Building a PDMS Infrastructure for XML Data Sharing with SUNRISE [*]

Federica Mandreoli, Riccardo Martoglia, Simona Sassatelli and Giorgio Villani
DII - University of Modena and Reggio Emilia, Italy
{federica.mandreoli, riccardo.martoglia, simona.sassatelli, giorgio.villani}@unimo.it

Wilma Penzo
DEIS - University of Bologna, Italy
wilma.penzo@unibo.it

ABSTRACT

Semantic support for data representation as well as a flexible machine-readable format have made XML the de facto standard for Internet applications semantic interoperability. Its applicability is primarily evident in realities where actors are heterogeneous data sources which interact each other for data sharing purposes. This is exactly the scenario envisioned by Peer Data Management Systems (PDMSs), where autonomous sources (peers) model their local data according to a schema, and are connected in a peer-to-peer network by means of pairwise semantic mappings between the peers' own schemas. One of the main challenges in such a semantically heterogeneous environment is concerned with query processing when dealing with the inherent semantic approximations occurring in the data.

In this paper we present an instantiation of SUNRISE (System for Unified Network Routing, Indexing and Semantic Exploration) for XML data sources. SUNRISE is a complete PDMS infrastructure which extends each peer with functionalities for capturing the semantic approximation originating from schema heterogeneity and exploiting it for a semantically driven network organization and query routing.

Categories and Subject Descriptors

C.2.3 [**Computer-Communication Networks**]: Network Operations—*Network Management*; E.1 [**Data**]: Data Structures—*Distributed Data Structures*; H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Search Process*

General Terms

Management

DataX'08, March 25, 2008 Nantes, France

Keywords

XML, PDMS, Semantics, Query processing, Network organization

1. MOTIVATION AND RELATED WORK

In recent years, the enormous success of Internet has stressed the importance of a general agreement on the format for data exchange. For this purpose, XML proved to be a widely accepted standard both for its flexible machine-readable form, and for the semantic support it provides for data representation. This last feature, in line with the Semantic Web vision [4], has made XML to be extensively and successfully used by several applications dealing with semantically rich data. Its applicability is primarily evident in distributed realities, where actors are heterogeneous data sources which interact with each other for data sharing purposes.

This is exactly the scenario envisioned by Peer Data Management Systems (PDMSs)[9, 23, 2], a recent evolution of peer-to-peer (P2P) systems towards a more semantics-based description of peers' contents and relationships. In a PDMS peers are autonomous sources which model their local data according to a schema, and are connected in a peer-to-peer network by means of pairwise semantic mappings between the peers' own schemas. Because of the absence of common understanding of the vocabulary used at each peer's schema, the semantic mappings established between the peers implement a decentralized schema mediation [9].

One of the main challenges in such a semantically heterogeneous environment is concerned with query processing. A query is routed through the network by means of a sequence of reformulations, according to the semantic mappings encountered in the routing path. As reformulations may lead to semantic approximations, thus inducing information loss, for a given peer, the linkage closeness to semantically similar peers is a crucial issue. This matter has also been evidenced recently by works on Semantic Overlay Networks (SONs) [1, 6, 8, 12, 24] for P2P systems, where peers with semantically similar content are clustered together in logical subnetworks. The main aim of a SON is to improve the efficiency of query processing by limiting the number of contacts only to relevant peers. Nevertheless, in a more complex environment like the PDMS one, SON principles substantially improve the effectiveness of query answering, by reducing the semantic degradation due to the traversal of semantic mappings towards irrelevant peers [13].

---

[*] This work is partially supported by the Italian Council co-funded project NeP4B

However, the problem of answering queries efficiently is only partially solved by simply relying on a carefully designed network organization. Indeed, SONs would largely benefit of a support for query routing, i.e. a mechanism for selecting a small subset of relevant peers to forward a query to. This issue has been the subject of several research works [5, 11, 15], with the aim of cutting off the negative effects of query flooding techniques which both overwhelm the network with messages, and often return lots of irrelevant results.

In this paper we present an instantiation of SUNRISE (System for Unified Network Routing, Indexing and Semantic Exploration) for XML data sources. SUNRISE is a complete PDMS infrastructure which offers peers specific functionalities in the following stages which characterize a PDMS life:

Network construction. Techniques and index structures for selecting the best SONs to join to, as well as for efficiently locating the semantically closest neighbors to be connected to, are provided for each peer entering the network. This is achieved thanks to a suite of protocols and algorithms for managing the update and evolution of the infrastructure in an incremental fashion;

Network exploration. Routing algorithms and a specifically devised indexing mechanism are at peers' disposal for a wise query answering which selectively locates the most relevant peers to be contacted.

The overall process of network management/usage is semantics-driven, in that it is aware of the semantic approximations originating from the peers' schemas heterogeneity. Further, the system is complemented with two nice features: 1) a simulation environment able to reproduce the main characteristics of a PDMS setting without requiring a real network of peers; 2) a user-friendly GUI providing an easy-to-use layout of the main functions of the system and showing its behavior, also in a step-by-step fashion, during the interaction with the user. Leveraging on our previous works [20, 15, 16, 13, 17], the main aim of this paper is to show that it is possible to construct a P2P network for XML data sharing where peers interact for a semantics-driven effective and efficient query processing.

In Section 2 we provide an overview of the system working, by introducing a running example of network organization. Section 3 is devoted to the presentation of SUNRISE architecture through a detailed description of the modules composing the system. In Section 4 we discuss a series of experiments we conducted with SUNRISE. Finally, conclusions are drawn in Section 5.

2. THE SUNRISE INFRASTRUCTURE

The SUNRISE infrastructure relies on a PDMS architecture where a collection of autonomous peers model their local data through schemas and are pairwise connected through semantic mappings. SUNRISE supports the creation and maintenance of a flexible network organization for PDMSs that clusters together in SONs heterogeneous peers which are semantically related. Figure 1 shows a sample of network made up by two SONs concerning cinema-related data. Each peer is represented by the main topics of interest derived from its schema. In case of tree-based structure, like XML schemas, they correspond to the abstract elements which can be obtained by applying a schema summarization technique like the one proposed in [26]. Some peers of the network, such as the Internet Movie Database (IMDb, Peer B) and the web site HolidayInRome (Peer F) are "monothematic", i.e. they only deal with movies and movie theaters, respectively. Other peers, instead, are concerned with both themes, e.g. FindATheater.

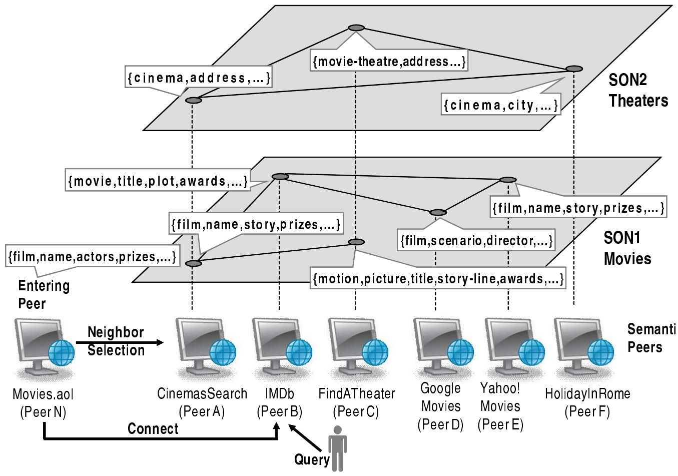


Figure 1: Sample network organization

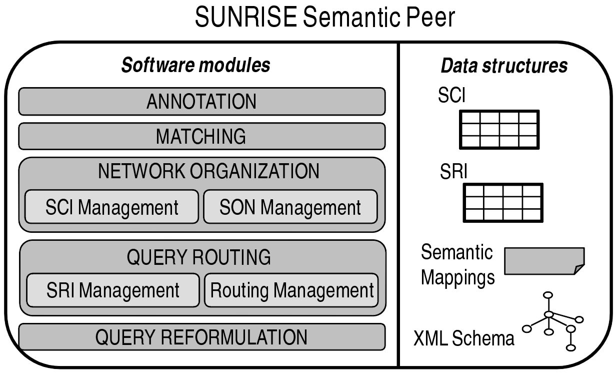


Figure 2: Peer's internal architecture in SUNRISE

Peers react to the events issued to the network by interacting on the basis of a message exchange protocol. Basically three kinds of events are supported: As to the *network construction* phase, the `NeighborSelection` event is devoted to assist each newly entering peer in the selection of the semantically closest peers as its neighbors while a `Connection` event allows the actual connection between peers; as to the *network exploration* phase, a `Query` event is posed at a peer in order to be answered by the most semantically related peers in the network. To implement the message exchange protocol, each peer maintains appropriate data structures and specific software modules (see Figure 2).

While the SUNRISE architecture has been conceived in order to be completely independent from the data model adopted for schema representation and query formulation, the data model peculiarities are supported by the actual implementation of these software modules. This paper focuses on the XML data model in order to provide a PDMS with an infrastructure for XML data sharing where peers interact for a semantic-driven effective and efficient query processing.

In the following, we will show how such modules interact and access the data structures whereas functionality details will be given in the next section. In particular, as to the data structures, besides the already presented *XML schema* each peer maintains the *Semantic Mappings* that provide the connections with its neighbors. Two index structures are maintained too: a *Semantic Clustering Index (SCI)* which is used in the network construction phase, and a *Semantic Routing Index (SRI)*, which is exploited in the network exploration phase for query routing purposes.

2.1 Network construction

As to *network construction*, every time a new peer joins the PDMS, it first activates the *Annotation Module* which makes explicit the semantics of its schema by associating each schema's term with the right concept. Then, the peer chooses its neighbors and a `NeighborSelection` event is generated. SUNRISE assists each newly entering peer in the selection of its neighbors in a two-fold fashion (see Figure 3): First, in a coarse-grained choice of the semantically closest overlay networks; Then, within each overlay network, in a fine-grained selection of the best neighbors among the most semantically related peers [13]. Peers are assigned to one or more SONs on the basis of their own concepts. In a PDMS, this operation is a really challenging one because of the lack of a common understanding among the peer's local dictionaries. This means that similar or even the same contents in different peers are not usually described by the same concepts. Our proposal is to solve such heterogeneity by clustering together in the same SON nodes with semantically similar concepts. Semantic similarity is also at the basis of the approach we propose to guide the selection of the neighbors within each SON. As a running example for the network construction phase, in the following we will consider the network join request of the new entering peer Movies.aol (Peer N), which we suppose being "monothematic" and willing to find only one peer, its semantically nearest neighbor, to connect to.

SON selection relies on a "light" and scalable structure, the Access Point Structure (APS), which maintains summarized descriptions about the SONs available in the network in order to help the newly entering peers to decide which SONs to join or whether to form new SONs. Though being conceptually centralized, the APS can be stored in a distributed manner among different peers and maintains information such as SONs' representative concepts which the peers can compare with the concepts in their own XML schemas. In order to perform such an operation, a preliminary phase of concepts extraction from each own XML schema must be executed by each entering peer aiming at identifying the concepts which better describe the peer's main topics of interest. As an example, Peer N in Figure 1 which, dealing with movies, extracts from its XML schema the concepts `{film, name, actors, prizes, ...}`, is assigned to SON1.

After SON selection, the peer starts to navigate the link structure within each selected SON from the entry-point peer associated to the SON. We assist the peer in the selection of the semantically closest peers by providing two policies: 1) Range-based, where the new peer connects to all the peers in a given semantic similarity range, and 2) kNN-based, where the k semantically closest peers are chosen. Each peer receiving a (knn or range) `NeighborSelection` message activates the *SON Management Module* which determines whether it could belong to the selection required or not and, in the `NeighborSelection` message forwarding phase, it exploits its SCI to prune out non-relevant neighbors. Going back to our example, we suppose Peer N wants to find its best neighbor in SON1. To this end, it executes a KNN-selection with $k = 1$ and connects to the peer with the most similar schema, i.e. Peer B.

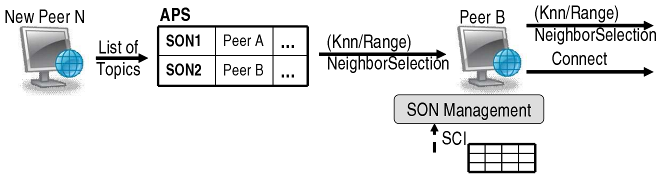


Figure 3: Actions performed following a `NeighborSelection` event

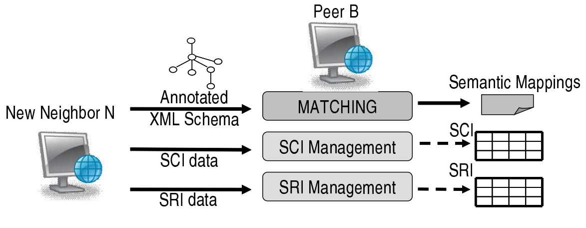


Figure 4: Actions performed following a `Connection` event

Figure 4 shows the actions which are performed when a `Connection` event occurs, i.e. every time a new connection is established in the network. Notice that each connection is a pairwise operation and consequently, as it is shown in the figure, basically involves two peers. Each peer receiving a `Connection` event (Peer B in the figure) also receives the annotated XML schema which is used by the *Matching Module* to individuate the semantic correspondences with its schema. Then, the *SRI Management Module* and the *SCI Management Module* add a reference to the new neighbor in the corresponding index structures.

2.2 Network exploration

In a PDMS a query is posed at a peer and answers can come from any peer in the PDMS which is connected through a semantic path of mappings. Broadly speaking, the PDMS starts from the querying peer and reformulates the query over its immediate neighbors, then over their immediate neighbors, and so on [9]. Thus, when a query is forwarded through a semantic path, it undergoes a multi-step reformulation which may involve a chain of semantic approximations. Due to the heterogeneity of the schemas, each reformulation step may lead to some semantic approximation and, consequently, the returned data may not exactly fit with the query conditions. SUNRISE avoids query broadcasting and exploits such approximations for selecting the direction which is *more likely* to provide the best results to a given query [15, 16]. As a reference example, we will consider the request of an IMDb (Peer B) user asking for "the plot of the movie titled Indiana Jones IV and directed by

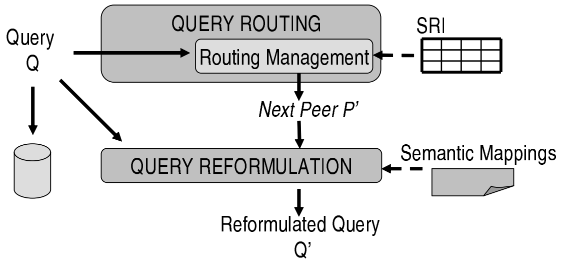


Figure 5: Actions performed following a `Query` event

Steven Spielberg”.

Figure 5 depicts the actions a peer performs when it receives a `Query` event. After the execution of the query on the local data set and the collection of local results, the *Query Routing Module* is activated. In particular, the *Routing Management* sub-module analyzes the received query and accesses the peer’s SRI in order to select the neighbor (P' in the figure) rooting the most relevant subnetwork among the unvisited ones. Neighbor selection is done on the basis of the policies presented in 3.4. Then, the *Query Reformulation Module* uses the semantic mappings towards P' to reformulate the received query Q into Q'. Q' can now be sent to P' which will manage the `Query` message it receives in a similar way.

3. AN XML INSTANTIATION OF SUNRISE

In this section we will show how the modules depicted in Fig. 2 act to create a P2P network for XML data sharing and to explore the network for executing XQuery queries.

3.1 Annotation Module

SUNRISE’s *Annotation Module* overcomes the ambiguity of natural language schema terms, as it makes explicit the meanings of the words employed in the peer’s schemas. Indeed, schemas often contain many polysemous words and their meanings could be very different one from the others. Let us examine, for instance, some of the terms in Peer B XML schema (consider again Figure 1) along with some of their meanings extracted from WordNet: `plot` could be “a secret scheme to do something” (sense 1), “the story that is told in a movie” (sense 2), or many others, `title` could be a “statute title” (sense 1), “the name of a work of art” (sense 2), “the status of being a champion” (sense 3), and seven others. In order for the schema matching and, consequently, the query processing phase to be effective, it is fundamental to be able to determine the right meaning of the employed terminology. To this end, the annotation module exploits the novel versatile structural disambiguation approach we proposed in [18, 19] and automatically annotates the schemas with the most probable senses extracted from WordNet.

The idea behind our annotation approach is to disambiguate the terms occurring in the nodes’ labels by analysing their schema context and by using WordNet as an external knowledge source. Starting from the original XML schema, the annotation module first derives a tree structure representing the underlying conceptual organization. Figure 6-a depicts a fragment of such structure for both Peer B and A: The trees abstract from the several complexities of the XML schema syntax and only represent the fundamental concepts (nodes) together with their relations (e.g. `Title` is an attribute of `Movie` and, therefore, is represented as its child).

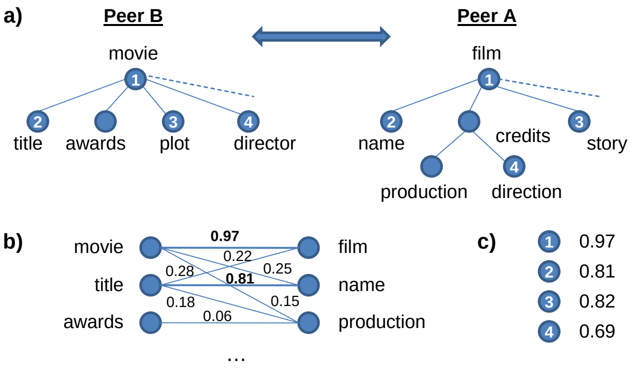


Figure 6: A portion of Peer A and Peer B schemas and some details about their matching process

Then, starting from each given node, several ways of navigating the schema tree are supported in order to extract their context. In its simplest form, each term’s context contains those terms labeling all the nodes in the tree which are reachable from the one considered. However, depending on the schema characteristics, it is possible to limit the context by excluding distant and, therefore, unrelated and potentially misleading terms. Finally, specifically devised disambiguation algorithms make use of the hypernymy/hyponymy hierarchy, as suggested by most of the classic WSD studies, in order to determine each term’s most probable meanings w.r.t. such context. For example, in Peer B’ schema, term `plot` has many movie-related terms, such as `movie` and `director` in its context, the resulting confidence in choosing sense 2 (the right one) will be much higher than, for instance, sense 1, which is about secret plans. Further, additional information coming from the thesaurus, such as the nouns used in the terms definitions and usage examples of each term’s senses, can be compared against the schema context so to automatically understand which is the meaning closest to the one in the schema. For instance, for `plot`, the most significant to our context are the nouns from sense 2 examples, containing terms like `movie character`. The outcome of the disambiguation process is a ranking of the plausible senses for each term.

3.2 Matching Module

The *Matching Module* is able to automatically generate semantic matches between the current peer’s schema (source) and a newly connected neighboring peer (target). In particular, it associates each source concept to a corresponding target concept according to a *score*, denoting the *degree of semantic similarity* between the two concepts. Such similarity is a number between 0 (no similarity) and 1 (identity). For instance, considering the two XML schema fragments from Peer A and B schemas in Figure 6-a: While their structure and element names are different, they clearly represent similar concepts and the correspondences resulting from their matching are represented by the same number (see Figure 6-c for the final match scores).

Since these matching scores have to be exploited for network organization and network exploration purposes, the main characteristic we required for our matching module is the ability to correctly capture the semantic approximation originating from schema heterogeneity and quantifying it by means of scores which have to be *comparable*. Among the several existing schema matching approaches [7, 3, 14, 10] we choose to drawn inspiration form the Similarity Flooding algorithm, originally proposed in [21]. Similarity Flooding is a generic graph matching algorithm which uses fixed point computation to determine corresponding nodes in the graph; the principle of the algorithm is that the similarity between two nodes must depend on the similarity between their adjacent nodes. Our approach goes beyond the Similarity Flooding algorithm by considering both the structure of the corresponding trees and the semantics of the involved terms, as extracted by the annotation module.

In particular, in order to identify the "best" matchings, SUNRISE *Matching Module* operates according to the following steps (see [20] for an in-depth explanation):

- the involved schemas are first converted into directed labelled graphs following the RDF specifications [1];
- from the RDF graphs of each pair of schemas a *pairwise connectivity graph (PCG)*, involving node pairs, is constructed [21], and an initial similarity score is computed for each node pair contained in the PCG. Similarly to the annotation approach, we follow a *linguistic approach* in the computation of the similarities between terms. Specifically, the scores for each pair of annotated terms (t_1, t_2) are obtained by computing their depths in the WN hypernyms hierarchy and the length of the path connecting them as follows [20]:
$$\frac{2 * \text{depth of the least common ancestor}}{\text{depth of } t_1 + \text{depth of } t_2};$$
- such similarities, reflecting the semantics of the single node pairs, are refined by an iterative fixpoint calculation [21], which brings the structural information of the schemas in the computation by propagating the similarity of the elements to their adjacent nodes;
- finally, a stable marriage filter and a threshold filter are applied to the resulting network of correspondences. The stable marriage filter guarantees that, for each pair of nodes (x, y), no other pair (x', y') exists such that x is more similar to y' than to y and y' is more similar to x than to x'; on the other hand, the threshold filter ensures that very loose (and, thus, potentially wrong) matches do not appear in the final matches. For instance, from the graph of Figure 6, the two filters extract the right correspondences for Peer B' `movie` and `title` (matches "1" and "2", respectively), while the `awards` node is not assigned to a corresponding one in Peer A's schema.

Generally speaking, schema matching is the first step towards mappings that defines how to represent the source schema's concepts in terms of the target schema vocabulary [22]. Obviously the quality of mappings influence the effectiveness of query processing in a PDMS but the techniques we propose for network construction and exploration are completely independent from the specific format that semantic mappings may have. Indeed, our main concern is about the approximation originating from vocabulary heterogeneity. For this reason, we consider a simplified scenario where the outcome of the matching module actually corresponds to the directional, pairwise and one-to-one semantic mappings each peer stores in its local folder.

3.3 Network Organization Module

The *Network Organization Module* contains the actions each peer executes for network creation and it is made up of two sub-modules: The *SCI Management* and the *SON Management* modules. Notice that the distance used to quantify the semantic (dis)similarity between concepts is required to be a metric [13].

3.3.1 SCI Management Module

The *SCI Management Module* provides each peer with the functionalities for the creation and management of the indexing structures used in the neighbors selection process: The Semantic Clustering Indexs (SCIs). Indeed, in order to guide a peer joining the network towards its best position in the selected SONs, each peer maintains a SCI which contains summarized information about the concepts which can be reached in each available direction. In particular, for each SON SON_i the peer belongs to and for each neighbor n_j, the SCI represents the set of SON_i's concepts which are reachable in n_j's subnetwork through a clustroid concept and a radius such that all such concepts are within the radius from the clustroid. The *SCI Management Module* assists each peer in creating its SCI when entering the network implementing a specifically devised protocol whose details are given in [13]. The same protocol regulates the actions which are executed in order to maintain the SCIs updated whenever the network changes.

3.3.2 SON Management Module

Each peer receiving a (knn or range) `NeighborSelection` message activates the *SON Management Module.* Such module exploits the peer's SCI to lighten the neighbor selection process. The objective is to reduce the network load, i.e. the number of accessed peers and the computational effort which is required from each accessed peer. To this end, the information stored in the SCI are appropriately exploited to effectively apply the triangular inequality to prune out non-relevant peers and to avoid useless distance computations.

3.4 Query Routing Module

The role of the *Query Routing Module* is to provide the PDMS with advanced semantic query routing functionalities and it consists of two parts: The *SRI Management Module*, which has the role of managing the creation and evolution of the index structures involved in the routing process (the *SRIs*); and the *Routing Management* one, which helps each peer receiving a query in routing it towards the best subnetworks originating at its neighbors.

3.4.1 SRI Management Module

Each peer maintains a *Semantic Routing Index (SRI)* containing cumulative information which summarize the semantic approximation capabilities w.r.t. its schema of the whole subnetworks rooted at each of its neighbors. In particular, for each schema concept c_i and each neighbor n_j, the SRI contains a score expressing how c_i is semantically approximated by the subnetwork rooted at n_j. For instance, in

---

[1] `http://www.w3.org/RDF/`

$\mathrm{SRI}_{\mathrm{PeerB}}$	movie	title	plot	...
PeerB	1.00	1.00	1.00	...
PeerA	0.91	0.88	0.83	...
PeerE	0.77	0.70	0.41	...
PeerD	0.72	0.68	0.30	...

Figure 7: A portion of Peer B's SRI for the reference example

Figure 7 representing Peer B's SRI for the reference example, the concept `plot` is approximated with a score of 0.83 by the subnetwork rooted at Peer A.

The operations for the creation and update of the SRIs are on the *SRI Management Module*'s responsibility. In particular, each peer creates its own SRI when entering the network following a specifically devised protocol whose details are given in [15]. Further, the same protocol regulates the update of the SRIs which must occur in response to network modifications.

3.4.2 Routing Management Module

With SRIs at the PDMS's disposal, in query forwarding phase, a peer P accesses its own index for determining the neighboring peers which are most semantically related to the concepts in q. For example, if a query q refers to a single concept C, the choice of the semantically best neighboring peers can be done by evaluating the column of its SRI corresponding to C: This means that Peer A would be the selected neighbor for the concept `plot` in Figure 7. In general, each more realistic and thus complex XQuery query involving several concepts can be interpreted as a formula of predicates specifying the query conditions and combined through logical connectives. In this case, the choice of the best neighbor can thus be done by applying scoring rules which, for each neighboring peer P_i, combine the corresponding grades in the SRI for all the corresponding concepts in q. Specifically, the fuzzy logic approach presented in [16] is adopted. Going back to our example (Fig. 7), the score of Peer A for a query involving the concepts `plot` and `title` connected through and AND operator would be $\min(0.83, 0.88)$, being conjunction dealt with the minimum.

Assuming that an overall score is somehow obtained for a complex query, different routing strategies can be executed, each having effectiveness, efficiency or a trade-off between the two as its priority. In particular, the routing strategies the *Routing Management Module* can implement belong to two main families of navigation policies: The *Depth First (DF)*, which pursues efficiency as its objective (i.e. its main objective is to minimize the query path), and the *Global (G)*, or *Goal-based* model, which is designed for effectiveness (i.e. its main objective is to maximize the relevance of the retrieved results). The DF model provides an SRI-oriented depth first visiting criteria: It progresses going deeper and deeper in the network following, at each forwarding step, the path toward the neighbor characterized by the highest SRI value; backtracking is only performed when a "blind alley" is reached. Based on the DF model, the two following routing policies are implemented:

- **DF policy**: the "standard" depth-first policy, straightly implementing the DF model;
- **DFF (Depth-First Fan) policy**: a variation of DF, performing depth-first visit with an added twist. Specifically, at each node, DFF performs a "fan" by exploring all the neighbors, then it proceeds in depth to the best subnetwork, as DF does. DFF is an attempt to enhance DF, as it tries to capture in less hops more answers coming from short semantic paths and, thus, being potentially more relevant than those retrieved by DF.

In order to better explain how the DF policies work, let us consider our reference network and see how a query posed on Peer B and involving the only concept `plot` would be routed (see Figures 1) and 7. We use the following notation: We present the routing sequence of hops as an ordered list, where each entry P means peer P is accessed and queried, while (P) denotes a backtracking hop through peer P. We consider the navigation until peers B through C (the most relevant ones) have been queried. For the DF policy this would be the behavior: Peer B, Peer A (most promising subnetwork rooted at Peer B), Peer C, (Peer A), (Peer B), Peer E, Peer D. For DFF: Peer B, Peer A, (Peer B), Peer E, (a fan is performed before exploring the best subnetwork), (Peer B), Peer D, (Peer B), (Peer A), Peer C.

Differently from the DF model, in the G one each peer chooses the best peer to forward the query to in a "global" way: It does not limit its choice among the neighbors but it considers all the peers already "discovered" (i.e. for which a navigation path leading to them has been found) during network exploration and that have still not been visited. This is mainly achieved by managing and passing along the network an additional structure, called *Goal List* (GL), which is a globally ordered list of *goals*. Each goal G contains information useful for next peer selection. In particular, it represents an arc in the network topology, starting from an already queried peer and going to a destination (and still unvisited) one. GL is always kept ordered on the basis of the goals' semantic relevances, which are calculated by means of an appropriate function taking into account the whole path originating from the querying peer. The G model simply progresses selecting the top goal in GL as the next peer to be queried. In this way, the G model constantly exploits backtracking in order to reach back potentially distant goals. Obviously going back to potentially distant goals (peers) has a cost in terms of efficiency but always ensures the highest possible effectiveness, since the most relevant discovered peers are always selected.

Based on the G model, two routing policies are implemented, which differ on the basis of the function used for the goals relevance computation:

- **G policy**: The function only considers the semantic relevance of the goals;
- **GH (Global Hybrid) policy**: This "hybrid" policy chooses goals following a trade-off between effectiveness and efficiency. This is achieved by introducing an ad-hoc parameterizable function f, which does not only consider a goal G's semantic relevance *semRel* but also its distance *hops* (expressed in number of hops) from current peer: $\mathtt{f}(semRel) = semRel/(hops)^k$, $k = 0 \ldots \infty$. By simply adjusting the value of k, the GH policy can be easily tuned more on efficiency ($k \to \infty$) or on effectiveness ($k \to 0$).

Going back to our reference example, this would be the routing sequence for the G policy: Peer B, Peer A, (Peer B), Peer E (since the relevance of the goal to Peer E is expected to be higher than the one to Peer C), (Peer B), (Peer A), Peer C.

3.5 Query Reformulation

The semantic mappings produced by the Matching module is exploited to reformulate the source query to a target one, compatible with the target peer's schema. As our mappings relate the target peer's schema concepts with the source ones, reformulation translates to unfolding [20].

At present, we support XQuery FLWOR conjunctive queries with standard variable use, predicates and wildcards. In particular, after having substituted each path in the `WHERE` and `RETURN` clauses with the corresponding full paths and then discarded the variable introduced in the `FOR` clause, all the full paths in the query are reformulated by using the best matches between the nodes in the given source schema and target schema (e.g. the path `/movie/director` of Peer B is automatically reformulated in the corresponding best match, `/film/credits/direction` of Peer A). Consider, for instance, the XQuery representation of our simple running example's query:

```
FOR $x IN /movie
WHERE $x/title = "Indiana Jones IV"
AND $x/director = "Steven Spielberg"
RETURN $x/plot
```

The reformulation module transforms it in the following target query, compatible with Peer A's schema:

```
FOR $x IN /film
WHERE $x/name = "Indiana Jones IV"
AND $x/credits/direction = "Steven Spielberg"
RETURN $x/story
```

A score is also assigned to the reformulated query: It is the composition (for instance, the average) of the scores assigned to each path reformulation which is based on the similarity between the involved nodes, as specified in the match. In our example (Figure 6): The final reformulation score is 0.82, since the exploited matches for `movie`, `title`, `plot` and `director` have scores of 0.97, 0.81, 0.82 and 0.69, respectively.

4. EXPERIMENTS

This section describes the empirical evaluation of SUNRISE, performed by means of its simulation environment [17] through which we were able to reproduce the main conditions characterizing a PDMS environment where autonomous peers freely decide when entering the system. The simulation engine is based on SimJava 2.0, a discrete, event-based, general purpose simulator. Figure 8 shows the GUI of our environment which provides a visual support during the evaluation. Through this framework we modelled scenarios corresponding to networks of semantic peers, each with its own XML schema describing a particular reality.

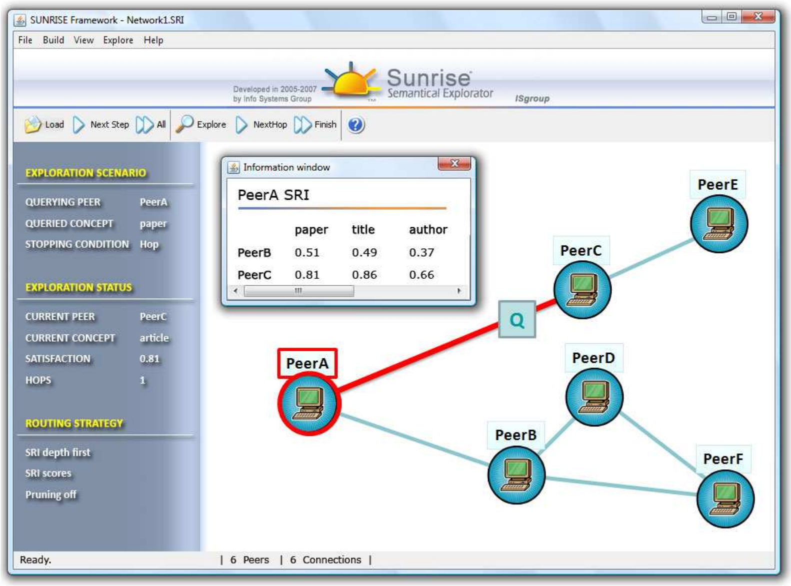
As in [25], the schemas are derived from real world-data sets, collected from many different available web sites, such as IMDb and DBLP Computer Society Bibliography, and enlarged with new schemas created by introducing structural and terminological variations on the original ones; in such a way we were able to fully test the potentialities of SUNRISE with large PDMSs of semantically related peers. The schemas differ for their complexity and dimension (they mean size is in the order of dozens of elements), and belong to three main domains: movies, publications and sport. The networks are automatically produced by SUNRISE network organization algorithms, which establish the connections among peers according to the semantic similarity between peers' schemas. The mean size of our networks is in the order of some hundreds of nodes.

In order to evaluate the benefits provided by the network construction and routing techniques, and thus the effectiveness and efficiency of SUNRISE query answering, we instantiated different queries on randomly selected peers where each query is a combination, through logical connectives, of a small number of predicates specifying conditions on concepts. This is the first time we evaluated SUNRISE exploiting all its features working together. More precisely, we quantified the advantages on query processing by propagating each query until a stopping condition is reached considering two alternatives: Stopping the querying process when a given number of hops (*hops*) has been performed and measuring the quality of the results (*satisfaction*) or, in a dual way, stopping when a given satisfaction is obtained and measuring the required number of hops. Satisfaction is a specifically introduced quantity that grows proportionally to the goodness of the results returned by each queried peer [15]. In particular, we compared the routing strategies presented in Section 3.4 together with the Global IP-based (GIP) policy, which is a variation of the Global (G) mechanism: A direct connection is established between the current peer and the peer chosen for the following step, avoiding the hops needed to reach it in the original network topology. This policy can not be considered a real P2P strategy, but it is an interesting upper-bound to be shown.

We considered two significant scenarios differing in the grade of semantic heterogeneity characterizing each peer's schema. In the first one, most peers's XML schemas are monothematic, while in the second one many are multithematic. Figure 9 shows the trend of the obtained satisfaction when we gradually vary the stopping condition on hops for the first scenario. As the Figure shows, SUNRISE network organization techniques allow a relevant improvement of the query processing effectiveness, even when no routing capabilities are available. In particular, comparing the R*[2] and R curves, we can appreciate the improvement provided by the SUNRISE network organization algorithms. Moreover, exploiting the routing techniques we can achieve even better effectiveness. Specifically, the obtained network organization allows the DF policy to achieve results near to the upper bound (GIP). Further, the DFF mechanism is initially less effective than the DF one, since it uses a large number of hops for performing its "fan" exploration. Nevertheless, for higher stopping conditions, it becomes increasingly more effective until it outperforms the DF policy: This is due to the fact that it visits nearer peers, which have a higher probability to provide better results. On the other hand, Figure 10 shows the results of the experiments aiming at verifying the efficiency of SUNRISE query routing: It represents the trend of the number of required hops for a given satisfaction goal. As we expected, the DF policy outperforms the oth-

---

[2] '*' denotes a test performed on a randomly constructed network topology.



EXPLORATION SCENARIO

QUERYING PEER PeerA
QUERIED CONCEPT paper
STOPPING CONDITION Hop

EXPLORATION STATUS

CURRENT PEER PeerC
CURRENT CONCEPT article
SATISFACTION 0.81
HOPS 1

ROUTING STRATEGY

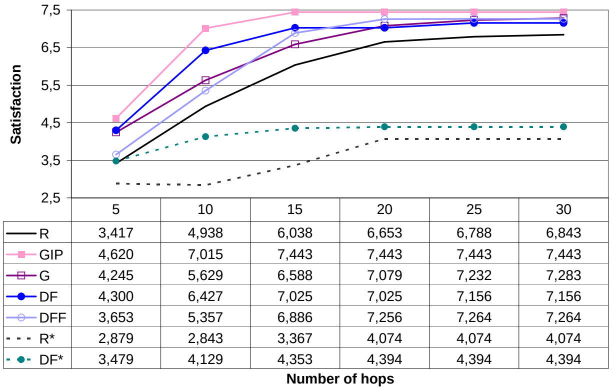
SRI depth first
SRI scores
Pruning off

PeerA SRI

	paper	title	author
PeerB	0.51	0.49	0.37
PeerC	0.81	0.86	0.66

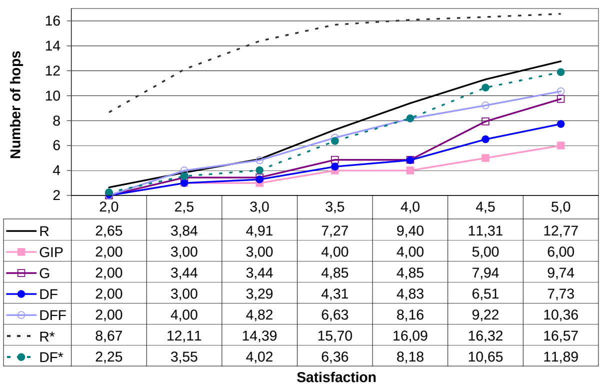
Ready. | 6 Peers | 6 Connections |

Figure 8: The graphical user interface of SUNRISE



	5	10	15	20	25	30
R	3,417	4,938	6,038	6,653	6,788	6,843
GIP	4,620	7,015	7,443	7,443	7,443	7,443
G	4,245	5,629	6,588	7,079	7,232	7,283
DF	4,300	6,427	7,025	7,025	7,156	7,156
DFF	3,653	5,357	6,886	7,256	7,264	7,264
R*	2,879	2,843	3,367	4,074	4,074	4,074
DF*	3,479	4,129	4,353	4,394	4,394	4,394

Figure 9: Satisfaction reached by routing policies given a maximum number of hops (first scenario)



	2,0	2,5	3,0	3,5	4,0	4,5	5,0
R	2,65	3,84	4,91	7,27	9,40	11,31	12,77
GIP	2,00	3,00	3,00	4,00	4,00	5,00	6,00
G	2,00	3,44	3,44	4,85	4,85	7,94	9,74
DF	2,00	3,00	3,29	4,31	4,83	6,51	7,73
DFF	2,00	4,00	4,82	6,63	8,16	9,22	10,36
R*	8,67	12,11	14,39	15,70	16,09	16,32	16,57
DF*	2,25	3,55	4,02	6,36	8,18	10,65	11,89

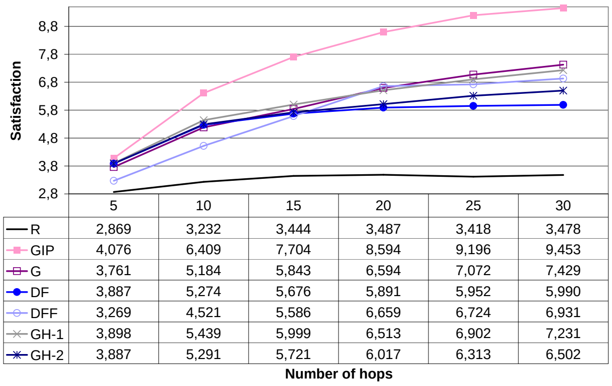
Figure 10: Mean number of hops needed to reach a given satisfaction (first scenario)

ers since its priority target is to minimize the query path. The DFF policy is instead the less efficient one, due to the number of hops for visiting all the neighboring peers.

In the more complex second scenario, routing becomes even more relevant for query processing, since interesting data are spread among a larger number of peers. In this scenario, the selection of the best peers to forward a query to is a fundamental challenge, as Figure 11 shows. Differently from the first scenario, the G policy shows the best behavior as it selects for each step the available peer with the highest semantic relevance approximation. The curves for the GH strategy are also represented: Notice that by tuning the k parameter, we can handle the trade-off between efficiency and effectiveness of the query routing. For clarity of presentation, we omitted the results of randomly constructed networks which are similar to the ones in the first scenario. Figure 12 shows the relation between the number of queried peers (efficiency) and the satisfaction that SUNRISE reaches (effectiveness) given a maximum number of hops. In this way, we are able to visualize how the results obtained by the different policies position themselves in a combined effectiveness/efficiency plane. As expected, we observe that the G policy is the most effective one, since its curve is located near to the satisfaction axis. In contrast, the DF policy appears as the most efficient one. Moreover, we can see the effect of k in the GH policy: Increasing k makes the GH policy more efficient, but less effective. Finally, notice that the DFF policy can reach satisfaction goals similar to the ones reached by the G strategy, but in a more efficient way.

5. CONCLUSIONS

In this paper, we demonstrated that a P2P network for XML data sharing can benefit from a semantics-aware infrastructure like SUNRISE. In our future work we plan to enhance query routing by also exploiting the information provided by SONs.



	5	10	15	20	25	30
R	2,869	3,232	3,444	3,487	3,418	3,478
GIP	4,076	6,409	7,704	8,594	9,196	9,453
G	3,761	5,184	5,843	6,594	7,072	7,429
DF	3,887	5,274	5,676	5,891	5,952	5,990
DFF	3,269	4,521	5,586	6,659	6,724	6,931
GH-1	3,898	5,439	5,999	6,513	6,902	7,231
GH-2	3,887	5,291	5,721	6,017	6,313	6,502

Figure 11: Satisfaction reached by routing policies given a maximum number of hops (second scenario)

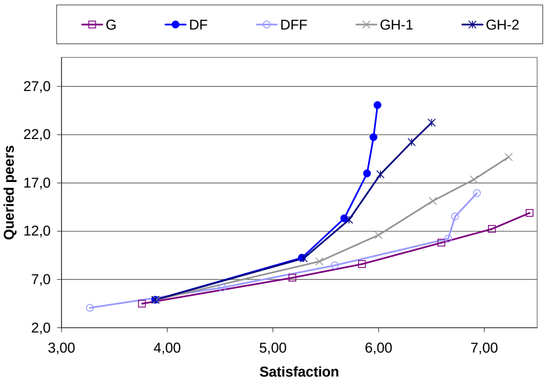


Figure 12: Effectiveness vs. efficiency of routing policies (second scenario)

6. REFERENCES

[1] K. Aberer, P. Cudré-Mauroux, M. Hauswirth, and T. V. Pelt. GridVine: Building Internet-Scale Semantic Overlay Networks. In *Proc. of ISWC*, 2004.

[2] M. Arenas, V. Kantere, A. Kementsietsidis, I. Kiringa, R. Miller, and J. Mylopoulos. The Hyperion Project: from Data Integration to Data Coordination. *SIGMOD Record*, 32(3):53–58, 2003.

[3] D. Aumueller, H. H. Do, S. Massmann, and E. Rahm. Schema and Ontology Matching with COMA++. In *SIGMOD Conference*, pages 906–908, 2005.

[4] T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, May 2001.

[5] A. Crespo and H. Garcia-Molina. Routing Indices for Peer-to-Peer Systems. In *Proc. of ICDCS*, 2002.

[6] A. Crespo and H. Garcia-Molina. Semantic Overlay Networks for P2P Systems. In *Proc. of the 3rd AP2PC Workshop*, pages 1–13, 2004.

[7] H. Do and E. Rahm. COMA – A System for Flexible Combination of Schema Matching Approaches. In *Proc. of the 28th VLDB*, pages 610–621, 2002.

[8] C. Doulkeridis, K. Nørvåg, and M. Vazirgiannis. DESENT: Decentralized and Distributed Semantic Overlay Generation in P2P Networks. *IEEE J. on Selected Areas in Comm.*, 25(1):25–34, 2007.

[9] A. Halevy, Z. Ives, J. Madhavan, P. Mork, D. Suciu, and I. Tatarinov. The Piazza Peer Data Management System. *IEEE TKDE*, 16(7):787–798, 2004.

[10] M. A. Hernández, R. J. Miller, and L. M. Haas. Clio: A Semi-Automatic Tool For Schema Mapping. In *SIGMOD Conference*, page 607, 2001.

[11] G. Koloniari and E. Pitoura. Content-Based Routing of Path Queries in Peer-to-Peer Systems. In *Proc. of the 9th EDBT Conf.*, pages 29–47, 2004.

[12] M. Li, W.-C. Lee, and A. Sivasubramaniam. Semantic Small World: An Overlay Network for Peer-to-Peer Search. In *Proc. of the 12th IEEE ICNP*, 2004.

[13] S. Lodi, F. Mandreoli, R. Martoglia, W. Penzo, and S. Sassatelli. Semantic Peer, Here are the Neighbors You Want! In *Accepted for publication at the 11th EDBT Conf.*, 2008.

[14] J. Madhavan, P. A. Bernstein, and E. Rahm. Generic Schema Matching with Cupid. In *Proc. of the 27th VLDB*, pages 49–58, 2001.

[15] F. Mandreoli, R. Martoglia, W. Penzo, and S. Sassatelli. SRI: Exploiting Semantic Information for Effective Query Routing in a PDMS. In *Proc. of the WIDM (in conj. with CIKM)*, pages 19–26, 2006.

[16] F. Mandreoli, R. Martoglia, W. Penzo, S. Sassatelli, and G. Villani. SRI@work: Efficient and Effective Routing Strategies in a PDMS. In *Proc. of WISE*, 2007.

[17] F. Mandreoli, R. Martoglia, W. Penzo, S. Sassatelli, and G. Villani. SUNRISE: Exploring PDMS Networks with Semantic Routing Indexes. In *Proc. of ESWC*, 2007.

[18] F. Mandreoli, R. Martoglia, and E. Ronchetti. Versatile Structural Disambiguation for Semantic-aware Applications. In *Proc. of CIKM*, 2005.

[19] F. Mandreoli, R. Martoglia, and E. Ronchetti. STRIDER: a Versatile System for Structural Disambiguation. In *Proc. of EDBT*, 2006.

[20] F. Mandreoli, R. Martoglia, and P. Tiberio. Approximate Query Answering for a Heterogeneous XML Document Base. In *Proc. of WISE*, 2004.

[21] S. Melnik, H. Garcia-Molina, and E. Rahm. Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching. In *Proc. of the 18th ICDE*, 2002.

[22] R. Miller, L. Haas, and M. Hernández. Schema Mapping as Query Discovery. In *Proc. of VLDB*, 2000.

[23] 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 *Proc. of the 11th WWW Conf.*, 2002.

[24] J. Parreira, S. Michel, and G. Weikum. P2PDating: Real Life Inspired Semantic Overlay Networks for Web Search. *Inf. Proc. & Manag.*, 43(3):643–664, 2007.

[25] I. Tatarinov and A. Halevy. Efficient Query Reformulation in Peer Data Management Systems. In *Proc. of the 2004 ACM SIGMOD Int. Conf. on Management of Data (SIGMOD 2004)*, 2004.

[26] C. Yu and H. Jagadish. Schema Summarization. In *Proc. of the 32nd VLDB Conf.*, pages 319–330, 2006.