Semantic Grouping of Social Networks in P2P Database Settings *

Verena Kantere, Dimitrios Tsoumakos, and Timos Sellis

School of Electr. and Comp. Engineering, National Technical University of Athens,
{vkante, dtsouma, timos}@dbnet.ece.ntua.gr

Abstract. Social network structures map network links to semantic relations between participants in order to assist in efficient resource discovery and information exchange. In this work, we propose a scheme that automates the process of creating schema synopses from semantic clusters of peers which own autonomous relational databases. The resulting mediated schemas can be used as global interfaces for relevant queries. As our experimental evaluations show, this method increases both the quality and the quantity of the retrieved answers and allows for faster discovery of semantic groups by joining peers.

1 Introduction

In the variety of P2P applications that have been proposed, Peer Data Management Systems (PDMSs) (e.g., [6, 19]) hold a leading role in sharing semantically rich information. In a PDMS, each peer is an autonomous source that has a local schema. Sources store and manage their data locally, revealing part of their schemas to the rest of the peers. Due to the lack of global schema, they express and answer queries based on their local schema. Peers also perform local coordination with their *acquaintees*, i.e., their one-hop neighbors in the overlay. During the acquaintance procedure, the two peers exchange information about their local schemas and create mediating mappings semi-automatically [9]. The establishment of an acquaintance implies an agreement for the performance of data coordination between the acquaintees based on the respective schema mapping. However, peers do not have to conform to any kind of data or schema transformation to establish acquaintances with other peers and participate in the system. The common procedure for query processing in such a system is the propagation of the query on paths of bounded depth in the overlay. At each routing step, the query is rewritten to the schema of its new host based on the respective acquaintance mappings. A query may have to be rewritten several times from peer to peer till it reaches nodes that are able to answer it sufficiently in terms of quality but also quantity.

In such systems, in order to enable efficient data sharing between heterogeneous sources, the properties of *social networks* [21] are usually applied: Just as humans direct their queries either to personal acquaintances or other knowledgeable individuals,

* This work has been funded by the project PENED 2003. The project is cofinanced 75% of public expenditure through EC - European Social Fund, 25% of public expenditure through Ministry of Development - General Secretariat of Research and Technology and through private sector, under measure 8.3 of OPERATIONAL PROGRAMME "COMPETITIVENESS" in the 3rd Community Support Programme.

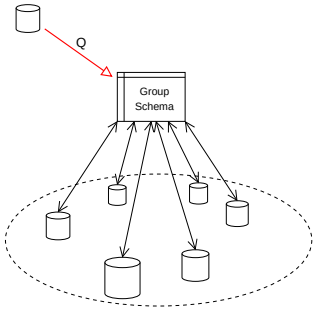

Fig. 1. Query directed towards a group schema

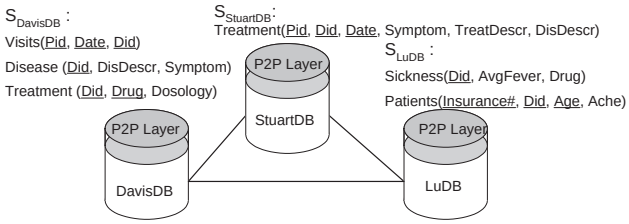

Fig. 2. Part of a P2P system with peer-databases from the health environment

peers try to identify other participants in the overlay with interests that match theirs. Similar to human social networks, social networking services such as MySpace [15], Orkut [17], etc, form virtual communities, with each participant setting her own characteristics and interests. Their goal is to allow members to form relationships through communication with other members and sharing of common interests. Extending this paradigm, computing social networks consist of a mesh of interconnected nodes (peers). Initially, each of the nodes is connected to a random subset of peers. Gradually, nodes get acquainted with each other, with the new connections indicating semantic proximity. Although not explicitly stated, there has been considerable work to apply the principle of semantic grouping and routing in order to improve performance in distributed systems (e.g., [1, 2, 10, 16, 18], etc).

Assuming a social network organization in a PDMS, an interesting question is how to automatically create a synopsis of the common interests of a group of semantically related nodes. This will be a mediating schema representative of the group along with its mappings with the local databases. Queries can then be expressed on this mediated schema (see Figure 1). This functionality is desirable for multiple reasons: First, it allows queries to be directed to a single, authoritative schema. Second, it actively expedites the acquaintance between semantically related peers. Finally, it minimizes human involvement in the process of creating/updating the group schema. Until now, nodes have been organized by means of a human-guided process (usually by one or more administrators and application experts) into groups of peers that store semantically related data. The administrator, using schema matching tools as well as domain knowledge, initiates and maintains these synopses. This approach requires manual work, extensive peer coordination and repetition of this process each time the group changes.

As a motivating example, envision a P2P system where the participating peers are databases of private doctors of various specialties, diagnostic laboratories and databases of hospitals. Figure 2 depicts a small part of this system, where the peer databases (or else, pDBMSs) are: DavisDB - the database of the private doctor Dr. Davis, LuDB - the database of pediatrist Dr Lu and StuartDB - the database of the pharmacist, Mr Stuart. A P2P layer, responsible for all data exchange of a peer with its acquaintees, sits on top of each database. Among others, the P2P layer is responsible for the creation and

maintenance of mappings of local schemas during the establishment of acquaintances towards the line of [9]. Moreover, each peer owns a query rewriting and a query-schema matching mechanism. The schemas of the databases are shown in Figure 2.

We would like to automatically produce a merged schema for all three peers of our example, semantically relevant to their local schemas. Such a merged schema could be the following:

Disease/Sickness(Did, DisDescr, Symptom, Drug)
Visits/Patients(Pid/Insurance#, Did, Date, Age, Ache)
Treatment(Did,Drug, Dosology)

Obviously, in the merged schema we would like alternative names for relations or attributes (separated by '/' above). We would also like the merged schema to contain relations or attributes according to their frequency in the set of local schemas. For example, the attribute *Patients.AvgFever* is not present, possibly because the respective concept is not considered to be frequent in the set of local schemas.

In this paper, we describe a mechanism that operates on a semantically clustered PDMS and automatically creates relational schemas that are representative of the existing clusters. Given the semantic neighborhoods, our system can initiate the creation of a mediating schema $\mathcal{S}_{\mathcal{G}}$ that summarizes the semantics of the participating database schemas. It is created by the gradual merging of peer schemas along the path followed by the process. We call *interest* or *semantic groups* the semantic clusters that exist in social networks operating on PDMSs; moreover, we call *group schema* the inferred schema of the group $\mathcal{S}_{\mathcal{G}}$. $\mathcal{S}_{\mathcal{G}}$ holds mappings with each of the peers involved in its creation and functions as a point of contact for all incoming queries, whether from inside or outside the semantic neighborhood. Thus, requesters of information need only maintain mappings and evaluate queries against one schema, instead of multiple ones. Our experimental evaluation shows that our group creation process increases both the accuracy and the number of answers compared to individually propagating and answering queries in an unstructured PDMS.

2 Interest Group Creation

Our goal in creating a group schema is to represent the semantic clusters in a social network using a distributed process that iteratively merges local schemas into the final group schema that preserves their most frequent semantics.

In the following, we assume a PDMS with a social-network organization of peers, i.e., semantically relevant pDBMSs are acquainted or close in the overlay. This can be achieved either manually or using one of the proposed schemes (e.g., [12, 16]). Finally, we also assume that peer mappings between acquaintees are of the widely-known GAV/LAV/GLAV form [6, 13] and peer schemas are relational, (i.e., the only internal mappings are foreign key constraints). Moreover, peers do not carry semantic information about their schemas and mappings.

2.1 Group Inference

In this section, we describe the process through which a group schema emerges from a set of clustered nodes in our system. The group-creation procedure (or group *inference*) comprises the following steps:

- Initialization: Who and when initiates the group schema inference
- Propagation: How does the process advance among peers of the same group
- Termination and Refinement: When is the process over/reiterated

Initialization: The nature of our application requires that the group inference is performed in a distributed manner, without global coordination. Peers should be able to start the process that creates the respective schema with minimum message exchange. In our system, each member of the social group is eligible to initiate the inference process. Nevertheless, such groups may consist of numerous participants resulting in very frequent collisions among competing initiators. Hence, we only allow *active* members to become the initiators of the process. This is enforced by a system-wide parameter that defines the minimum number of queries posed in the most recent time frame. Intuitively, more active peers have a better knowledge of the social network and the schemas of the other participants through the answers they receive.

The initiator's local schema becomes a point-of-reference regarding the inferred one. Thus, the peer schemas considered for the formation of the group schema should not differ semantically a lot from the schema of the initiator. Specifically, we require that the participating local schemas should be at least t-similar to the initiator's schema: t is a parameter that mainly determines how specialized (only peers very similar to the initiator considered) or general (a broad collection of peers participate in the process) the inferred schema will be. The initiator peer is called the *originator* of the group, its schema is the *origin* of the group schema and the maximum similarity distance between the origin and the peer schemas that participate in the group schema inference is the *semantic radius* of the group. The following function calculates the *directed* semantic similarity, *SS*, of two relational schemas:

$$SS(S,T) = \frac{\sum_i \sum_j w_{ij} Mapped_T(SR_{ij})}{\sum_i \sum_j w_{ij} SR_{ij}}$$

In the above function, S is the source schema and T is the target schema. *SS* calculates the portion of S's attributes (*SR*) that are mapped on T, with the indices i, j referring to the j^{th} attribute of the i^{th} relation. Also, $w_{ij} > 1$ for attributes that belong to relation keys and $w_{ij} = 1$ otherwise. Obviously, $SS(S,T) \neq SS(T,S)$ in general. *SS* achieves to measure semantic similarity because it takes into consideration the mapping of *concepts* beyond their structural interpretations on the schema level. In our setting we define a distinct concept of a schema S to be each element $R.A$, where A is an attribute of relation R of schema S [1]. Moreover, since *SS* ignores the schema structure, it is very easily calculated.

Propagation: The initiator I (with schema S_I) of the inference process initializes the group schema to its own and creates a stack $ST(I)$ with its acquaintees that are part of the cluster. Specifically, $ST(I) = \{A_1, A_2, ..., A_m\}$ is an ordered set of elements $A_j = \{P_j, SS(S_I, S_{P_j})\}$, where P_j is a peer with schema S_{P_j}. Elements A_j refer to the I's most similar acquaintees: $SS(S_I, S_{P_j}) \geq t$, $j = 1, .., m$ and $SS(S_I, S_{P_j}) \geq SS(S_I, S_{P_{j+1}})$, $j = 1, .., m-1$. The initiator propagates the inference procedure to the first peer on the stack. The latter is supposed to merge its own schema with the group schema it receives according to the merging procedure described in the section 2.2. Every peer P

[1] For more details on concepts, see [12]

on the network path of the inference process determines its acquaintees P_j for which $SS(S_I, S_{P_j}) \geq t$, adds the respective pair P_j, $SS(S_I, S_{P_j})$, to $ST(I)$ and orders it. Any peer P on the inference process path calculates $SS(S_I, S_{P_j})$ indirectly, as the product: $SS(S_I, S_P) \cdot SS(S'_P, S_{P_j})$, where S'_P is the part of S_P mapped on S_I. Essentially, $SS(S_I, S_P)$ aims to measure how much of the semantics of S_I can be found on schema S_P, independently of other semantics that the latter captures. The only way to measure this (without automatic matching) is through the chain of mappings of S_I all the way to S_P. Thus, the value of $SS(S_I, S_P)$ depends on the path followed by the inference process and fails to consider concepts that exist both in S_I and S_P but not in the schemas of intermediate nodes.

In order for this formula to produce a satisfactory result, the existing clustering in the social network should assure that the similarity between local schemas decreases with the hop-distance of the respective peers in the overlay. Therefore, schemas that are considered later in the process will have lower similarity than previously considered ones. Moreover, if a peer P already in $ST(I)$ is considered for addition, the entry with the highest $SS(S_I, S_P)$ value is kept.

Even though the participation or not of peers in the inference process is judged by a part of their schemas, their whole schema contributes to the inferred group schema (see subsection 2.2). Intuitively, the goal of the inference process is to produce a schema that represents semantics encapsulated in the cluster. In order to determine the cluster's semantic borders we use the semantics of the initiator as reference. This way, the process is safe from producing a schema too broad or distorted from the interests of the initiator.

Termination: As aforementioned, the group inference procedure ends when the stack of participating peers becomes empty. However, if too many peers own schemas very similar to the originator's schema or the similarity threshold t is too small (i.e., the semantic radius of the inferred group is big), then it may be the case that the stack is provided at each step with a lot of new entries. Thus, the inference procedure is prolonged taking into account a big number of peers. After a certain number of iterations, there is usually no point of considering more peer schemas in the inference procedure, because they do not alter the schema significantly. In order to expedite the inference process and reduce the exchanged messages, we add a limit to the maximum number of encountered peer schemas, *MaxP*, as a termination condition. *MaxP* is not a TTL condition, since successive hops are not always on the same path; *MaxP* refers to the total number of participating peers and not just the peers on one path.

2.2 Schema Merging Algorithm

The goal of the merging procedure is to produce a schema that represents the semantics of the majority of the peers that belong to the respective cluster. This is achieved gradually by merging the schemas of peers on consecutive steps of the path that the merging procedure follows. We need a merging procedure that preserves the most popular concepts of the respective peer schemas and produces a schema representative of almost all the source schemas. Thus, we require a merging procedure that performs high compression before throwing away schema elements (i.e., relations or attributes). Finally, we require that merging is only based on available information on the peers, i.e., it solely exploits the peer schemas and the peer mappings. Each mapping is considered

to be a set of 1-1 correspondences between attributes that hold with an optional set of value and join constraints on some attributes (see [12] for details).

The schema merging procedure is designed with respect to the following dictations:

D1 Fewer relations with more attributes are preferred to more relations with fewer attributes
D2 The semantic relevance of two relations is proportional to the number of correspondences between their sets of attributes
D3 If the keys of two relations are mapped thoroughly, both relations are considered to be projections of the same relation with the same key
D4 The key of a merged relation consists of the keys of both relations that are merged
D5 If two attributes are merged and at least one of them is a key, then the merged attribute is part of the key of the merged relation
D6 Correspondences that involve the same attribute imply that all involved attributes are semantically equivalent
D7 Correspondences that are based on any value constraints are considered valid only under certain conditions and never produce merged attributes.
D8 There are two pre-specified constants that represent the maximum number of relations that the schema of the interest group is allowed to have and the maximum number of attributes per relation

Briefly, the schema merging procedure produces the interest group schema but also a set of internal mappings and a dictionary. The internal mappings are the peer mappings that were not consumed in the successive schema merges. These hold additional syntactic and implicit semantic information for the group schema elements; thus, they can be very helpful to peers that would like to join the group and create mappings to their local schema. Moreover, this set of mappings includes all mappings with value constraints met during the merging procedure. Such mappings cannot be consumed: the involved relations/attributes cannot be merged, since they are mapped under certain conditions (the value constraints). Furthermore, the merged schema has alternative keywords for the same element that result from the merged mapping correspondences. These alternatives are entered in the dictionary that accompanies the group schema. The dictionary is then used to identify semantic similarity between a group and a new node and also assist in the creation of mappings if so desired.

The algorithm first merges relations that share the same key and then those that do not. In the latter case, priority is given to relations that share most of their attributes. Additional criteria in order to break ties can be based on whether the corresponding attributes are parts of the relation keys, or whether unmapped attributes are parts of the relation keys. Nevertheless, refining the algorithm based on additional criteria is future work. At the end of the schema merging procedure, i.e., when all relevant peer schemas have been merged, relations and relation attributes that have been rarely met during the procedure can be dropped. In order to do this, we need to keep a counter for each of them during the merging. For a thorough analysis and presentation of the merging algorithm the reader is referred to [11]. We present a simple merging example.

Example: Assume the pDBMSs of the motivating example in Section 1. The schemas of DavisDB and LuDB are presented in Figure 2; assume that the databases have the following mapping:

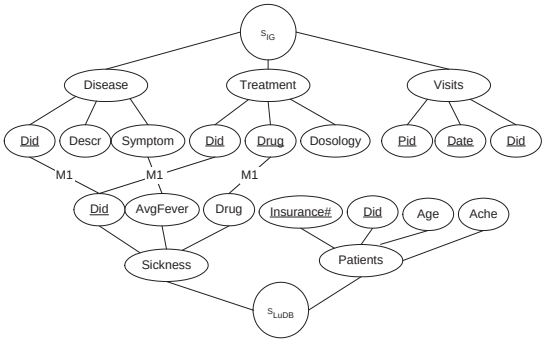

Fig. 3. S_{IG} is initialized to $S_{DavisDB}$ and there is mapping $M1$ between S_{IG} and S_{LuDB}

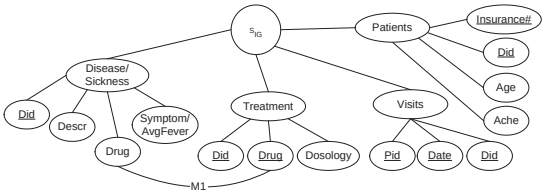

Fig. 4. Relations Disease and Sickness of Figure 3 are merged

$M1_{LuDB\_DavisDB}$:

```
Disease (Did, _, Symptom), Treatment (Did, Drug, _):-
Sickness(Did, AvgFever, Drug),
```

where the correspondences Symptom = AvgFever and Disease = Sickness are implied and '_' is introduced for attributes that are not needed.

As shown in Figure 3, there are three correspondences that are encapsulated in mapping $M1$. We assume that the peer of Dr Davis initializes the schema merge. Thus, the group schema S_{IG} is initialized to $S_{DavisDB}$. First, all relations of S_{LuDB} are added to S_{IG}. Relations *Disease* and *Sickness* are merged in one (Figure 4), since they share the same key; thus, attributes *Symptom* and *AvgFever* are merged. The correspondence $Disease/Sickness.Drug = Treatment.Drug$ is kept as an internal one. Also, the dictionary D is enriched with correspondences $Disease = Sickness$ and $Symptom = AvgFever$; actually the schema keeps one name for each relation or attribute from the alternative ones. At the end of the schema merging procedure we propose that the schema keeps for relation and attribute names the most common ones encountered during the procedure. Relations *Disease*/*Sickness* and *Treatment* are merged (Figure 5), since they are the only ones related with a mapping. Now there is one attribute named 'Drug' and it is part of the relation key, even though just one of the attributes that where merged was a key. Additional iterations can merge relations based on foreign key constraints, since no other internal mappings exist.

3 Performance Evaluation

To evaluate the performance of the proposed group inference procedure, we use a message-level simulator that implements it over an unstructured overlay of semanti-

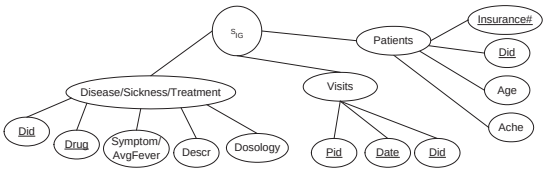

Fig. 5. Relations Disease/Sickness and Treatment of Figure 4 are merged

cally clustered nodes. The clustering is performed using the *GrouPeer* system [12]. In GrouPeer, peers decide to add new (and abolish old) one-hop neighbors in the overlay (acquaintees) according to the accuracy of the answers they receive from remote peers. This is measured using a function that tries to capture the semantic similarity between rewritten versions of a query. Specifically, requesters (i.e., peers that pose queries) accumulate correct and erroneous mappings with remote peers through a learning procedure. Based on these mappings, they decide to become acquainted with peers that store information similar to their interests. The result is an effective semantic clustering of the overlay, where the accuracy of query rewritings and answers is a lot higher compared to the unclustered overlay (for details see [12]).

We compare the query evaluation performed by GrouPeer with the evaluation that utilizes the inferred groups on the overlay. When the first group is created, we direct relevant queries to the inferred schema. The basic performance metrics are the average *accuracy* of answers to the original queries (i.e., the similarity of the rewritten query that is answered over the original one), as well as the number of nodes that provide an answer. Similarity is calculated by a formula presented in [12] that identifies erroneous or not-preserved correspondences in mappings, which degrade the complete and perfect rewriting. To identify the gains of our grouping approach, we present the percentile increase/decrease in accuracy and number of answers compared to GrouPeer's clustering as these are measured on the *first* created group. Participants of the group hold mappings with the group schema; thus, when the query is rewritten to the group schema, the successive rewritings through the chain of mappings are avoided. Non-members create mappings with relevant group schemas.

We present results for 1,000-node random graphs (an adequate number of participants regarding our motivating application) with average node degrees around 4, created by the *BRITE* [14] topology generator. Results are averaged over 20 graphs of the same type and size, with multiple runs in each. Results using power-law topologies constructed by Inet-3.0 [8] with the same number of peers are qualitatively similar.

For the schemas stored at each node, we use an initial schema whose relations and attributes are uniformly distributed at the nodes. The initial schema comprises of 5 relations and 33 attributes. Seven attributes are keys with a total of 11 correspondences between them. Each peer stores 10 table columns (attributes) on average. Queries are formed on a single or multiple tables if applicable (join queries). The maximum size of the inferred schema is always in the order of the size of the initial schema used to produce the local ones during start-up.

First, we vary the maximum group size limit, *MaxP*, as well as the minimum similarity of participating peers to the initiator node, t. Figures 6 and 7 show the obtained results for 100 requesters and maximum 100 queries each. As t increases, the group

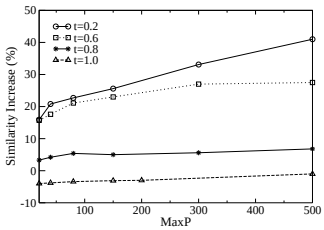

Fig. 6. % Increase in answer similarity over variable MaxP and t

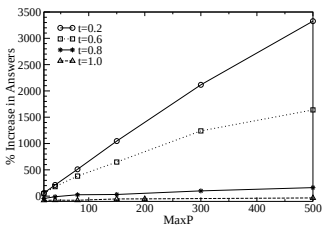

Fig. 7. % Increase in number of answers over variable MaxP and t

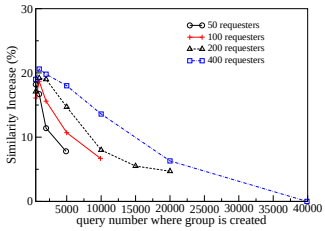

Fig. 8. % Increase in answer similarity over variable group creation time

becomes more specialized and less general. In contrast, small similarity values produce groups too general that incorporate many concepts foreign to the initiator. This results in *specialized* groups (i.e., high value of t) that receive fewer queries, while more "general" ones receive more but cannot answer them all satisfactorily. As the graphs show, there exists a point where grouping ceases to increase its relative gains to clustering.

Both metrics increase as *MaxP* increases. This is reasonable since more nodes can participate and produce results. Very specialized grouping causes significantly less populated groups, which in turn affects the number of returned answers. As groups get more general (around $t = 0.6$), an improvement of 13-23% in accuracy is achieved, while the gains in replies are 40-900%. As t decreases, the gains in accuracy decrease but more results are generated. These curves show that a t value of around 0.65 with the group initiator and $MaxP = 80$ achieve good results without too much generalization. These will be our default values for the rest of this discussion.

Next, we try to determine the quality of the created group based on the quality of the semantic clustering. Figure 8 show the percentile improvement in the similarity of answers when the first group is created at various points (i.e., number of queries) in the clustering process. The results show a decrease in the relative gains in accuracy which is due to the improvement of clustering with time. What is important is that groups that are allowed to be created as soon as possible (which would be the common case) show about 20% more accurate answers and return about three times more results compared to clustering, even though the inference procedure is performed on a less optimally clustered overlay. The clustering process is expedited with more active requesters, which suits the purposes of grouping. An extended experimental study is presented in [11].

4 Related Work

There exist several interesting research efforts that have discussed about semantics and semantic clustering of peers. The work in [3] is one of the first to consider semantics in P2P systems and suggest the construction of semantic overlay networks, i.e., SONs. Various other researchers have attempted to go beyond the a priori static formulation of SONs: the work in [18] suggests the dynamic construction of the interest-based shortcuts in order for peers to route queries to nodes that are more likely to answer them. Inspired by this work, the authors of [20] and [7] exploit implicit approaches for discovering semantic proximity based on the history of query answering and the least recently

used nodes. In the same spirit, the work in [4] presents preliminary results about the clustering of the workload on the popular e-Donkey and Kazaa systems.

Finally, Bibster [5] is a project that exploits ontologies in order to enable P2P sharing of bibliographic data. Ontologies are used for importing data, formulating and routing queries and processing answers. Peers advertise their expertise and learn through ontologies about peers with similar data and interests.

5 Summary

In this paper we have described a method to automatically create schemas in order to characterize semantic clusters in PDMSs. Our scheme operates on clustered unstructured P2P overlays. By iteratively merging relevant peer schemas and maintaining only the most frequent common characteristics, we provide a schema representative of the cluster. Group schemas can be used in order to increase both query performance and the volume of returned data. Our experimental evaluations confirm these observations in a detailed comparison with the GrouPeer system.

References

[1] K. Aberer, P. Cudre-Mauroux, M. Hauswirth, and T. Van Pelt. Gridvine:Building internet-scale semantic overlay networks. In *International Semantic Web Conference*, 2004.

[2] E. Cohen, A. Fiat, and H. Kaplan. Associative search in peer to peer networks: Harnessing latent semantics. In *INFOCOM*, 2003.

[3] A. Crespo and H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. In *Technical Report*, 2003.

[4] F. Le Fessant, S. Handurukande, A.-M. Kermarrec, and L. Massoulie. Clustering in Peer-to-Peer File Sharing Work-Loads. In *IPTPS*, 2004.

[5] P. Haase, B. Schnizler, J. Broekstra, M. Ehrig, F. van Harmelen, M. Menken, P. Mika, M. Plechawski, P. Pyszlak, R. Siebes, S. Staab, and C. Tempich. Bibster - a semantics-based bibliographic peer-to-peer system. In *Journal of Web Semantics*, 2005.

[6] A. Halevy, Z. Ives, D. Suciu, and I. Tatarinov. Schema Mediation in Peer Data Management Systems. In *ICDE*, 2003.

[7] S. Handurukande, A.-M. Kermarrec, F. Le Fessant, and L. Massoulie. Exploiting Semantic Clustering in the eDonkey P2P Network. In *ACM SIGOPS*, 2004.

[8] C. Jin, Q. Chen, and S. Jamin. Inet: Internet Topology Generator. Technical Report CSE-TR443-00, Department of EECS, University of Michigan, 2000.

[9] V. Kantere, I. Kiringa, J. Mylopoulos, A. Kementsientidis, and M. Arenas. Coordinating P2P Databases Using ECA Rules. In *DBISP2P*, 2003.

[10] V. Kantere, D. Tsoumakos, and N. Roussopoulos. Querying Structured Data in an Unstructured P2P System. In *WIDM*, 2004.

[11] V. Kantere, D. Tsoumakos, and T. Sellis. Semantic Grouping of Social Networks in P2P Database Settings. Technical Report TR-2007-2, National Technical Un. of Athens, 2007. http://www.dbnet.ece.ntua.gr/pubs/uploads/TR-2007-2.

[12] V. Kantere, D. Tsoumakos, T. Sellis, and N. Roussopoulos. GrouPeer: Dynamic Clustering of P2P Databases. Technical Report TR-2006-4, National Technical Un. of Athens, 2006. http://www.dbnet.ece.ntua.gr/pubs/uploads/TR-2006-4.

[13] A. Y. Levy. Answering Queries Using Views: A Survey. In *VLDB Journal*, 2001.

[14] A. Medina, A. Lakhina, I. Matta, and J. Byers. BRITE: An Approach to Universal Topology Generation. In *MASCOTS*, 2001.

[15] MySpace website: http://www.myspace.com.

[16] B. Ooi, Y. Shu, K.L. Tan, and A.Y. Zhou. PeerDB: A P2P-based System for Distributed Data Sharing. In *ICDE*, 2003.

[17] Orkut website: http://www.orkut.com.

[18] K. Sripanidkulchai, B. Maggs, and H. Zhang. Efficient Content Location Using Interest-Based Locality in Peer-to-Peer Systems. In *INFOCOM*, 2003.

[19] I. Tatarinov and A.Halevy. Efficient Query Reformulation in Peer-Data Management Systems. In *SIGMOD*, 2004.

[20] S. Voulgaris, A.-M. Kermarrec, L. Massoulie, and M. van Steen. Exploiting Semantic Proximity in Peer-to-Peer Content Searching. In *FTDCS*, 2004.

[21] F. Wang, Y. Moreno, and Y. Sun. Structure of peer-to-peer social networks. *Physical Review E*, 73, 2006.