GrouPeer: A System for Clustering PDMSs

Verena Kantere[*]
Cyprus University of Technology
verena.kantere@cut.ac.cy

Dimos Bousounis
Swiss Federal Institute Of Technology Zurich
dbousoun@student.ethz.ch

Timos Sellis
National Technical University of Athens
timos@dblab.ece.ntua.gr

ABSTRACT

Sharing structured data in a PDMS is hard due to schema heterogeneity and peer autonomy. To overcome heterogeneity, peer databases employ mappings that partially match local information to that of their direct neighbors. Traditionally, a query is successively rewritten along the propagation path on each peer. This results in gradual query degradation and the inability to retrieve data pertinent to the original version, even from peers that store such data. This demonstration presents *GrouPeer*, a system that overcomes the query degradation problem and enables the dynamic clustering of the overlay according to the semantics of the peer data, utilizing normal query traffic. Peers are provided with a methodology that allows them to choose which rewritten version of a query to answer and discover remote information-rich sources. The demonstration illustrates the functionalities in the clustering mechanism of *GrouPeer*: approximate query rewriting, query similarity methodology, construction of new mappings, iterative learning process, employment of automatic schema matching, and proves the capability of the system to perform gradual semantic clustering and enable high quality answers to peer queries.

1. INTRODUCTION

In an unstructured Peer-to-Peer data management system (hereafter PDMS) that shares structured data, the major obstacle for acquiring high quality answers to peer queries is the heterogeneity of peer database schemas in combination with peer autonomy. Answers to peer queries that are propagated in the overlay are enabled by schema mappings between pairs of directly acquainted peers [4]. Mappings are employed in order to rewrite and answer an incoming query on the local peer database. Yet, matching schema information between acquainted peers is often partial and can be poor in case of great schema dissimilarity. In large scale PDMSs it is expected that joining peers are acquainted in a random way; frequently, they are far away (in terms of overlay hops) from peers with relevant data and, thus, interests. Such overlay connectivity situations may permanently condemn various peers to ignorance of other information-rich peers because of enforced reformulation of queries on each node of the propagation path. The following is an example that exhibits this problem through the illustration of an extreme unfortunate case of overlay connectivity.

Example: Assume a three-peer overlay with the nodes successively acquainted in the following order: P_1, P_2, P_3 with schemas S_1, S_2, S_3, respectively. Peers P_1 and P_3 have the same schema, i.e. $S_1 \equiv S_3$, since they use the same commercial database product. Also S_2 is substantially different than S_1, S_3, such that a lot of relations and attributes of the latter cannot be mapped on S_2. Queries by P_1 reach P_3 after they have been locally rewritten on S_2. It is certain that most such queries cannot be completely rewritten on S_2 and thus a degraded version reaches P_3, even though the latter has the missing requested information. These queries would be completely rewritten and answered, without any loss, if P_1 and P_3 were directly acquainted.

The work in [9] proposes *GrouPeer*, a framework that provides a PDMS with accurate answers to locally posed queries in the absence of a global schema. *GrouPeer* presents a procedure that supports the evasion of successive rewritings on every peer of a query's propagation path, instead of, sometimes hopelessly, refining query reformulation. This methodology enables peers to discover others with similar interests and schemas, that cannot be tracked otherwise. Pairs of remote peers that exchange queries and answers learn gradually about the schema of the other party. Learning is performed through query answering and evaluation and it is formed through the creation of mappings between the peer schemas. These mappings encapsulate the common peer interest, since they refer to the vital schema parts on which they express and answer queries. If the peers decide to become acquainted, these mappings are already a language for their communication and alleviate the administrator's load for manual creation of mappings for the new acquaintance. This methodology leads to the gradual clustering of the PDMS in groups with common interests.

In this demonstration we present a full-fledged implementation of the *GrouPeer* framework. Our goal is to exhibit the effectiveness of the clustering process in order to overcome the query degradation and, in general, the heterogeneity problem in PDMSs. The demonstration presents the complex and successful collaboration of the several partial mechanisms implementing the various system functionalities such that remote schema learning becomes feasible.

More information about *GrouPeer* can be found in the site [3].

2. THE CLUSTERING PROCESS

We summarize the clustering process that enables remote peers with similar interests and schemas to mutually discover each other, even if they are hidden in the query propagation paths by other peers with dissimilar interests.

[*]This research was conducted while the first and second author were working at the National Technical University of Athens.

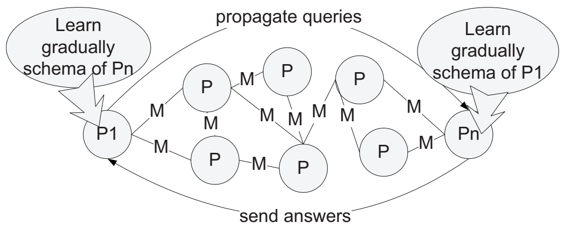


Figure 1: Learning about a remote peer.

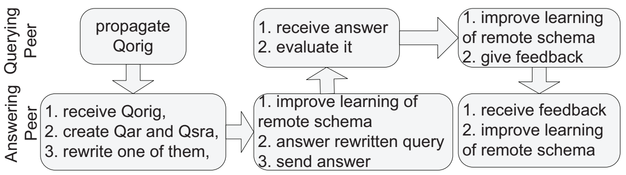


Figure 2: One iteration of the learning procedure.

2.1 Learning about Remote Peers

Peers propagate their queries in the overlay, which are rewritten on each peer database according to the mappings that the latter holds with the previous database in the propagation path. The query that reaches a remote peer has lost parts of it, due to the mismatches in mapped schema information in consecutive peers in the propagation path. The two remote peers try to learn the schema of the other part, during the propagations of queries and respective answers to each other. Figure 1 depicts this overlay situation. The peer that receives a query tries to retrieve any query attributes that are lost in the successive rewriting of the query along the propagation path. The answer is sent back to the peer that posed the query; the latter evaluates the answer and improves its knowledge about the schema of the remote answering peer. Moreover, it sends this evaluation to the answering peer, so that the latter can improve its own knowledge about the schema of the remote query-posing peer. Figure 2 shows the flow of one iteration of the learning procedure between two remote peers. The peer that posed the query accumulates the answers it receives; each time it receives a new answer from a specific peer, it computes the current overall respective similarity of answers by this peer. If this similarity exceeds a user-tunable threshold, the peer may decide to ask this peer to be its acquaintee. The mappings that are formed during the query position-answering between them are used as an initial set of communication mappings. Overall, the proposed methodology of making new acquaintances in the overlay leads to the restructuring and, moreover, the gradual clustering of the P2P system in groups with common interests.

2.2 Approximate Query Answering

In order to achieve the discovery of remote semantically related peers, the key idea of our method is to propagate not only the successively rewritten version along the query path but also the original one. In this way, the peers receiving this pair of query versions can individually decide which one to answer. Peers are equipped with an approximate query rewriting mechanism and an automatic schema-matching tool. The rewriting mechanism is used in order to rewrite queries expressed on schemas of acquaintees based on existing mappings, or schemas of remote peers with revealing mappings. In the latter case, the automatic schema-matching tool is used in order to create concept correspondences encapsulated in queries expressed on schemas for which mappings are being built.

Successive query rewriting produces query versions that deviate structurally and semantically from the original query. If the chain

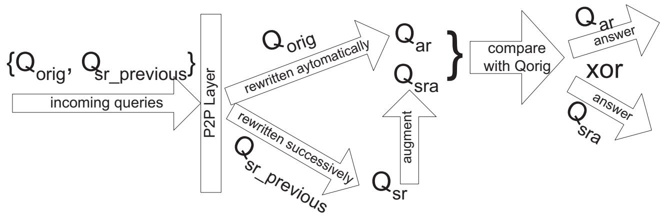


Figure 3: Representation of the query answering procedure.

of peer mappings used for the rewriting is poor in information relevant to the query (i.e., query parts cannot be accurately reformulated), this can result in fast degradation within a few hops. Query parts that cannot be translated through existing mappings are eliminated in the rewritten version.Even if the following nodes on the query path encapsulate the eliminated concepts in their schemas, they still cannot contribute them to the rewriting, since the version they receive does not include them. Our goal is to keep the eliminated concepts aside and try to match them in follow-up schemas.

Overall, an initiated query Q_{orig} is propagated along several query paths. On each node, the query is rewritten through mappings with the previous node to Q_{sr}, which is augmented with automatically rewritten query parts to Q_{sra}. Also, Q_{orig} is automatically rewritten from scratch to Q_{ar}. The answering node compares the two rewritten versions with the original one, using a special similarity function and answers the version it deems most similar to it. Figure 3 summarizes the main part of the query answering procedure on a peer. The query initiator evaluates the received answer and sends its feedback to the answering peer.

The approximate successive rewriting of Q_{orig} to Q_{sr} and the approximate automatic rewriting of Q_{orig} to Q_{ar} are performed using a novel technique [7] that adapts a state of the art algorithm for traditional query rewriting to the needs of information in PDMSs. The proposed technique achieves to find the most suitable approximate query version to be answered in absence of any data value information and in a very efficient way.

2.3 Mapping Creation and Evolution

Through the query exchange and evaluation, two remote peers keep record of bad and good rewritings of each other's schema elements, and, gradually, they build mappings. The mappings between remote peer databases are built employing a novel technique described in [5, 6]. Briefly, this is a technique that aims at discovering GAV and LAV mappings in a semi-automatic manner as relational schemas are revealed. The technique is schema-centric instead of mapping-centric, meaning that the mapping accuracy is adapted to the incremental newly disclosed semantics, as the schema is revealed. The technique not only provides possible mappings, but rank-orders them, so that the user is presented with the mapping that is expected to be more accurate. The mapping space is efficiently searched so that the more accurate mappings are encountered first. The technique is enhanced with a simple interface that enables the user to lightly guide the schema mapping through coarsely expressing her opinion on the mapping structure. These easy-to-make estimations are valuable to the mechanism since they are used as mapping experience. The latter facilitates future mapping adaptation to new schema semantics, in that the same mistakes are avoided but, also, search in the mapping space becomes more efficient. The technqiue produces mappings with value conditions, exploiting query traffic between the matched remote schemas.

Based on the evolving mappings, the two peers can decide that they have common interests ask each other to become acquaintees. After that, new acquaintees can base their communication on already created mappings.

3. THE GROUPEER SYSTEM

The core functionalities of the *GrouPeer* are:

- Communication management: refers to all tasks involved in the communication with the acquaintees, the user and the local database.
- Approximate query rewriting: enables partial successive rewriting of a query based on the available mappings.
- Associative automatic schema discovery: refers to query increment with automatically discovered schema elements and matching with query rewriting.
- Mapping construction: refers to the formulation of schema learning into mappings between the remote peers.
- Clustering algorithm: realizes the overall gradual learning process between remote peers and decides if and when clustering should be requested/performed.
- Query similarity methodology: qualifies and quantifies the similarity of rewritten query versions.

GrouPeer is combines various mechanisms that implement the functionalities presented earlier. We describe the modules that implement these mechanisms and their interactions (see Figure 4).

Peer Database: is the local database on top of which the P2P layer sits, which implements the *GrouPeer* functionality. We have employed MySQL [1].

Communication manager: enables communication with the P2P system (i.e., acquaintees) and the local peer database.

- I/O module: receives and sends messages to acquaintees. The messages encapsulate queries and meta-data.
- User interface: enables the interaction with the user, who can pose queries, watch the query answering and learning procedures, give feedback and fine-tune the system.
- Acquaintance manager: manages meta-data, mappings and status of acquaintances.

Query rewriting mechanism: produces the successively rewritten versions of the incoming queries based on the available mappings. The rewriting is performed through the:

- GAV rewriting module: produces the rewritten query based on the available GAV mappings. The module preprocesses the query so that it is approximated in order to be rewritable by the available mappings and implements the GAV rewriting algorithm as in [4].
- LAV rewriting module: produces the rewritten query based on the available respective LAV mappings. The module implements an algorithm for approximate query rewriting that processes the query towards a suitable approximation as it is trying to rewrite it. This algorithm is described in [7].

Schema discovery mechanism: gradually builds and comprehends the schema of remote peers. The remote schema is augmented with new elements (relations, attributes, constraints, values) that arrive with queries. Schema comprehension is performed using:

- Automatic schema matching module: discovers schema correspondences between the already known part of the remote schema and the local one. *GrouPeer* employs COMA++ [2].
- Mapping construction module: formulates GAV and LAV mappings between the local and remote schemas based on schema correspondences and rewritings of incoming queries produced by the query rewriting mechanism [6, 5].

Learning mechanism: implements the core iterative learning functionality of the system. It consists of the following:

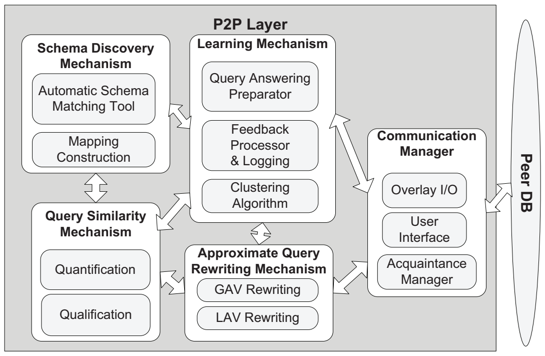


Figure 4: Architecture of *GrouPeer*.

- Query answering preparator: implements the procedure the takes as input the incoming query and prepares the versions Q_{ar} and Q_{sra} that are candidates to be rewritten locally by the approximate query rewriting mechanism, and decides which one should be propagated.
- Feedback processor and logging: manages the user feedback and keeps logs about the history of the remote schema learning procedure, as well as for the incoming queries, such as previous decisions and their success.
- Clustering module: accumulates and statistics related to learning, queries and feedback per remote peer. It monitors the progress of learning and decides if the remote peer is of enough local interest to become an acquaintee.

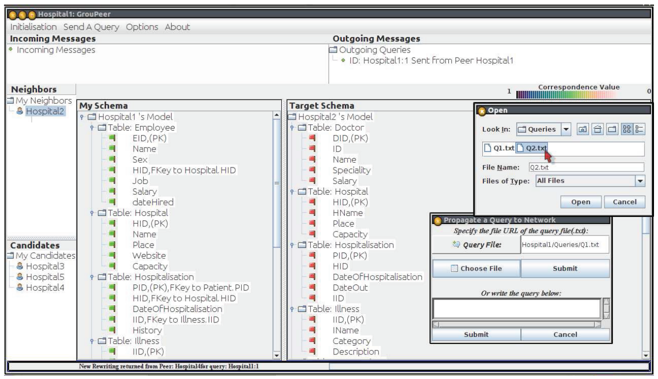
Query similarity mechanism: implements the semantic query similarity rationale and gives similarity evaluations. It consists of:

- Quantification module: contains a pool of functions that quantify query similarity according to various rationales of semantic similarity based on structural query features.
- Qualification module: contains tools of domain knowledge (dictionaries, ontologies, libraries, etc) that can assist in choosing the appropriate similarity function.

4. DEMONSTRATION

The goal of the demonstration is twofold: (a) to verify that all the modules of *GrouPeer* implement the proposed novel underlying techniques and algorithms, and exhibit how the latter behave in practice, and (b) to show that all the modules can work tightly together and achieve dynamic ongoing clustering of peer databases with random initial overlay connectivity, according to the semantics of their schemas and queries, and, therefore, their interests in information exchange, as proposed in our earlier work [3, 9].

Database Schemas. We demonstrate the effectiveness of *GrouPeer* in realistic overlays of peer databases that belong to either (a) the medical domain (i.e. hospitals, doctors, clinics etc), or (b) the education domain (i.e. universities, schools, colleges etc). The peer schemas are selected from two big collections of real schemas that were created as follows: For each one of these two domains, we created a large pool of related concepts; we gave the latter to people with good knowledge of database design (undergraduate and graduate students that have taken at least one course in data management) and we asked them to produce relevant original schemas with names of schema features or even data values that come from the respective pool of concepts. After collecting these original schemas, we artificially produced additional new schema groups in order achieve various schema similarities. Each collection (medical and education domain) consists of 50 schemas. More



Incoming Messages

ID: Hospital1:1 Sent From Peer Hospital1

File Name: Q2.txt

Query File: Hospital1/Queries/Q1.txt

New Rewriting returned from Peer: Hospital4for query: Hospital1:1

Figure 5: Peer 'Hospital1' issues a query.

on the schema collections can be found in [8]. After selecting peer schemas, these are populated with data. Visitors are enabled to select a peer schema and populate it.

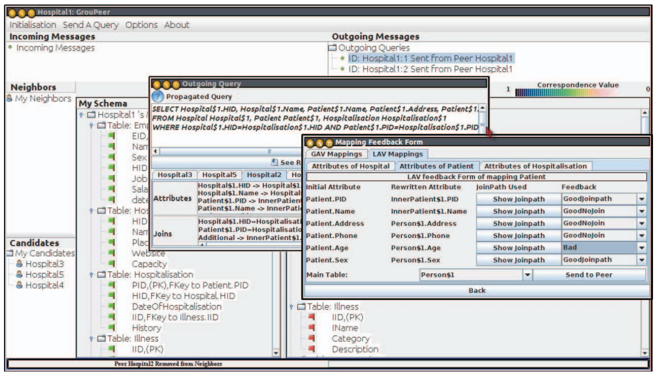
Demonstration Scenarios. The demonstration includes three sets of scenarios that aim to show all aspects of the system functionality.

Scenarios A: These scenarios aim to show the overall functionality of GrouPeer. The visitors are presented with an overlay of on average 8 real peers with random initial connectivity. The peers issue and broadcast a number of queries that are answered by other peers (TTL = 7), e.g. Figure 5 shows the *GrouPeer* interface of peer 'Hospital1' that issues and propagates query 'Q2'. It also shows the peer's schema as well as the mapped part of the schema of its neighbor 'Hospital2'. We demonstrate the overall learning procedure for any pair of peers that exchange queries and answers (data), including the gradual revealing of remote peer schemas and gradual building of mappings, the improvement of approximate query rewritings and the user feedback. Visitors also see graphically the alternative query propagation paths between pairs of peers (and we will discuss how these lead to learning) as well as a graphical representation of the clustering overlay.

Scenarios B: The visitors are able to see and play with interesting scenarios of approximate successive and automatic query rewriting. The demonstration shows the functionality of the approximate query answering mechanism which collaborates with the automatic schema matching mechanism. The scenarios will show that, even with no or basic dictionary, without any sophisticated complex general knowledge, automatic matching is a key advantage to the learning procedure.

The demonstration also shows the formulation of knowledge of remote schemas into schema mappings. The visitors watch the gradual construction of mappings, as an application tightly coupled with the automatic schema matching tool. Mapping construction is based on the posed queries and user feedback in consequent iterations. We emphasize on the fact that these are not only correct mappings, but also useful, since they reflect the mutual interests of the two remote peers, based on which they may decide to become acquainted. Figure 6 shows peer 'Hospital1' evaluating a query answer and advising the mapping creation procedure.

Scenarios C: These scenarios focus on vital details on the approximate query answering and mapping construction algorithms.
i. similarity functions: The similarity function is an input to the query rewriting algorithm. We show the effect of similarity functions on the lines of those in [9] and we will discuss their appropriateness depending on the semantics and structure of the query.
ii. correspondences: The schema correspondences are an input to the mapping algorithm and can change while mappings are evolving. Figure 7 shows an evolving correspondence. We show cases



Incoming Messages

ID: Hospital1:1 Sent From Peer Hospital1
ID: Hospital1:2 Sent From Peer Hospital1

```
SELECT Hospital$1.HID, Hospital$1.Name, Patient$1.Name, Patient$1.Address, Patient$1.
FROM Hospital Hospital$1, Patient Patient$1, Hospitalisation Hospitalisation$1
WHERE Hospital$1.HID=Hospitalisation$1.HID AND Patient$1.PID=Hospitalisation$1.PID
```

```
Hospital$1.HID -> Hospital$1
Hospital$1.Name -> Hospital
Patient$1.PID -> InnerPatient
Patient$1.Name -> InnerPati
Hospital$1.HID=Hospitalisati
Patient$1.PID=Hospitalisatio
Additional -> InnerPatient$1.
```

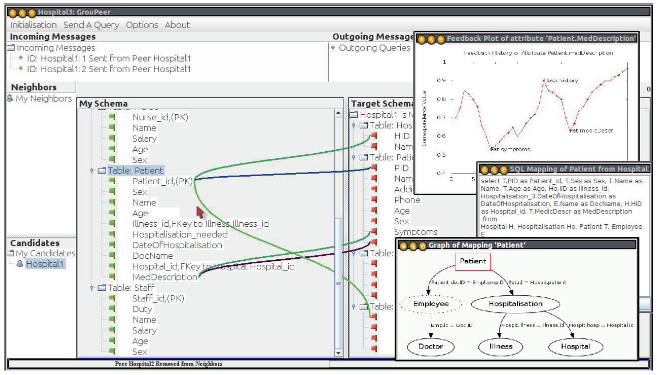
LAV feedback Form of mapping Patient

Initial Attribute	Rewritten Attribute	JoinPath Used	Feedback
Patient.PID	InnerPatient$1.PID	Show Joinpath	GoodJoinpath
Patient.Name	InnerPatient$1.Name	Show Joinpath	GoodNoJoin
Patient.Address	Person$1.Address	Show Joinpath	GoodNoJoin
Patient.Phone	Person$1.Phone	Show Joinpath	GoodNoJoin
Patient.Age	Person$1.Age	Show Joinpath	Bad
Patient.Sex	Person$1.Sex	Show Joinpath	GoodJoinpath

Main Table: Person$1 Send to Peer

Peer Hospital2 Removed from Neighbors

Figure 6: Peer 'Hospital1' evaluates query answer.



ID: Hospital1:1 Sent From Peer Hospital1
ID: Hospital1:2 Sent From Peer Hospital1

```
select T.PID as Patient_id, T.Sex as Sex, T.Name as
Name, T.Age as Age, Ho.IID as Illness_id,
Hospitalisation_3.DateOfHospitalisation as
DateOfHospitalisation, E.Name as DocName, H.HID
as Hospital_id, T.MedicDescr as MedDescription
 from
Hospital H, Hospitalisation Ho, Patient T, Employee
E
```

Peer Hospital2 Removed from Neighbors

Figure 7: Peer 'Hospital3' building mappings.

where the correspondences change according to the revelation of schema knowledge (example of specific case: bad correspondences change to good ones) and how the mapping algorithm adapts the mappings to these changes.
iii. mapping joinpath selection: The mapping algorithm takes as input the criteria according to which it selects one among alternative joinpaths for pair of relations that participate in a mapping. Namely, the decision is a weighted average of the following criteria: (a) user feedback, (b) overlap with the rest of the joinpaths in the same mapping, (c) joinpath length. We show how the evolving mappings improve when the weights change dynamically: e.g. in the beginning, automatically assign equal weight values (e.g. 0.33 to all), but later, change based on feedback. We also discuss how the different criteria are more successful depending on star or chain mappings. Figure 7 shows the joinpaths for a mapping.

5. REFERENCES

[1] MySQL.http://www.mysql.com/.
[2] COMA++.http://dbs.uni-leipzig.de/Research/coma.html.
[3] GroupPeer. http://www.dblab.ece.ntua.gr/∼vkante/groupeer.
[4] A. Halevy, Z. Ives, D. Suciu, and I. Tatarinov. Schema Mediation in Peer Data Management Systems. In *ICDE*, pages 505–516, 2003.
[5] V. Kantere, D. Bousounis, and T. Sellis. Mapping discovery over revealing schemas. (tech. rep. available, submitted for publication).
[6] V. Kantere, D. Bousounis, and Timos K. Sellis. A tool for mapping discovery over revealing schemas. In *EDBT*, pages 1124–1127, 2009.
[7] V. Kantere, G. Orfanoudakis, and T. Sellis. Approximate query answering in a pdms. (tech. rep. available, submitted for publication).
[8] V. Kantere, M.-E. Politou, and T. Sellis. Conceptual synopses of semantics in social networks sharing structured data. In *OTM Conferences (2)*, pages 1367–1384, 2008.
[9] V. Kantere, D. Tsoumakos, T. Sellis, and N. Roussopoulos. Groupeer: Dynamic clustering of p2p databases. In *Inf. Syst.*, volume 34, pages 62–86, 2009.