote peers in the network via uni-directional shortcuts. A peer can link to any remote peer in the network if it knows its PID. Following the social metaphors outlined in section I, INGA distinguishes between the following shortcut types:

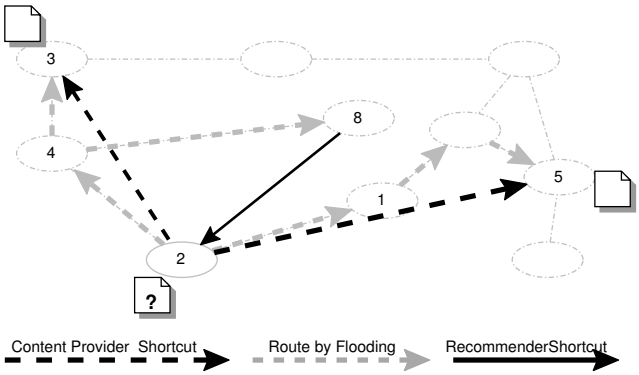

Fig. 1. Recommender shortcut creation

A. Content Provider and Recommender Shortcuts

Content Provider Layer. The design of the content provider shortcut overlay extends interest-based locality [13]. If the local peer receives answers from remote peers, these are potential candidates to be added to the content provider shortcut list. Each time the querying peer receives an answer from a remote peer, INGA creates content provider shortcuts *sc* to the responding remote peers in the form: *sc(topic, pid, query hits,'c', update)*, where *topic* is the query topics taken from the query message, *pid* is the unique identifier of the answering peer, *query hits* is the number of returned statements, *'c'* represents a type of shortcut, viz. 'content provider shortcut' and *update* is the time, when the shortcut was created or the last time, when the shortcut was used. Subsequent queries of the local peer or of a remote peer are matched against the topic column of the content provider shortcut list. If a peer cannot find suitable shortcuts in the list, INGA uses the bootstrapping or default network layer to select remote peers. In Figure 1 Peer 2 discovers shortcuts for the topic */Education/UML* by flooding the default network (TTL=3) and it creates two content provider shortcuts to peer 3 and peer 5.

Recommender Layer. The recommender layer maintains the recommender shortcuts. A recommender shortcut *sc(topic,pid,query hits, maxsim, 'r', update)* is created if a query is routed through the local peer. *Topic* is the set of query topics from the query q. The *pid* refers to the unique identifier of the querying remote peer. Since there is no information about the number of results retrieved for the query, we set the number of query hits to 1. *maxsim* indicates the similarity between the *topic* of the shortcut and the locally stored content. Finally *'r'* indicates the shortcut is a recommender shortcut and *update* is the time, when the shortcut was created or the last time, when the shortcut was used. In Figure 1 Peer 2 issues the query /Top/Education/UML. Peer 8 creates a shortcut to peer 2 since this query was routed through peer 8.

Content Provider and Recommender Index. The volatility of the peers in the network and their interest shifts require to update the local indices. INGA assumes that each peer may only store a limited number of shortcuts, viz. only knows a

limited set of topic specific neighbors it can route a query to. If the local index size is reached a peer decides, which shortcuts should be deleted from the index. For each shortcut in the index INGA computes a rank based on the following types of localities:

Semantic locality. INGA measures the maximum semantic similarity $maxsim$ between the topic of a shortcut and the topics represented by the local content of a peer according to equation 1. Hence, INGA retains a shortcut about topic t to a remote peer, if t is close to its interests.

LRU locality. To adapt to changes in the content and interests INGA uses a LRU replacement policy [21]. Each local shortcut comes with a time stamp reflecting the latest point in time at which it was created or used by the local peer for query forwarding. Thus, each peer has an 'oldest' and a 'most recent' shortcut, for which $update$ values of 0 and 1 are assigned, respectively. Time points in between receive $update$ values by linear interpolation between the two extremes.

Community locality. The community locality takes into account the minimal distance between a shortcut and the respective content. Content provider shortcuts marked with a c provide a one hop distance to the content, viz. $type = 1$. Recommender shortcuts, marked with an r require at least two hops to reach a peer with relevant documents, viz. $type = 0.5$.

INGA determines the relevance of an index entry by a weighted sum of its different locality values (equation 3). Shortcuts with the lowest relevance are discarded first.

$$relevance = \frac{a * maxsim + b * type + c * update}{a + b + c} \quad (3)$$

B. Bootstrapping Shortcuts

Bootstrapping shortcuts link to peers which are well connected in the network. INGA determines the bootstrapping capability by analyzing the in-degree and out-degree of a peer. The out-degree of a peer is a measure for its ability to discover remote peers. The in-degree of a peer measures its popularity.

In order to disseminate the bootstrapping capability of a peer its queries include this information. While a peer is online it continually updates its content/recommender index based on incoming queries and stores additional bootstrapping shortcuts in the form *sc(pid, bo)*, where *pid* is the PID of the querying peer and *bo* its bootstrapping capability. Once an initial set of bootstrapping nodes is found, a peer may route its queries to the nodes with the highest bo value, which is calculated according to Equation 4.

$$Bo = (1 + |outdegree|) \times (1 + |indegree|) \quad (4)$$

In Equation 4 *out-degree* refers to the number of distinct remote peers a peer knows. To compute the *in-degree* INGA counts the number of distinct remote peers that route a query through the peer. INGA extracts this information from the message path mp.

A peer stores bootstrapping information only about remote peers which are part of the content provider or recommender layer. This ensures that not all peers use the same bootstrapping peer, because the selection of remote peers partly depends on the local content.

C. Default Network Shortcuts

When a new peer enters the network, it has not yet stored any specific shortcuts in its index. Default network shortcuts connect each peer p to a set of other peers (p's neighbors) chosen at random, as in typical Gnutella-like networks (e.g. using rendezvous techniques).

IV. Dynamic Shortcut Selection

The basic principle laying behind the shortcut mechanism consists of dynamically adapting the topology of the P2P network so that the peers that share common interests spontaneously form well-connected semantic communities. [22] shows that each user is only interested in a limited number of different topics. Therefore being part of a community that shares common interests is likely to increase search efficiency and success rate. To optimize the overall message traffic this paper propose a dynamic shortcut selection strategy, where each peer selects only a certain number *k* of most promising shortcuts for query forwarding.

A. Overview

INGA consists of several steps executed locally and across the network when recommending peers for a query and retrieving or returning results. If a query is submitted to the P2P network the following process is executed:

- **Across the network: Recommending.** Whenever a peer receives a query message, it first extracts meta-information about the querying peer and updates its bootstrapping and recommender index if needed. Then the INGA forwarding strategy is invoked to select a set of *k* peers that appear most promising to successfully answer the query . The original query message is forwarded to these *k* peers.
- **Across the network: Answering Queries.** When a peer receives a query, it will try to answer the query with local content. INGA only returns non-empty, exact results and routes them directly to the querying peer. If the maximum number of hops is not yet reached, the query is forwarded to a set of remote peers selected as described above.
- **Locally: Receiving Results.** On the arrival of result items a querying peer analyzes the message path and the respective number of results to create or update local content provider shortcuts.

B. Selecting best matching shortcuts

The INGA shortcut selection algorithm determines the candidate peers that are most promising to forward the given query to. The INGA strategy is based on the local knowledge about the query topic as stored in the index of the peer:

- INGA only forwards a query via its *k best matching* shortcuts.
- INGA prefers content and recommender shortcuts over bootstrapping and default network shortcuts.
- The INGA strategy is a greedy *k best*-search heuristics. As such it might be led astray into a subnetwork of peers that appear to be the optimal choice from a local point of view, but that do not yield all the appropriate answers.

To let the search escape such local optima, some queries are forwarded to a random set of peers.

This randomness will later on show two major beneficial effects: First, it allows the individual peer to have a larger overview of the whole network and, hence, to establish the appropriate short distance *and* long distance shortcuts.[3] Second, it facilitates accommodation to volatility (especially in the form of new joining peers).

Algorithm 1 defines the basic peer selection procedure for choosing k peers: In step 1 it selects at most k peers from content or recommender shortcuts that match the topic of the query with the highest similarity. To avoid forwarding queries along shortcuts with only low topic similarity a minimum similarity threshold t_{greedy} is required to hold between the topic(s) of the query and the shortcut. If less then k shortcuts have been found, the algorithm selects the top bootstrapping shortcuts (step 3). Finally, remaining slots for query forwarding are filled by a random selection from the default network. The algorithm terminates if the query has reached its maximum number of hops. Furthermore, the algorithm is constrained such that a query is not forwarded to a peer if this peer has already occurred in the message path of the query (step 6).

Algorithm 1 Dynamic

Require: **Query** q, **MsgPath** *mp*, **int** k, **float** t_{Greedy}, **float** f, **Set** *topicDependentShortcuts*, **Set** *bootstrappingShortcuts*, **Set** *defaultNetworkShortcuts*
Ensure: $TTL_q < maxTTL$
1: **Set** $s \leftarrow$ TopGreedy(*q*,*topicDependentShortcuts*,*k*,t_{Greedy})
2: **if** ($|s| < k$) **then**
3: $s \leftarrow$ TopBoot(*s*, *bootstrappingShortcuts*, k)
4: **end if**
5: $s \leftarrow$ RandomFill(*s,defaultNetworkShortcuts,f,k)*
6: $s \leftarrow$ removeAlreadyVisitedPeers(*s*,*mp*)
7: **Return** s.

The remainder of this section describes the subroutines of the INGA algorithm with more details.

Algorithm *TopGreedy* allows for selecting the top peers above a similarity threshold. The algorithm browses trough the index of all topic dependent shortcuts (step 3) and identifies the most similar shortcuts for a query (step 4) above t_{greedy} (step 5). If two shortcuts have the same similarity, it selects the shortcut with the higher value of query hits (not shown in the algorithm below). The algorithm selects the top-k peers for a query (step 7).The *TopBoot* Algorithm works similarly to the *TopGreedy* Algorithm, but selects the peers with highest known bootstrapping capability (line 3).

The task of algorithm *RandomFill* is twofold: if the other subroutines fail to discover k peers for a query, it fills up remaining peers until k is reached. The second task of the algorithm is to contribute some randomly chosen peers to the selected set of k peers to avoid overfitting of the selection process as known from simulated annealing techniques. Depending on the probability f the algorithm exchanges already selected peers with randomly chosen ones. [3] elaborates on the effects of different random contribution levels and finds that $f = 20\%$ produces the best results.

[3] 'short' and 'long distance' as seen from the default underlying network.

Algorithm 2 TopGreedy

Require: **Query** q, **Set** *topicDependentShortcuts*, **int** k, **float** t_{greedy}
1: **Set** *topShortcuts* $\leftarrow\{\}$
2: **Set** *s_tmp* $\leftarrow$ *topicDependentShortcuts*
3: **while** (*s_tmp* is not empty) $\wedge$ ($|$*topShortcuts*$| < k$) **do**
4: *Next* $\leftarrow$ argmax$_{p \in s_tmp}$ sim$_{\text{Topic}}(q, p)$
5: **if** sim$_{\text{Topic}}$(*q*,*Next*) $\leq t_{greedy}$ **then**
6: break
7: **end if**
8: *s_tmp* $\leftarrow$ *s_tmp* $- \{$*Next*$\}$
9: *topShortcuts* $\leftarrow$ *topShortcuts* $\cup \{$*Next*$\}$
10: **end while**
11: **Return** *topShortcuts*

Algorithm 3 TopBoot

Require: **Set** *topShortcuts*, **Set** *bootstrappingShortcuts*, **int** k
1: **Set** *s_tmp* $\leftarrow$ *bootstrappingShortcuts*
2: **while** (*s_tmp is* not empty) $\wedge$ ($|$*topShortcuts*$| < k$) **do**
3: *Next* $\leftarrow$ argmax$_{p \in s_tmp}$ topBoot(p)
4: *s_tmp* $\leftarrow$ *s_tmp* $- \{$*Next*$\}$
5: *topShortcuts* $\leftarrow$ *topShortcuts* $\cup \{$*Next*$\}$
6: **end while**
7: **Return** *topShortcuts*

V. Experimental Setup

INGA was conceptually evaluated in a real world case study [18] while different parameter settings were compared using a simulation environment (the focus of this paper). This section describes the experimental setup while Section VI presents the evaluation results.

A. Content Distribution

Three different data sets – one synthetic and two data sets from real-world observations – build the basis of the evaluation. The data sets exhibited different characteristics with regard to dimensions like size, relational structure (i.e. being able to be used for simulation runs with conjunctive queries) and 'being natural' for the task at hand. All three data sets had a low level of replication, i.e. few data items were assigned to multiple peers, and all data sets exhibited a hyperbolic (Zipf-like) distribution of topics.

Open directory project. The first data set is based on the open directory *DMOZ.org*. *DMOZ.org* constitutes a *large* data set of content distributed among a substantial community of content editors. The data set was so large that for the purpose of our simulations we have selected a subset consisting of the first three levels of the DMOZ hierarchy.

Each editor is responsible for one or several content topics and maintains the corresponding topic pages. At the first three levels we found 1657 topics. There is a Zipf-like skew in the distribution of editors to topics: 991 editors only maintain content about one topic, 295 about two, 128 about three, ...

one editor about 20, and one editor maintains content about 22 topics. Vice versa: 755 topics are dealt with by 1 editor, 333 by 2, 204 by 3, . . . , 44 by 6, . . . ,14 by 10, and 1 topic has 32 editors. Thus, mapping an individual editor onto an individual peer and the topic content of the editor onto the local content database of the corresponding peer appears to be a rather natural, realistic choice for a basic data set. This data set does contain very few relational structures, and predominantly information about instantiation of one topic, e.g. '$AirplaneArrivals2004.pdf'$ is an instance of topic $topic\ '/TourismActivity/TravelDistribution/docs'$. It is thus not possible to test conjunctive queries on this data set, as only 96 of the 43894 different instances are assigned to more than one topic.

Synthetic data. The number of classes, the number of properties and the number of sub-class relationships together with their respective distributions determine the schema of the ontology. The number of instances and the number of relations between instances determine the distribution of the instance data. The distributions are modeled as a Zipf distribution with parameter settings according to observations from real world data sets. The parameter settings for the schema generation are based on [23] while the parameter settings for instance generation are based on [24]. Data distribution on the peers follow the model presented in [22].

For the schema we choose to generate an ontology with 1.000 classes and each class was assigned a popularity based on a Zipf distribution with skew factor 1. The popularity of a class influences its number of instances, its replication in the network and its connectivity with other classes through properties. We selected the number of sub-classes and the number of properties of a class (Zipf with skew factor 1.1). This resulted in 357 properties.

200,000 instances were generated and assigned to one class based on the popularity of the different classes. $TotalNoPropertyInstances = 100 * TotalNoProperties$ many properties between instances were generated. Likewise to classes, a property schema (one could also call it a binary relation), had a popularity based on a Zipf distribution with skew factor 1 that was considered when generating properties between instances (i.e. when populating the binary relations).

Assignment of data to peers is done based on the conjecture that users are generally interested in a small subset of the entire content available in a peer-to-peer network. We have modeled that the interests are more likely in only a limited number of content classes and thus users would be more interested in some classes while less in others. The maximum number of classes that a peer is interested in and its content is computed by $ClassesOfInterest = ln(NoOfClasses) * 2$. The actual number of classes is chosen randomly from a uniform distribution. Observing the studies in [24] all peers do not share the same amount of data and also do not exhibit the same 'social behaviour'. For instance, a large number of users are so-called free riders or freeloaders who do not contribute anything to the network but essentially behave like clients. On the other hand, a small number of users (less than 5%) provide more than two thirds of the totally available amount of data and thus behave like servers. Considering the study in [25] the following storage capacity was assigned to the peers in the network: 70% of the peers do not share any instances (free riders); 20% share 100 instances or less; 7% share 101 up to 1000 instances and finally, only 3% of the peers share between 1001 and 2000 instances.

Bibster data set. This data set bases on real captured query data from the peer-to-peer bibliography network 'Bibster' [18]. The data set contains 27.037 distinct bibliographic entries. These are categorized according to the ACM topic hierarchy for computer science.[4]

B. Query Distribution

For the different data sets and their assignments to peers, we generated queries in the simulation runs as follows:

Open directory project. Queries were generated in the experiments by instantiating the blueprint query $(*; \textit{rdf:type}; topic)$ with topics arbitrarily chosen from the set of topics that had at least one document. We generated 30000 queries, uniformly distributed over the 1657 different topics. We choose a uniform query distribution instead of a ZIPF-distribution, which is typically observed in file sharing networks [26]. This simulates the worst case scenario, where we do not take advantage of often repeated queries for popular topics.

Synthetic data. The query set for the synthetic ontology is based on a special type of queries that request instances satisfying a varying number of constraints. The basic concept for the queries is built on the following schema:

$(instance; \textit{rdf:type}; class) \wedge (instance; \textit{owl:hasProperty}; instance2) \wedge (instance2; \textit{rdf:type}; class2) \wedge (instance; \textit{owl:hasProperty}; instance3) \wedge (instance3; \textit{rdf:type}; class3)$.

Informally, this concept requests all $instances$ of a certain $class$ with the constraint that the $instances$ have two particular $properties$ pointing to other $instances$. An example of a SeRQL query is:

```
construct * from {instance}
   <!http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
      {<!http://swap.simulation#class0>};
   <!http://swap.simulation#property2> {}
         <!http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
            {<!http://swap.simulation#class51>},
{instance}
   <!http://swap.simulation#property1> {}
         <!http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
            {<!http://swap.simulation#class37>}
```

The built query set only contains queries that can be answered by the network and is uniformly distributed .

Bibster data set. The extracted queryset for the Bibster-based ontology contains several types of queries that request instances satisfying different constraints. It is worth mentioning that the queryset was adopted the way it was originally created and therefore, it also consists of non-conjunctive queries.

C. Peer-to-Peer Network Setup

Gnutella style network. The simulation is initialized with a network topology which resembles the small world properties of file sharing networks[5] with 1024 peers. In the simulation, peers were chosen randomly and they were given a randomly selected query to question the remote peers in the network. The

[4]The ACM-Index file is available online at: http://www.aifb.uni-karlsruhe.de/WBS/pha/bib/acmtopics.rdf

[5]We used the Colt library `http://nicewww.cern.ch/~hoschek/colt/`

Parameter	Value
Queries	30.000
Queries per peer	ca. 30
Query time to life	6
Selected peers per query (k)	2
Greedy search threshold (t_{Greedy})	15 %
Random contribution (f)	20 %
Index size (if no other size is mentioned)	40
Open Directory data set	
Topics	1646
Before interest shift	823
After interest shift	823
Simulated peers	1024
Bibster data set	
Topics	1293
Simulated Peers	520
Synthetic data set	
Topics	1000
Simulated Peers	1024

TABLE I
SIMULATION PARAMETER SETTING

peers decide on the basis of their local short cut index which remote peers to send the query to. Each peer uses INGA to select up to $pmax = 2$ peers to send the query to. Each query was forwarded until the maximal number of hops $hmax = 6$ was reached — unless the peer selection algorithm choose not to forward further also at an earlier point in time.

Volatile network and interest shifts. We implemented the dynamic network model observed for Gnutella networks of [26]: 60% of the peers have a availability of less then 20%, while 20% of the peers are available between 20 and 60% and 20 % are available more then 60%. Hence only a small fraction of peers is available more than half of the simulation time, while the majority of the peers is only online a fraction of the simulation time. Users' interests may change over time, e.g. to account for different search goals. To simulate changing interests, after 15 queries, equal to ca. 15.000 queries over all peers, each peer queries a completely different, previously unused set of topics.

D. Simulator setup and simulation statistics

We used a round based simulation framework which was setup using the parameter setting listed in Table I. In total we simulated 1024 peers. To determine the standard error of our observations of 95 percent confidence interval ($p<0.05$) each simulation was executed six times. We set the greedy search threshold for algorithm to 0.15 and the amount of random contribution to 0.20

E. Evaluation Measures

INGA is evaluated using the following metrics:

- **Recall** describes the proportion between all relevant documents in peer network and the retrieved ones. Hereby, we defined 'relevant' as 'matches the query'. We did not use any gold standard document set where relevance to a query would be assigned by a user. Therefore, *precision* would have been meaningless in our evaluation.
- **Messages** represent the required search costs per query thus measuring system scalability.
- **Message Gain** is defined as the recall per message, hence we divide the recall of a query with the proportion of messages to achieve the recall.

VI. EXPERIMENTAL RESULTS

INGA was evaluated in a large number of experiments in order to compare its performance to related state-of-the-art approaches, to set the optimal parameter setting and to test the applicability of the algorithm for different scenarios. Before the presentation of the final evaluation results, the following list summarizes the major hypotheses that were investigated:

1) Shortcut networks outperform the *Default* approach.
2) INGA outperforms state-of the art shortcut networks.
3) Semantic similarity supports the peer selection process. It helps to improve recall and to reduce the number of messages. However, shortcut networks perform reasonably even without the support of a semantic similarity function.
4) Each layer contributes to improve routing efficiency. Depending on the scenario, dynamic bootstrapping peers help to reduce the number of messages, while recommender peers increase the recall.
5) Our algorithms performs well with a limited index size.
6) Combining different index policies supports efficient routing much more than relying on a simple LRU strategy.
7) Shortcut networks are capable to handle conjunctive queries efficiently.
8) Shortcut networks perform well in both, dynamic and static, networks.

[27], [28] present a more detailed analysis of the evaluation results and additionally investigate the influence of different parameter settings on network characteristics, such as the clustering coefficient.

A. INGA outperforms state-of-the-art approaches

As a baseline we compare INGA with an index size of 40 entries against the interest based locality strategy (IBL) of [13] with an LRU strategy and an index size of 40 entries and the default algorithm of Gnutella (*Default*).

During the simulation run each peer issues in average 30 queries approximately corresponding to a 2 hour period in a real file sharing network. After 15 queries the peers shift interest and choose queries from the second half of the available set of queries. Studies of interest shift show a smoother transition between interests positively influencing the performance of all routing approaches [29].

Figure 2 focuses on the recall achieved by different routing strategies. The total number of issued queries is plotted against the achieved recall. The line 'Online available' indicates the maximally achievable recall as not all peers are always online. The recall of INGA increases with the number of queries sent and levels off at about 25% corresponding to 46% of the achievable recall. In the event of interest shift the recall nearly halves. It recovers slightly but does not achieve its former levels. The continued black line represents the mean of eight simulations runs with the same parameter setting. The gray range around it additionally visualizes the 95% confidence

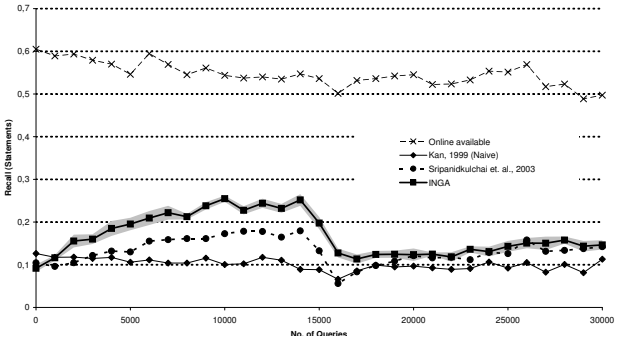

Fig. 2. Recall: Related Approaches

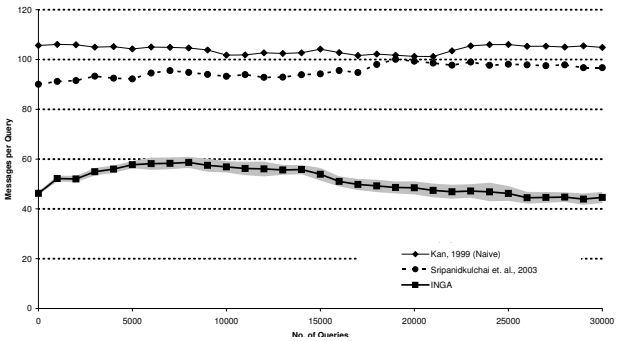

Fig. 3. Messages: Related Approaches

interval for the simulation results. The range shows that INGA is statistically significantly better than the related approaches. The IBL strategy produces recall levels of around 18% or 33% of the achievable recall with a drop in the event of an interest shift. The naive routing strategy results in circa 10% (19% of achievable) recall independently of the interest shift.

Figure 3 plots the number of messages per query against the total number of sent queries. The number of messages per query produced by INGA decreases over time independently of the interest shift and levels off at around 45. The number of messages produced by the naive approach is constant at around 105. The number of messages produced by IBL increases a little bit and levels off at around 95 messages. The interest shift does not affect the number of messages produced by the IBL strategy.

The simulations validate the hypothesis that INGA has a better performance than related routing approaches for dynamic unstructured P2P networks. In a volatile network the bootstrapping peers correspond to the peers which are most of the time online. This behavior results in a decreasing number of messages per query over the entire simulation, but it also limits the discovery of new peers.

B. Layer and semantic similarity function contribution

Figure 4 illustrates the contribution of the different layers to the performance of INGA. Each chart is the result of using the indicated layer and all lower layers, e.g., the content provider layer uses also the default network layer. The figure plots the message gain against the total number of sent queries. The default network layer follows a naive routing strategy and offers a constant but low performance contribution. The use

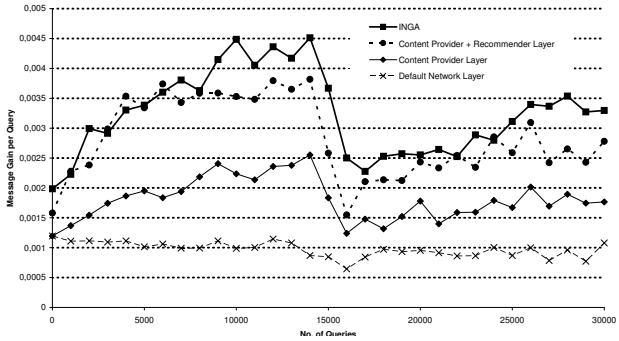

Fig. 4. Message gain for each layer

of content provider layer steadily increases the recall, while it leaves the number of messages per query unaffected. In the event of new queries the recall falls back and recovers only slowly. The recommender layer allows the algorithm to learn quicker and to achieve a higher recall. Finally, the bootstrapping layer reduces the number of messages per query, but slightly decreases the recall. In total, however, the message gain is at its highest level using all three layers.

The introduction of the recommender layer has the strongest influence on the performance of INGA. One reason is the caching of frequent queries issued by the peers in the network. If the network becomes clustered, especially neighboring nodes with similar interests will benefit of cached queries. Furthermore query routing is based on a similarity function, so queries are routed along shortcuts representing similar queries. Especially for nodes that are not clustered so far, this similarity helps to find adequate clusters and route queries to nodes that have similar interests. Thus, including a similarity in the peer selection process speeds up the clustering process, however shortcut networks will exploit small world characteristics even if an exact match paradigm is used only. In this case shortcut networks profit in particular from popular queries that are used to establish topic specific shortcuts between remote peers.

C. Influence of the index size

Figure 5 presents the evaluation results comparing the performance of INGA using different index sizes. The figure plots the message gain against the total number of sent queries. The size of the index determines the required resource allocation for routing purposes at the local peer. The index size is constantly increased starting from 20, to 40, 100 and finally unlimited number of shortcuts per peer.

We observe that within a certain range the limit to the number of shortcuts stored at the local peer does not affect the results in terms of message gain. Peers store different shortcuts, because the ranking of the shortcuts partially depends on the local knowledge of the peer. Although the number of local shortcuts is low, the collective number of different shortcuts is high. This allows INGA to achieve the performance levels already with a relatively small index size. An index size between 40 and 100 is sufficient.

D. Influence of the index weight

Figure 6 plots the message gain against the total number of sent queries for four different parameter settings and the naive

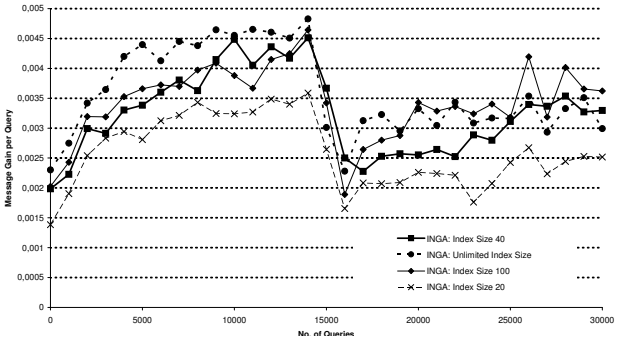

Fig. 5. Message gain for different index size

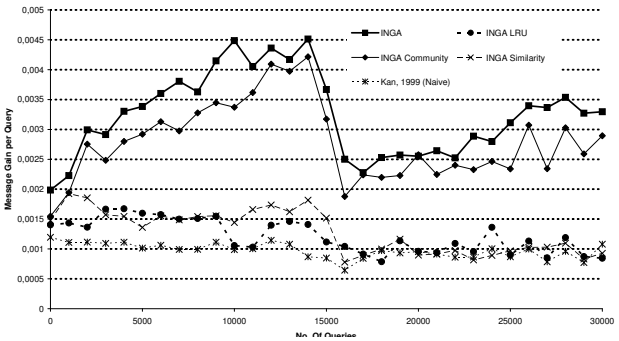

Fig. 6. Message gain for different index weights

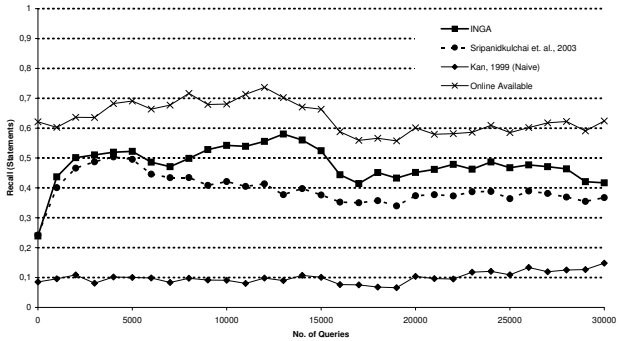

Fig. 7. Recall for conjunctive queries

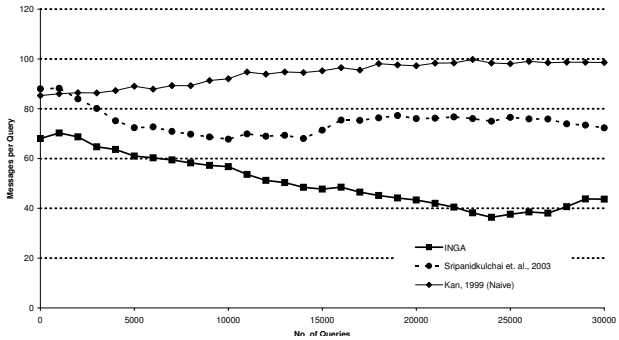

Fig. 8. Messages for conjunctive queries

routing approach. In each simulation only the values set for a, b, and c are changed. In INGA all three parameters influence the index management. The parameters are set to $a = 0.1, b = 0.8$ and $c = 0.1$. INGA LRU corresponds to the parameter setting in which only the time is considered to rank shortcuts for index maintenance ($a = 0, b = 0, c = 1$). Similarly INGA community considers only the shortcut type ($a = 0, b = 1, c = 0$). Finally, INGA similarity considers only the similarity of shortcuts to the local content ($a = 1, b = 0, c = 0$). The pure indexing strategies only based on similarity and time cannot increase the message gain levels above the levels observed for the naive algorithm. Only with the community strategy message gain levels comparable to INGA are achievable. No parameter setting alleviates the problem that recall levels do not recover to levels observed during the learning phase. The bootstrapping peers receive a larger proportion of the queries, reducing the number of messages but also the recall as they are not aware of peers able to answer the new set of queries.

The community locality raises the message gain, even after changing the interests of each peer, while the combined strategy performs best.

E. Performance for conjunctive queries

Figure 7 compares the recall for different routing algorithms using conjunctive queries on the synthetic data set[6]. After a warm up phase of 2.000 queries, or approximately two queries per peer, INGA constantly reaches around 75% of the available content. The observed recall for the Gnutella routing algorithm (Kan, 1999 (Naive)) reaches the same level as for none conjunctive queries, while the IBL approach (Sripanidkulchai et. al., 2003) shows a better performance. The different data distributions in the two data sets explain the difference. Not all peers are always online, thus the line *Online Available* represents the maximum available content at query time.

Figure 8 visualizes the number of messages produced to achieve this recall. The number of messages produced by the Gnutella approach slightly increases over time. Due to the high network churn the peers have to discover new remote peers since the available ones assigned in the setup phase are off line. Thus, the necessity to send a query to a remote peer which has not received the query yet increases over time. The IBL approach produces less messages than the Gnutella approach. In contrast to the observation made for related approaches the number of messages produced based on the shortcut selection decreases significantly. The number of messages decreases because of the small world properties of the network: queries are only forwarded to a focused set of peers; and it decreases, because INGA does not forward queries to peers that have already received a query (see Section IV and Algorithm 1).

VII. SUMMARY

The INGA routing algorithm dynamically adapts to the network topology, building on the history of successful and semantically similar queries. This is realized using bounded local shortcut indexes, storing semantically labelled shortcuts, and dynamically selecting shortcuts. The selection algorithm forwards queries to a community of peers that are most promising to successfully answer a query. Shortcuts connect peers that share similar interests and thus spontaneously form semantic communities.

[6]The results using the Bibster data set are comparable to the results using the synthetic data set. [27] includes a detailed presentation of those results.

Extensive simulations show that the application of the INGA routing algorithm results in a higher recall or a reduction of messages per query in comparison to other routing approaches. INGA is non-intrusive as it is solely based on the observation of network behavior keeping the load for administration messages at nil. Thus, INGA exhibits a characteristics of properties that sets it apart in intriguing ways from structured DHT-style networks and it may provide a substantial benefit to all unstructured peer-to-peer networks.

ACKNOWLEDGMENT

Research reported in this paper has been partially financed by EU in the IST projects SEKT (IST-2003-506826), SWAP (IST-2001-34103) and ASG (IST FP6-004617). Alexander Löser was supported by the German Research Foundation, Berlin-Brandenburg School for Distributed Information Systems (DFG grant GRK 316/3).

REFERENCES

[1] A. Löser, C. Tempich, B. Quilitz, W.-T. Balke, S. Staab, and W. Nejdl, "Searching dynamic communities with personal indexes," in *3rd. Int. Semantic Web Conference (ISWC)*, 2005.

[2] C. Tempich, A. Löser, and J. Heizmann, "Community Based Ranking in Peer-to-Peer Networks," in *Int. Conference on Ontologies, Databases and Applications of Semantics (ODBASE)*, 2005.

[3] C. Tempich, S. Staab, and A. Wranik, "REMINDIN': Semantic Query Routing in Peer-to-Peer Networks based on Social Metaphers," in *13th World Wide Web Conference (WWW)*, 2004.

[4] S. Milgram, "The small world problem," *Psychology Today*, vol. 67, no. 1, 1967.

[5] J. Kleinberg, "Navigation in a small world," *Nature*, no. 406, p. 845, 2000.

[6] B. Loo, J. Hellerstein, R. Huebsch, S. Shenker, and I. Stoica, "Enhancing p2p file-sharing with an internet-scale query processor," in *30th. Int. Conference on Very Large Databases (VLDB)*, 2004.

[7] S. Stuckenschmidt and S. Staab, Eds., *Semantic Web and Peer-to-Peer*. Springer, 2005.

[8] L. Gravano and H. García-Molina, "Generalizing GlOSS to vector-space databases and broker hierarchies," in *VLDB-95*, 1995.

[9] K. Aberer, P. Cudre-Mauroux, M. Hauswirth, and T. van Pelt, "GridVine: Building Internet-Scale Semantic Overlay Networks," in *3rd. Int. Semantic Web Conference (ISWC)*, 2004.

[10] S. Androutsellis-Theotokis and D. Spinellis, "A survey of peer-to-peer content distribution technologies," *ACM Comput. Surv.*, vol. 36, no. 4, 2004.

[11] P. J. Keleher, B. Bhattacharjee, and B. D. Silaghi, "Are virtualized overlay networks too much of a good thing?" in *Revised Papers from the 1st Int. Workshop on Peer-to-Peer Systems (IPTPS)*, 2002.

[12] D. Tsoumakos and N. Roussopoulos, "A comparison of peer-to-peer search methods," in *6th Int. Workshop on Web and Databases (WebDB) at SIGMOD2003*, 2003.

[13] K. Sripanidkulchai, B. Maggs, and H. Zhang, "Efficient content location using interest based locality in peer-to-peer system," in *22nd Int. Conf. of the IEEE Computer and Communications Societies (INFOCOM)*, 2003.

[14] V. Kalogeraki, D. Gunopulos, and D. Zeinalipour-Yazti, "A Local Search Mechanism for Peer-to-Peer Networks," in *11th. Int. Conference on Information and Knowledge Management (CIKM)*, 2002.

[15] D. Tsoumakos and N. Roussopoulos, "Adaptive probabilistic search for peer-to-peer networks." in *3rd Int. Conference on Peer-to-Peer Computing (P2P2003)*, 2003.

[16] B. Cooper, "Guiding queries to information sources with infobeacons," in *ACM/IFIP/USENIX 5th Int. Middleware Conference*, 2004.

[17] F. M. Cuenca-Acuna, C. Peery, R. P. Martin, and T. D. Nguyen, "PlanetP: Using Gossiping to Build Content Addressable Peer-to-Peer Information Sharing Communities," Department of Computer Science, Rutgers University, Tech. Rep. DCS-TR-487, 2002.

[18] P. Haase, J. Broekstra, M.Ehrig, M. Menken, P.Mika, M. Plechawski, P. Pyszlak, B. Schnizler, R. Siebes, S. Staab, and C. Tempich, "Bibster - a semantics-based bibliographic peer-to-peer system," in *3rd. Int. Semantic Web Conference (ISWC)*, 2004.

[19] O. Lassila and R. Swick, "Resource description framework (RDF) model and syntax specification," World Wide Web Consortium, Bosten, W3C Recommendation, 1999, http://www.w3c.org/TR/WD-rdf-syntax.

[20] Y. Li, Z. Bandar, and D. McLean, "An approach for measuring semantic similarity between words using multiple information sources." *IEEE Trans. Knowledge Data Eng.*, vol. 15, no. 4, 2003.

[21] A. V. Aho, P. J. Denning, and J. D. Ullman, "Principles of optimal page replacement," *J. ACM*, vol. 18, no. 1, 1971.

[22] A. Crespo and H. Garcia-Molina, "Semantic Overlay Networks for P2P Systems," Comp. Science Dep., Stanford University, Tech. Rep., 2002.

[23] C. Tempich and R. Volz, "Towards a benchmark for semantic web reasoners - an analysis of the DAML ontology library," in *Workshop on Evaluation of Ontology-based Tools (EON2003) at 2nd Int. Semantic Web Conference (ISWC)*, 2003.

[24] V. Cholvi, P. Felber, and E. Biersack, "Efficient search in unstructured peer-to-peer networks," *European Transactions on Telecommunications: Special Issue on P2P Networking and P2P Services*, no. 15, 2004.

[25] E. Adar and B. Huberman, "Free riding on gnutella," *First Monday*, vol. 5, no. 10, 2000.

[26] S. Saroiu, P. K. Gummadi, and S. D. Gribble, "A measurement study of peer-to-peer file sharing systems," *Multimedia Systems*, vol. 9, no. 2, 2003.

[27] C. Tempich, "Ontology engineering and routing in distributed knowledge management applications," Ph.D. dissertation, Universität Karlsruhe (TH), 2006.

[28] A. Löser, "Adaptive Overlays in Peer-to-Peer Netzwerken," Ph.D. dissertation, Technische Universität Berlin, 2005.

[29] J. Liang, R. Kumar, Y. Xi, and K. Ross, "Pollution in p2p file sharing systems," in *IEEE INFOCOM*, 2005.

Alexander Löser recently finished his PhD at the University of Technology Berlin (TUB), at Institute for Software Engineering and Theoretical Computer Science and is now a PostDoc with IBM Research Almaden. Learn more about him at `http://cis.cs.tu-berlin.de/~aloeser`.

Steffen Staab is an associate professor for databases and information systems at University Koblenz-Landau where he heads the lab for information systems and semantic web. Learn more about him at `http://isweb.uni-koblenz.de`.

Christoph Tempich is an PhD student at the Institute of Applied Computer Science and Formal Description Methods (AIFB) at the University of Karlsruhe (TH). Learn more about him at `http://www.aifb.uni-karlsruhe.de/WBS/cte`.