
Efficient Query Routing in RDF/S schema based P2P Systems

HDMS 2005

Lefteris Sidirouros, George Kokkinidis
Institute of Computer Science - FORTH
{lsidir,kokkinid}@ics.forth.gr

Theodore Dalamagas
School of Electr. and Comp. Engineering
National Technical University of Athens
dalamag@dblabb.ece.ntua.gr

1

Schema-Based P2P Systems

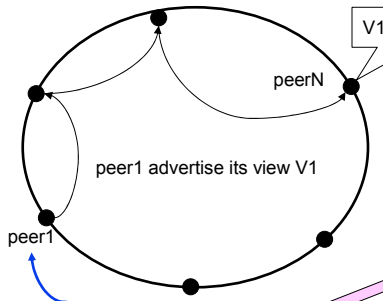
- In schema-based P2P systems, joining peers advertise their actual data using the schema of their bases
- Capturing explicitly the semantics of data bases available in the network using a schema enables us to:
 - ◆ support expressive queries on (semi-)structured data
 - ◆ deploy effective methods for locating remote peers that can answer these queries
 - ◆ build efficient distributed query processing mechanisms

the main challenge in this setting is to build an effective and efficient lookup service for identifying, in a complete decentralized fashion, which peer views can actually contribute to the answer of a specific query

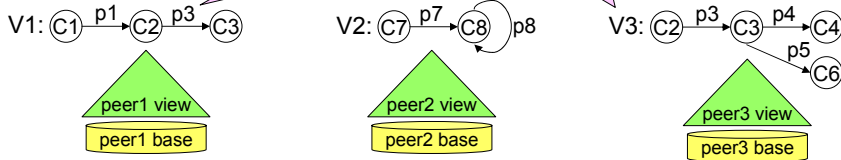
2

Our Framework

Chord Ring: DHT Index of view advertisement



- Build on top:
- large scale collaborative applications:
 - ◆ no centralized warehouse
 - ◆ no unlimited data migration
 - sophisticated data management services:
 - ◆ query processing
 - ◆ load balance
 - ◆ data replication



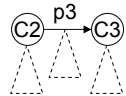
RDF/S Query/View Subsumption

Queries are formulated using an RDF/S query language (e.g., RQL)

query pattern:
(x, p3,y)

a pattern specifies the **fragment** of the RDF/S graph which is involved in the evaluation of the query

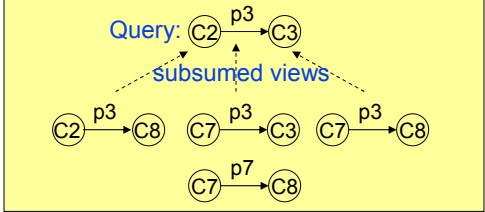
query fragment:



Views specify **fragments** of the RDF/S graph too

view fragment:
C1 -p1-> C2 -p3-> C3

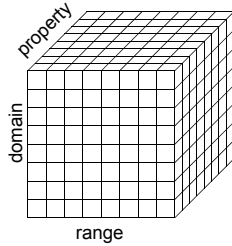
Views that are **subsumed** by the query contribute to the answer:



How views are placed on the Chord Ring in order to efficiently decide view/query subsumption?

Encoding RDF/S fragments

AdjSub Cube

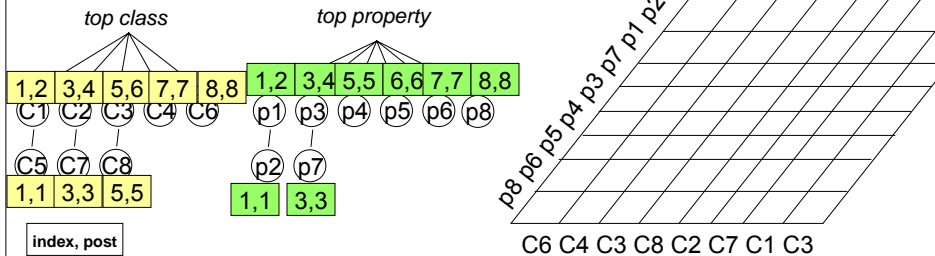


- The AdjSub Cube is a **succinct representation** of RDF/S schema graphs allowing to check whether a graph fragment is subsumed by another
 - ◆ each cell represents a **schema triple**; each schema triple is assigned a number through the function **pos()**, according to the **position of the corresponding cell in the cube**; the encoding of an RDF/S schema fragment of n schema triples st_n, \dots, st_1 is the **unique number** $N = 2^{\text{pos}(st_n)} + \dots + 2^{\text{pos}(st_1)}$ or equivalent the set $\{\text{pos}(st_n), \dots, \text{pos}(st_1)\}$
- Does not need to be implemented, instead it is used to derive an encoding function of **RDF/S schema fragments**

5

Encoding RDF/S fragments

Subsumption Hierarchies



- AdjSub Cube relies on an interval-based encoding of subsumption hierarchies
 - ◆ imposes a **total ordering of classes and properties** on each dimension based on the post-order enumeration
 - ◆ places subsuming views on the Chord Ring sequentially to facilitate the lookup service

6

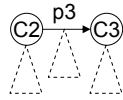
Query Routing

- The sequence in which the subsumed views are looked up is given by the AdjSub Cube
- ◆ this sequence is important since there must be no lookups that address preceding peers in order to avoid costly jumps over the Chord ring

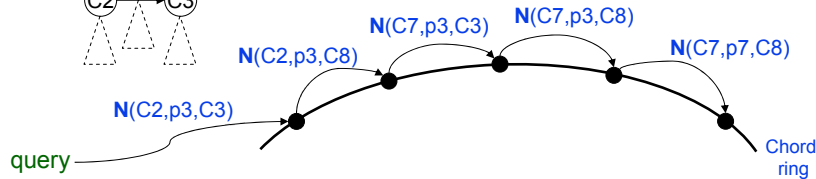
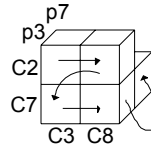
example:

Query Pattern:
(x, p3,y)

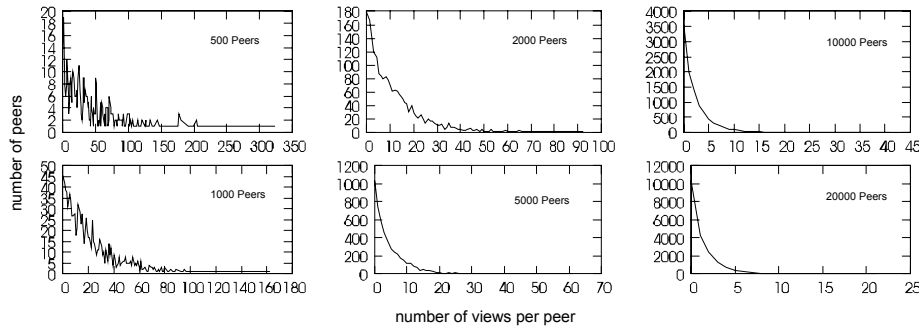
RDF/S graph fragment:



AdjSub Cube traversal for the RDF/S fragment:



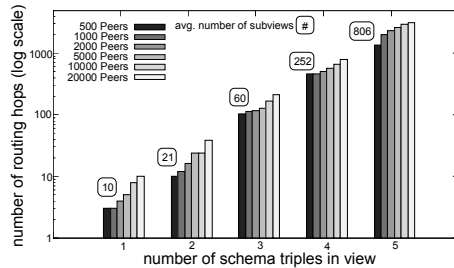
Experimental Evaluation – DHT Index



- views are stored **uniformly** over the Chord Ring
- as the network grows, views are distributed in more peers thus there is no introduction of hot-spots and the index **scales gracefully**

Experimental Evaluation - Lookup Service

Lookup Service



- increasing the number of schema triples of a view implies more routing hops, because:

- ◆ there exist additional subsumed views, and
- ◆ for large views, subsumed views are distant in the Chord ring

- less than $S \times \log(N)$ hops

- ◆ ex. for views with 3 schema triples:

$$S = 60, N = 20000, \log(N) = 14 : 60 \times 14 = 840$$

but, experimental evaluation shows no more than 350 hops

9

Conclusions & Future Work

- a DHT-based framework to efficiently route expressive RDF/S queries
- introduced a succinct representation of RDF/S schema graphs that ensures fast query/view subsumption checking
- designed a distributed lookup service
- experimentally evaluated our framework to demonstrate that it scales gracefully

Future Work:

- ◆ extend with distributed query planning capabilities
- ◆ interleaved execution of routing/planning phases
- ◆ distributed trees?

10

Thank you!