Super-Peer-Based Routing and Clustering Strategies for RDF-Based Peer-To-Peer Networks

Wolfgang Nejdl, Martin Wolpers, Wolf Siberski, Christoph Schmitz,
Mario Schlosser, Ingo Brunkhorst[*] , Alexander Löser[†]

ABSTRACT

RDF-based P2P networks have a number of advantages compared with simpler P2P networks such as Napster, Gnutella or with approaches based on distributed indices such as CAN and CHORD. RDF-based P2P networks allow complex and extendable descriptions of resources instead of fixed and limited ones, and they provide complex query facilities against these metadata instead of simple keyword-based searches.

In previous papers, we have described the Edutella infrastructure and different kinds of Edutella peers implementing such an RDF-based P2P network. In this paper we will discuss these RDF-based P2P networks as a specific example of a new type of P2P networks, schema-based P2P networks, and describe the use of super-peer based topologies for these networks. Super-peer based networks can provide better scalability than broadcast based networks, and do provide perfect support for inhomogeneous schema-based networks, which support different metadata schemas and ontologies (crucial for the Semantic Web). Furthermore, as we will show in this paper, they are able to support sophisticated routing and clustering strategies based on the metadata schemas, attributes and ontologies used. Especially helpful in this context is the RDF functionality to uniquely identify schemas, attributes and ontologies. The resulting routing indices can be built using dynamic frequency counting algorithms and support local mediation and transformation rules, and we will sketch some first ideas for implementing these advanced functionalities as well.

Categories and Subject Descriptors

C.2 [**Network Protocols**]: Routing protocols; H.3.5 [**Information Storage and Retrieval**]: Web-based services; I.2.4 [**Knowledge Representations Formalisms and Methods**]: Semantic Networks

General Terms

Design, Algorithms

Keywords

Peer-to-Peer, Semantic Web, Schema-Based Routing, Distributed RDF Repositories

[*]Learning Lab Lower Saxony, University of Hannover, 30167 Hannover, Germany, {nejdl,wolpers,siberski,schmitz,schlosser, brunkhorst}@learninglab.de

[†]Computer Informations Systems Institute, Technical University Berlin, 10587 Berlin, Germany, aloeser@cs.tu-berlin.de

WWW2003, May 20–24, 2003, Budapest, Hungary.
ACM xxx.

1. SUPER-PEER NETWORKS FOR DISTRIBUTED RDF REPOSITORIES

Peer-to-peer (P2P) networks have become an important infrastructure during the last years, and P2P networks have evolved from simple systems like Napster and Gnutella to more sophisticated ones based on distributed indices (e.g. distributed hash tables) such as CAN and CHORD [12] and [16]. Still, while these new systems do provide more efficient topologies than early P2P networks, they neither address more complex metadata sets nor do they support more complex queries.

In the Semantic Web, an important aspect for its overall design is the exchange of data among computer systems without the need of explicit consumer-producer relationships. RDF and RDF Schema are used to annotate resources on the Web thus providing the means by which computer systems can exchange and comprehend data. All resources are uniquely identifiable by an URI. The annotations about resources are based on various schemas that are built based on RDFS (and possible extensions) and are stored in what we call RDF repositories possibly using more than one schema.

One important characteristic of RDF metadata is the ability to use distributed annotations for one and the same resource. In contrast to traditional database systems, it is not necessary that all annotations of a resource are stored on one server. One server might store metadata which include properties such as name for specific resources possibly using the Dublin Core metadata standard. Other servers also could hold metadata that provide properties for the same resources, possibly using other metadata standards / schemas. This ability for distributed allocation of metadata makes RDF very suitable for the construction of distributed repositories.

Furthermore RDF schemas are flexible and extendable such that schemas can evolve over time, and RDF allows the easy extension of schemas with additional properties. As such RDF is capable of overcoming the problems of fixed and unchangeable metadata schemas which often occur in recent peer-to-peer (P2P) systems. Current P2P systems however support only limited metadata sets such as simple filenames [7], so it is easy for example to search Gnutella for music composed by Beethoven, but retrieving all his symphonies is much more difficult. To solve the shortcomings of P2P networks with restricted and fixed metadata elements and in order to enable distributed repositories about resources we have to move towards more sophisticated P2P networks called schema-based P2P networks. Schema-based P2P networks build upon peers that use explicit schemas describing their content and where the metadata of peers can be based on heterogeneous schemas.

There are only a few research groups that have investigated these schema-based P2P networks so far. In our group we have been working on a schema-based network called Edutella [11] (see http://edutella.jxta.org for the source code), which aims at provid-

ing access to distributed collections of digital resources through a P2P network. Resources in the Edutella network are not described using ad hoc metadata fields (like Napster & Co), but use RDF schemas and RDF metadata for their description. In order to access content stored on the Edutella network we use the query language RDF-QEL. RDF-QEL is based on Datalog semantics and thus compatible with all existing query languages, supporting query functionalities extending the usual relationally complete query languages.

Two other interesting approaches are the ones investigated by Bernstein and Aberer. Bernstein et.al. [3] propose the Local Relational Model (LRM) enabling general queries to be translated into local queries with respect to the schema supported at the respective peer, using the concept of local translation/coordination formulas to translate between different schemas. Aberer et.al. [1] proposes schema-based peers and local translations to accommodate more sophisticated information providers connected by a Gnutella-like P2P topology.

All these approaches focus on providing improved search functionalities in P2P networks. As schemas describe the content stored at peers, query can be more precise and flexible by using these schemas. Still, if we use simple broadcast topologies for these networks, queries are broadcast to all peers and consume network bandwidth and processing power at each peer. Obviously we have to investigate more efficient approaches which query only those peers that are indeed capable of understanding and answering the query. Therefore, in schema-based networks we should use schema information not only for providing improved query capabilities, but also to support more sophisticated routing of queries. Queries and answers to queries are represented using RDF metadata which we can use together with the RDF metadata describing the content of peers to build explicit routing indices which facilitate more sophisticated routing approaches. Queries can then be distributed relying on these routing indices, which contain metadata information plus appropriate pointers to other (neighboring) peers indicating the direction where specific metadata (schemas) are used. These routing indices do not rely on a single schema but can contain information about arbitrary schemas used in the network.

In general, these routing indices could be located at each peer, but that would require a considerable amount of processing power and network bandwidth at each peer. Processing power is needed to construct the routing indices while network bandwidth is needed for sending queries and their respective answers. Furthermore peers tend to behave unpredictably, joining and leaving the P2P network at random resulting in a constant reorganization of the network topology.

Therefore, we suggest in this paper to use a super-peer topology for these schema-based networks, where each peer connects to one super-peer only. Super-peer then connect to other super-peers and build up the backbone of the super-peer network (see [18] for the general characteristics of super-peer networks, and Kazaa, Grokster and Morpheus for existing super-peer systems).

In schema-based networks, super-peers manage the routing indices and determine which query is forwarded to which peer or to which super-peer.

In the remainder of this paper we will describe the general topology of our schema-based super-peer network in section 2 and discuss the two necessary kinds of routing indices in sections 2.1 and 2.2. We will further sketch first ideas on how to construct these routing indices dynamically based on query characteristics and query distribution in section 2.3. We will then discuss in sections 2.4 on how we can use this information also for mediation between different schemas. Finally, section 3 discusses our simulation framework as well as our prototype implementation.

2. SCHEMA-BASED ROUTING IN P2P NETWORKS

P2P networks that broadcast all queries to all peers don't scale. To take the semantic heterogeneity of schema-based P2P networks into account, we therefore propose a super-peer topology for these networks and the use of indices at these super-peers to address scalability requirements. The super-peer network constitutes the "backbone" of the P2P network which tkes care of message routing and integration / mediation of metadata.

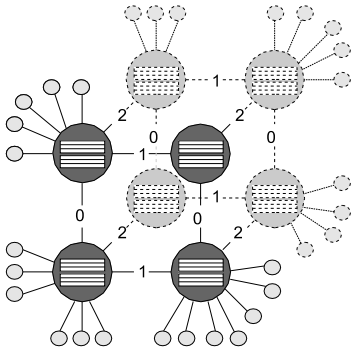

Figure 1: peers connected to the super-peer "backbone"

We will assume that the super-peers in our network are arranged in the HyperCuP topology [14]. Most solutions we propose in this paper can be realized also with other super-peer topologies, which would actually lead to interesting extensions derived from the ideas in this paper. We focus on HyperCuP, because first, it is the topology we have implemented in our super-peer network, and second, it is very efficient for broadcasts and partitioning which makes it quite suitable as a super-peer topology.

Scaling a P2P network to a large number of super-peers while maintaining certain properties such as low network diameter requires guiding the evolution of the network topology upon peer joins and departures. The HyperCuP algorithm described in [14] is capable of organizing peers in a P2P network into a recursive graph structure from the family of Cayley graphs, out of which the hypercube is the most well-known topology. We organize super-peers in the network into a hypercube topology using the HyperCuP protocol. A new super-peer is able to join the network by asking any other, already integrated super-peer which then carries out the peer integration protocol. $O(\log(N))$ messages are sent in order to integrate the new super-peer and maintain a hypercube-like topology. Any number of super-peers can be accommodated in the network: If some peers are "missing" in order to construct a complete hypercube topology which consists of 2^d nodes in a d-dimensional binary hypercube, some super-peers in the network occupy more than one position on the hypercube. When new super-peers join the network, they fill the gaps in the hypercube topology and possibly extend the dimensionality of the hypercube.

HyperCuP enables efficient query broadcasts and guarantees non-redundant broadcast. For broadcasts, each node can be thought of as the root of a specific spanning tree through the P2P network. The topology allows for $\log_2 N$ path length and $\log_2 N$ number of neighbors, where N is the total number of nodes in the network (i.e. the number of super-peers in our case). Moreover, the topology is vertex-symmetric and thus features inherent load balancing among super-peers. Thus, we can use the topology to carry out effi-

cient communication and message forwarding among super-peers: Certain updates (which we will communicate by broadcast to other super-peers) can be executed efficiently, without message overhead. Also, a path of $\log_2 N$ length exists between any two super-peers, thus any two distinct schemas can be reached within a short number of hops from each other.

Peers connect to the super-peers in a star-like fashion, providing content and content metadata (see Figure 1 for a small HyperCuP topology).

The introduction of super-peers in combination with routing indices reduces the workload of peers significantly by distributing queries only to the appropriate subset of all possible peers (see also [6] who discusses routing indices based on various aggregation strategies of content indices). In the next sections we will discuss these routing indices in more detail.

2.1 Routing Super-Peer-Peer Queries and Responses

The first kind of indices needed in super-peers are so-called super-peer/peer routing indices (SP/P-RIs). In these indices each super-peer stores information about metadata usage at each peer. This includes schema information such as schemas or attributes used, as well as possibly conventional indices on attribute values.

On registration the peer provides the super-peer with its metadata information by publishing an advertisement. This advertisement encapsulates a metadata based description of the most significant properties of the peer. As this may involve quite a large amount of metadata, we build upon the schema-based approaches which have successfully been used in the context of mediator-based information systems (e.g. [17]).

To ensure that the indices are always up-to-date, peers notify super-peers when their content changes in ways that trigger an update of the index. For example, if a peer had announced that it uses the Dublin Core schema dc during the last connection to its super-peer, but now also uses the Learning Object Metadata schema lom to describe resources, it needs to announce this to the super-peer. If a peer leaves the network, all references to this peer are removed from the indices. In contrast to some other approaches, our indices do not contain content elements but peers (as in CHORD).

At each super-peer, elements used in a query are matched against the SP/P-RIs in order to determine local peers which are able to answer the query (see also [1] for a similar approach). A match means that a peer understands and can answer a specific query, but does not guarantee a non-empty answer set. The indices can contain the information about other peers or super-peers at different granularities: schema identifiers, schema properties, property value ranges, and individual property values.

To illustrate index usage, we will use the following sample query: *find lectures in German language from the area of software engineering suitable for undergraduates.* In the Semantic Web context this query would probably be formalized using the dc schema for document specific properties (e.g. title, creator, subject) and the lom schema which provides learning material specific properties, in combination with classification hierarchies (like the ACM Computing Classification System, ACM CCS) in the subject field. In line with RDF/XML conventions, we will identify properties by their name and their schema (expressed by a namespace): "dc:subject" therefore denotes the property "subject" of the DC schema. So, written in a more formal manner, the query becomes:

> *Find any resource where the property dc:subject is equal to ccs:softwareengineering, dc:language is equal to "de" and lom:context is equal to "undergrad".*

Granularity	Query	
Schema	dc, lom	
Property	dc:subject, dc:language, lom:context	
Property Value Range	dc:subject	ccs:sw'engineering
Property Value	lom:context dc:language	"undergrad" "de"

Table 1: contents of the sample query at different granularities

Table 1 shows the values requested in the query at the different granularities; e.g. the query asks for DC and LOM at the schema level, while it requests a lom:context value of "undergrad" at the property value level, etc.

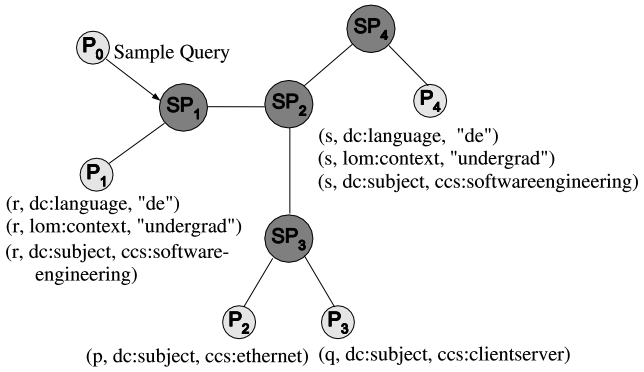

Figure 2: routing example network

In order to further clarify things we consider the scenario shown in figure 2. In this network, various resources are described on different peers, which in turn are attached to super-peers.

Peer P_0 sends the sample query mentioned above to its super-peer SP_1. In our example, this query could be answered by the peers P_1 and P_4, attached to SP_1 and SP_4, respectively. These contain metadata about resources r and s which match the query.

The following paragraphs will explain how the routing indices at the different granularities facilitate routing the query to the right peers.

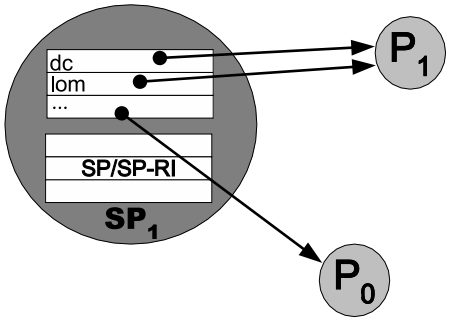

Figure 3: super-peer/peer routing index

Schema Index. We assume that different peers will support different schemas and that these schemas can be uniquely identified (e.g. the dc and lom namespaces are uniquely identified by an URI). The routing index contains the schema identifier as well as the peers supporting this schema. Figure 3 shows a sample of such

an index. Queries are forwarded only to peers which support the schemas used in the query. Super-peer SP_1 will forward the sample query to attached peers which use DC and LOM to annotate resources (peer P_1 in Figure 3). Mediation [5] between different schemas will be supported at the super-peer level, we discuss how it can be supported in subsection 2.4.

Property/Sets of Properties Index. Peers might choose to use only parts of (one or more) schemas, i.e. certain properties, to describe their content. While this is unusual in conventional database systems, it is more often used for data stores using semi-structured data, and very common for RDF-based systems. In this kind of index, super-peers use the properties (uniquely identified by namespace/schema ID plus property name) or sets of properties to describe their peers. Our sample query will be sent to peers using at least dc:subject, dc:language and lom:context (e.g. SP_1 will send the query to P_1, as P_1 contains all of these properties). Sets of properties can be useful to characterize queries (i.e. we might use a "sets-of-properties index" to characterize and route the most common queries).

Property Value Range Index. For properties which contain values from a predefined hierarchical vocabulary we can use an index which specifies taxonomies or part of a taxonomy for properties. This is a common case in Edutella, because in the context of the semantic web quite a few applications use standard vocabularies or ontologies. In our example, peers could be characterized by their possible values in the dc:subject field, and the query would not be forwarded to peers managing "ccs:networks" or "ccs:artificial_intelligence" content (as these sub-hierarchies are disjoint from the ccs:software_engineering sub-hierarchy), and will not be forwarded to peers which use the MeSH vocabulary (because these peers manage medical content).

Note that the subsumption hierarchy in a taxonomy such as ACM CCS can be used to aggregate routing information in order to reduce index size.

Property Value Index. For some properties it may also be advantageous to create value indices to reduce network traffic. This case is identical to a classical database index with the exception that the index entries do not refer to the resource, but the peer providing it. This index contains only properties that are used very often compared to the rest of the data stored at the peers.

In the example, this is used to index string valued properties such as dc:language or lom:context.

2.2 Routing among Super-Peers based on Routing Indices

As with peers, we want to avoid broadcasting queries to all super-peers. To achieve this goal we introduce super-peer/super-peer routing indices to route among the super-peers. These SP/SP indices are essentially extracts and summaries (possibly also approximations thereof) from all local SP/P indices. They contain the same kind of information as SP/P indices, but refer to the (direct) neighbors of a super-peer (as shown in Figure 4). Queries are forwarded to super-peer neighbors based on the SP/SP indices, and sent to connected peers based on the SP/P indices.

Table 2 gives a full example of the SP/SP routing index of SP_2 at the different granularities. For example, SP_2 knows at the schema level that all of its neighbors (SP_1, SP_3, SP_4) use the DC namespace, but only SP_1 and SP_4 contain information described in the LOM schema. Thus, the sample query will not be routed to SP_3, as it requires both DC and LOM.

Granularity	**Index of SP_2**		
Schema	dc		SP_1, SP_3, SP_4
	lom		SP_1, SP_4
Property	dc:subject		SP_1, SP_3, SP_4
	dc:language		SP_1, SP_4
	lom:context		SP_1, SP_4
Property Value	dc:subject	ccs:networks	SP_3
Range	dc:subject	ccs:software-engineering	SP_1, SP_4
Property Value	lom:context	"undergrad"	SP_1, SP_4
	dc:language	"de"	SP_1, SP_4

Table 2: SP/SP index of SP_2 at different granularities

The same applies for the other levels of granularity. A special case is the Property Value Range level; note that ccs:networks is a common super concept of ccs:ethernet and ccs:clientserver in the ACM CCS taxonomy. Making use of the topic hierarchy, the routing index can contain aggregate information like this in order to reduce index size.

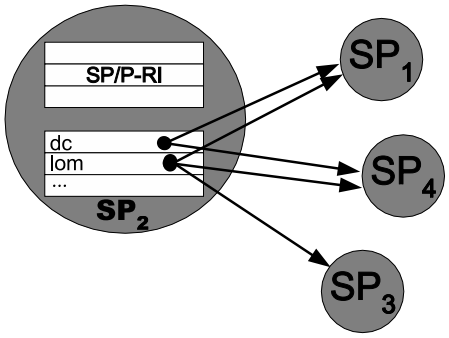

Figure 4: super-peer/super-peer routing index

Update of SP/SP indices is based on the registration (or update) messages from connected peers. We assume for the moment that a peer can connect to an arbitrary super-peer and define the index update procedure as follows: when a new peer registers with a super-peer, it announces the necessary schema (and possibly content) information to the super-peer. The super-peer matches this information against the entries in its SP/P index. If new elements have to be added in order to include the peer into the SP/P index, the super-peer broadcasts an announcement of the new peer to the super-peer network (according to the HyperCuP protocol, so that it reaches each super-peer exactly once). The other super-peers update their SP/SP indices accordingly.

Although such a broadcast is not optimal, it is not too costly either. First, the number of super-peers is much less than the number of all peers. Second, if peers join the super-peer frequently, we can send a summary announcement containing all new elements only in pre-specified intervals instead of sending a separate announcement for each new peer. Third, an announcement is necessary only if the SP/P index changes because of the integration of the new peer. As soon as the super-peer has collected a significant amount of peers (hopefully with the same characteristics, see our discussion on clustering in the next section), these announcements will rather be an exception. Similarly, indices have to be updated when peers disconnect from their super-peers.

The process of super-peers joining the network consists of two parts, namely, taking the appropriate position in the HyperCuP topology and announcing themselves to their neighbors, so that

these can update their SP/SP indices accordingly. The HyperCuP protocol handles the proper positioning and bookkeeping with regard to the topology of the super-peer network. Announcing a new super-peer to its neighbors and updating their SP/SP indices works similarly as the construction of SP/P indices described in section 2.1.

If a super-peer fails, its formerly connected peers must register with another super-peer chosen at random (or find one that contains similar peers as described in the next section). The respective SP/SP indices entries at other super-peers are removed based on the dynamic optimizations described in the next chapter.

Obviously, with an arbitrary distribution of peers to super-peers, the majority of all queries would still have to be sent to most super-peers. Therefore we have to investigate clustering techniques based on peer characteristics, which we will discuss in the next section.

2.3 Dynamic Routing Indices

In the last section we described how queries can be routed using schema-based routing indices. This routing still has the problem that most queries must be broadcast if the peer distribution is arbitrary. In order to avoid broadcasting as much as possible the SP/P and SP/SP routing indices have to be extended with additional frequency information about queries. This allows us to adapt the network topology and peer clustering based on this frequency information, as discussed in the following paragraphs.

Similarity-Based Clustering of Peers.. Clustering in our super-peer network is based on the idea of integrating peers into locations already populated with peers of similar characteristics. In contrast to randomly assigning peers to super-peers this will reduce the amount of messages sent in the network. HyperCuP is a deterministic topology which partitions and sub-partitions the network in a regular way. Connections from a super-peer to its neighbors can be viewed as connections into other partitions and sub-partitions. Forwarding a query to a subset of neighbors therefore results in the distribution of the query within a subgraph.

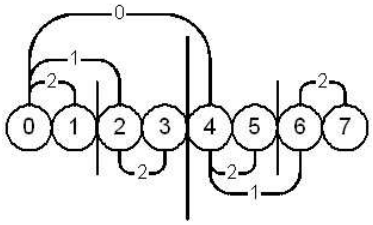

Figure 5: small serialized hypercube, with partitions

As discussed in [14], HyperCuP partitions are connected redundantly using links with different dimensions, and broadcast messages which arrive along a dimension k are forwarded only along links with dimension $i > k$. If we assume for example that the super-peers 4 to 7 in Figure 5 all manage peers with the same characteristics, which are not present at super-peers 0 to 3, a query using these characteristics just has to be forwarded once to the right cluster using a dimension 0 link, and then broadcast to all super-peers in this partition. If a query uses the characteristics of the peers attached to super-peers 0 and 1, we just have to reach this cluster and then distribute the query to the appropriate peers of these two super-peers (based on their SP/P indices).

Obviously we still have to define what kind of similarity measures we want to use for our partitions. In [14] we have discussed how to partition a HyperCuP-based P2P network based on a topic ontology shared by all peers. Another, more dynamic way is to take the characteristics of queries into account when deciding on the clustering parameters. As query characteristics in a large and possibly heterogeneous P2P network cannot be defined in advance, we propose to use frequency counting algorithms on streams such as the ones discussed in [10] to identify the most commonly used schemas, properties and sets of properties in the sent queries. This means that we have to add a frequency property in our SP/SP routing indices, that we focus on including the most frequently used index items in our indices, and that we use these statistics to define our similarity measures responsible for clustering peers to super-peers. Queries not covered by these (possibly incomplete) SP/SP indices can be broadcast in the super-peer HyperCuP network, and then forwarded to the appropriate peers based on the SP/P indices.

The algorithms for estimating the frequencies of specific items discussed in [10], such as sticky sampling and lossy counting, can both be used in this context, as they scan the input stream only once, and do not need much temporary storage to hold the frequency counts. In our scenario, items to be counted could be schemas, schema properties or value entries from a taxonomy, thus clustering the peers according to the schemas or schema properties used, or to the taxonomy entries (or their super-topics) most commonly used in the queries. The algorithm for estimating frequencies of item sets (instead of single items) from [10] is more difficult to use because of its space requirements, though it would be useful to characterize queries as sets of used properties and value ranges and cluster the peers accordingly.

2.4 Mediation between Different Schemas

As outlined in section 2.1 each peer registers with a super-peer using so called advertisements which contain the metadata schema used at the peer. Since schema-based approaches of model correspondences [9, 4] have been successfully used in the context of mediator-based information systems (MBIS) [17] we will apply this approach also to our super-peer networks. The approach of [9] uses rules that describe query capabilities of a peer. In our example in table 2 we assume that each peer provides only one query and one result schema and both schemas are equal. But in some cases a peer will provide many different query schemas and one result schema. In that case both query and result rules will be published to super peers by each peer in our network as an valid advertisement.

Since a peer will only answer queries corresponding to a rule, a super peer will route only relevant queries to its peers. At the moment we distinguish between two relevant roles of peers in our network: *information provider role* and *information consumer role*. Since peers acting as consumers only will not be able to respond to any kind of queries, they need not be considered by the super peers when looking for suitable advertisements.

Typically super peers will collect several advertisements related to their peers. If a super peer receives a query it tries to identify relevant advertisements matching the schema of the query. We distinguish between the following three cases:

1. A query exactly matches an advertisement of only one potential peer.
2. A query exactly matches advertisements of many peers, using one homogeneous schema (or a set of those).
3. A query could be resolved combining results from many peers using heterogeneous schemas.

For case three we have to investigate more sophisticated methods to transform schemas between different peers (i.e. mediation), in-

tegrating different query schemas with each other. In the following we will discuss transformation rules between different schemas, so called *correspondences*, which have already been used in MBIS.

In contrast to MBIS where correspondences are used as rules to translate between global and local schemas, in super peer networks we can typically assume only translations between different local schemas. We will use MBIS-based correspondences as rules to describe such translations, and use property names as arguments in query literals for a concise notation.

In the following example the administrator of the super peer defines a query schema *lectures(lecture:identifier, lecture:language, lecture:subject, lecture:educationalcontext)* which will return documents identified by its URL. First we will define correspondences between attributes of the peer schema and the corresponding attributes the lecture schemas:

1. lectures:Identifier = dc:title
   lectures:language=dc:lang
   lectures:subject=dc:subject

2. lectures:Identifier = lom:general.identifier
   lectures:language=lom:general.language
   lectures:context=lom:educational.context

Using the above mentioned correspondences we can now create views on the peer specific schemas:

1. lecturesViewDC(lectures:Identifier,lectures:language,lectures:subject) ← DC(dc:title, dc:lang, dc:subject)

2. lecturesViewLOM(lectures:Identifier, lectures:language, lectures:context) ← LOM(lom:general.identifier,lom:general.language, lom:educational.context)

Then we can describe, which attributes of the super peers lectures schema could be answered by the local peer schemas:

1. lectures(lectures:identifier,lectures:language,lectures:subject,-) ← lecturesViewDC(lectures:Identifier,lectures:Language,lectures:subject)

2. lectures(lecture:identifier,lecture:language,-, lecture:context) ← (lectures:Identifier,lectures:Language,lectures:context)

Combining all correspondences then results in two main schema correspondences bridging the heterogeneity between the peers P1 and P2.

Peer1:Correspondence1 lectures(lectures:identifier,lectures:language,-, lectures:educationalcontext)
← v(lectures:Identifier,lectures:language,lectures:context)
← LOM(lom:general.identifier,lom:general.language, lom:educational.context)

Peer2:Correspondence2 lectures(lectures:identifier,lectures:language, lectures:subject,-)
← v(lectures:Identifier,lectures:language,lectures:subject)
← DC(dc:title,dc:subject,dc:lang)

A super peer will store relations between correspondences and peers in his indices. When a super peer receives a query *lecture (lecture:identifier, lecture:language, lecture:subject, lecture:educationalcontext)* the super peer identifies P1:Correspondence1 and P2:Correspondence2 as a combination of relevant correspondences that are semantically included in the user query and is able to compute correct results. The query will then be forwarded to the peers Peer 1 and Peer 2. Afterwards, the results have to be collected and combined by the super peer.

We identified *Query Correspondence Assertions (QCA)*[9] and *model correspondences (MOCA)*[4] as a flexible mechanisms to express such correspondences between heterogeneous schemas. The previous paragraphs described the use of QCAs in our network, in the future we will also explore the use of MOCAs for these tasks.

3. IMPLEMENTATION

We are currently in the process of verifying the performance of our protocol and routing mechanisms. To simulate a system based on the protocol, we have implemented our algorithms within the the current Edutella framework. The JXTA Framework from SUN (`www.jxta.org`) is used as the basic P2P infrastructure.

The existing Edutella framework is extended in two areas: The first area consists of support for construction of a network of super-peers based on such topologies as the HyperCup topology discussed in section 1. The second area introduces components for super-peer construction, including services for peer registration and query routing table management.

Figure 6 shows an UML collaboration diagram of a super-peer configured for sending queries to registered provider peers and other super-peers.

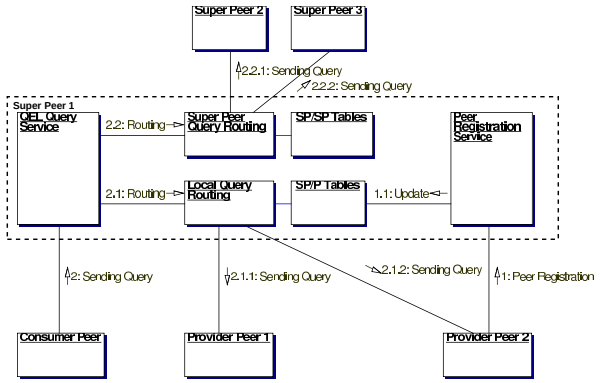

Figure 6: UML Collaboration Diagram

In this example ProviderPeer2 wants to integrate itself into the Edutella network and sends a message to an arbitrary super-peer (collaboration one 1 in figure 6). The super-peers registration service adds the peer schema information included in that message (as described in section 2.1) to the SP/P indices (1.1).

Queries sent from the consumer-peer are received by the super-peer query service (2) and then routed to locally registered provider-peers (2.1) as well as to other super-peers on the super-peer network (2.2). In this example the provider peers 1 and 2 are selected as targets for the query using the SP/P tables, as shown in figure 3. Routing the query into the super-peer network is based on the SP/SP indices, as shown in figure 4 and described in 2.2. We are currently working on the implementation of our SP/SP indices, the current prototype (as of November 15) broadcasts the queries to the other super-peers using the Hypercup [14] protocol only, but the SP/SP functionality will be available shortly. In this example super-peer 2 and 3 are selected as neighboring super-peers (2.2.1, 2.2.2).

Super-peers constructed using this new framework are built from small functional elements we call Services. Communication between them is done sending events to monitoring listeners, according to the Observer design pattern.

A Service could be an endpoint for a JXTA communication pipe, and this way receiving messages from other peers.

```
public interface GroupConfiguration {
  public Iterator getAllServices();
  public Class getMatchingHandler(String service);
  public Iterator getMatchingActors(String service);
  public SPAdvertisement
                   getAdvertisement(String service);
  [...]
}

public abstract class Service {
  private Vector listeners

  public void addEventListener(EventListener svc)
  { ... }
  public void removeEventListener(EventListener svc)
  { ... }
  public void fireEvent(Event event)
  { ... }
  [...]
}
```

In the example shown in figure 6 we use a query service for distributing queries to peers in the network. During initialization of the super-peer, the system iterates over all services of a given GroupConfiguration and instantiates the appropriate Services, e.g. the QueryService class. Additionally the system iterates through all Services connected to the query service by using the `getMatchingActors()` method and registers them to the query service using the `addEventListener()` method.

This way the queries received by the QueryService are passed on to the routing services by sending an Event message to all listeners (fireEvent() method).

In our simulations, we attempt to accurately model traffic patterns that are observable in real-world P2P systems. In recent studies on existing P2P networks [13], important characteristics of both typical traffic patterns and content distribution in P2P networks have been observed. Simulations of P2P algorithms have to be carried out in an environment that closely reflects these findings in order to yield sensible results. Briefly, our simulations exhibit two important characteristics: First, data objects are assigned different popularities which reflect to what degree they are queried for (and possibly replicated) throughout the network. Second, peers are interested only in a subset of the available content on the network.

The latter is shown in [15]: Data objects in a P2P file sharing network can be classified in a number of content categories. In [15], it is also observed that peers are often interested only in data objects from a few content categories. For example, in the domain of educational resources [11], users have a certain affinity towards learning materials related to the course of study they undertake.

It has been observed in [8] that many document storage systems, including the WWW, exhibit Zipf distributions on the popularity of documents. This reflects the fact that some popular documents are very widely copied and held, while most documents are held by far fewer peers: The popularity of the r-th most popular data object in the network is proportional to r^{α}, with α close to uniform. Such distributions fit very well with our frequency counting techniques discussed in 2.3, because it means that we can rely on such techniques for identifying and incorporating the most popular queries / query characteristics in our routing indices.

In our simulations, we combine these characteristics: Peers that join the network pick content categories that they are interested in. A content category c is picked by a peer with probability $\frac{\frac{1}{c}}{\sum_{i=0}^{n}\frac{1}{i}}$ where n is the total number of content categories in the network, i.e., we also assume a Zipf-like popularity distribution on content categories. The set of schemas that a peer uses to mark up its content is chosen in the same way: Some schemas are widely popular in the network, while others are used by only a few peers. Then, each peer is assigned a number of shared data objects: Here, we use a distribution measured on the Gnutella network in [13], where only a few peers share a large part of data objects on the network. As on real-world P2P networks, data objects have different popularities, too - each peer picks data objects from only the set of content categories that it is interested in, and it picks a data object d with a probability proportional to its popularity, which again is governed by a Zipf distribution.

This model reflects observable traffic and content patterns in real-world P2P networks much more closely than, for example, random distribution of documents which so far has been a popular assumption in simulations of P2P networks [2]. In our simulations, popular queries yield responses by many peers, less popular queries can be answered by only a few peers. Also, many peers usually have one or two common schemas that they both use to mark up their data - but on top of that, many peers also have some of their data marked up in more exotic and less common schemas. To emulate the dynamics of a P2P network, we also use distributions measured in [13] to determine the uptime, session duration and bandwidth of peers.

4. CONCLUSION

RDF-based P2P networks have a number of important advantages over previous, more simple P2P networks. In this paper we have discussed RDF-based P2P networks as prime examples of a new kind of P2P networks, schema-based P2P networks. Peers provide and use explicit (possibly heterogeneous) schema descriptions of their content, and are therefore an infrastructure ideally suited for P2P networks consisting of heterogeneous information providers.

We proposed a super-peer topology as a suitable topology for these schema-based P2P networks and discussed, how this additional schema information can be used for routing and clustering in such a network. Super-peer indices exploit the RDF ability to uniquely identify schemas, schema attributes and ontologies, and are used for routing between super-peers and peers as well as within the super-peer backbone network. We further identified and sketched possible algorithms for constructing these indices dynamically and for implementing local transformation rules in these super-peers, and discussed our current implementation as well as simulation environment.

5. REFERENCES

[1] K. Aberer and M. Hauswirth. Semantic gossiping. In *Database and Information Systems Research for Semantic Web and Enterprises, Invitational Workshop*, University of Georgia, Amicalola Falls and State Park, Georgia, April 2002.

[2] L. A. Adamic, R. M. Lukose, A. R. Puniyani, and B. A. Huberman. Search in Power-law Networks. In *Physical Review E, 64 46135*, 2001.

[3] P. A. Bernstein, F. Giunchiglia, A. Kementsietsidis, J. Mylopoulos, L. Serafini, and I. Zaihrayeu. Data management for peer-to-peer computing: A vision. In *Proceedings of the Fifth International Workshop on the Web and Databases*, Madison, Wisconsin, June 2002.

[4] S. Busse. *Model Correspondences in Continuous Engineering of MBIS - doctorial thesis*. Logos Verlag,

September 2002.

[5] S. Chawathe, H. Garcia-Molina, J. Hammer, K. Ireland, Y. Papakonstantinou, J. Ullman, and J. Widom. The TSIMMIS project: Integration of heterogeneous information sources. In *Proceedings of IPSJ Conference*, Tokyo, Japan, October 1994.

[6] A. Crespo and H. Garcia-Molina. Routing indices for peer-to-peer systems. In *Proceedings International Conference on Distributed Computing Systems*, July 2002.

[7] M. Harren, J. M. Hellerstein, R. Huebsch, B. T. Loo, S. Shenker, and I. Stoica. Complex queries in DHT-based peer-to-peer networks. In F. Kaashoek and A. Rowstron, editors, *Proceedings for the 1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*, March 2002.

[8] R. Korfhage. *Information Storage and Retrieval*. John Wiley, New York, 1997.

[9] U. Leser. *Query Planning in Mediator Based Information Systems - doctorial thesis*. TU Berlin, June 2000.

[10] G. S. Manku and R. Motwani. Approximate frequency counts over data streams. In *Proceedings of the 28th International Conference on Very Large Data Bases*, Hong Kong, China, August 2002.

[11] W. Nejdl, B. Wolf, C. Qu, S. Decker, M. Sintek, A. Naeve, M. Nilsson, M. Palmér, and T. Risch. EDUTELLA: a P2P Networking Infrastructure based on RDF. In *WWW 11 Conference Proceedings*, Hawaii, USA, May 2002.

[12] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content addressable network. In *Proceedings of the 2001 Conference on applications, technologies, architectures, and protocols for computer communications*. ACM Press New York, NY, USA, 2001.

[13] S. Saroiu, P. K. Gummadi, and S. D. Gribble. A measurement study of peer-to-peer file sharing systems. In *Proceedings of Multimedia Computing and Networking (MMCN)*, January 2002.

[14] M. Schlosser, M. Sintek, S. Decker, and W. Nejdl. HyperCuP—Hypercubes, Ontologies and Efficient Search on P2P Networks. In *International Workshop on Agents and Peer-to-Peer Computing*, Bologna, Italy, July 2002.

[15] Semantic overlay networks, November 2002. Submitted for publication.

[16] I. Stoica, R. Morris, D. Karger, M. F. Kaashoek, and H. Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 Conference on applications, technologies, architectures, and protocols for computer communications*. ACM Press New York, NY, USA, 2001.

[17] G. Wiederhold. Mediators in the architecture of future information systems. *IEEE Computer*, 25(3):38 – 49, 1992.

[18] B. Yang and H. Garcia-Molina. Designing a super-peer network. `http://dbpubs.stanford.edu:8090/pub/2002-13`, 2002.