ViP2P: XML Warehousing in P2P Networks

Ioana Manolescu-Goujot

Equipe Leo
INRIA Saclay–Île-de-France & LRI, Université de Paris Sud-11

Supported by: ANR-08-DEFIS-004 CODEX, INRIA ADT ViP2P & ViRDF
Leo and friends in front of our lab
Part I

Introduction
Distributed data management: old goal (1970)
Distributed data management: old goal (1970)

- distributed versions of industrial-strength DBMSs
- map/reduce style systems for massively parallel computations
Motivation

Distributed data management: old goal (1970)
- distributed versions of industrial-strength DBMSs
- map/reduce style systems for massively parallel computations

Still missing: the flexible federation
- high independence of the sites: when to be in, what to store
- data distribution transparency
- ... with the usual performance requirements
Motivation

Distributed warehouses of Web content

Web content
structured documents, schemas, annotations, concepts, mappings, Web services, inter-document links

Web content warehouse
Distributed database of selected content, whose users may:
- publish resources
- connect (annotate, map, link...) existing resources
- update resources
- enhance resources by combining them

In the style of the RNTL WebContent project (2005-2009)
INRIA annual activity report in XML

https://irabot.inrialpes.fr/RA2009/gemo.xml

<?xml version="1.0" encoding="iso-8859-1"?>
<!DOCTYPE raweb PUBLIC "raweb2.dtd">
<raweb year="2009">
  <identification id="gemo" isproject="true">
    <shortname>gemo</shortname>
    <domaine>Perception, Cognition, Interaction</domaine>
    <UR name="Saclay"/>
  </identification>
  <team>
    <person>
      <web>http://www-roc.inria.fr/~manolesc</web>
      <firstname>Ioana</firstname>
      <lastname>Manolescu</lastname>
    </person>...
    </team>...
  </raweb>
Motivation

INRIA annual activity report in XML

(Very) popular data exchange format
Many W3C standards + efficient tools
XML and the Semantic Web

XML

<table>
<thead>
<tr>
<th>identification</th>
<th>team</th>
</tr>
</thead>
<tbody>
<tr>
<td>@id</td>
<td>@isproject</td>
</tr>
<tr>
<td>gemo</td>
<td>true</td>
</tr>
<tr>
<td>Domaine</td>
<td>UR</td>
</tr>
<tr>
<td>Perception, Cognition, Interaction</td>
<td>name</td>
</tr>
<tr>
<td>Saclay</td>
<td>web</td>
</tr>
<tr>
<td>firstname</td>
<td>lastname</td>
</tr>
<tr>
<td>Ioana</td>
<td>Ioana Manolescu</td>
</tr>
<tr>
<td>www-roc.inria.fr/~manolesc</td>
<td><a href="http://www.inria.fr/types.xsd#ContratFR">http://www.inria.fr/types.xsd#ContratFR</a></td>
</tr>
</tbody>
</table>

RDF

<table>
<thead>
<tr>
<th><a href="http://gemo.saclay.inria.fr">http://gemo.saclay.inria.fr</a></th>
<th>wroteReport</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="https://irabot.inrialpes.fr/RA2009/gemo.xml">https://irabot.inrialpes.fr/RA2009/gemo.xml</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><a href="http://gemo.saclay.inria.fr">http://gemo.saclay.inria.fr</a></th>
<th>coordinates</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://codex.saclay.inria.fr">http://codex.saclay.inria.fr</a></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><a href="http://codex.saclay.inria.fr">http://codex.saclay.inria.fr</a></th>
<th>hasType</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://www.inria.fr/types.xsd#ContratFR">http://www.inria.fr/types.xsd#ContratFR</a></td>
<td></td>
</tr>
</tbody>
</table>
Motivation

XML and the Semantic Web

Possible queries

- European projects involving at least two INRIA teams?
- Most frequent keywords for each INRIA theme?
- PhD students co-supervised with a foreign University?
Distributed warehouses of Web content

Distributed XML

https://irabot.inrialpes.fr/RA2009/gemo.xml

raweb

identification

@id @isproject domaine UR

gemo true

Perception, Cognition, Interaction

name web firstname lastname

www-roc.inria.fr/~manolesc

http://gemo.saclay.inria.fr

http://codex.saclay.inria.fr

Distributed RDF


INRIA

http://pierre.senellart.com

collaboratesWith

https://gemo.saclay.inria.fr

Indiana Univ.

http://pierre.senellart.com

coordinates

dbweb.enst.fr/events/sigmod10contest/

Splitted Desktop

http://gemo.saclay.inria.fr

possibleCollaborator

http://www.splitted-desktop.com/fr/
Distributed warehouses of Web content

Distributed XML
https://irabot.inrialpes.fr/RA2009/gemo.xml

raweb
identification
@gemoid @isproject
@gemot @isproject
domaine
Perception, Cognition, Interaction
UR
www-roc.inria.fr/~manolesc

Distributed RDF

Serv. Inf. Sci. Tech. Université

http://pierre.senellart.com

collaboratesWith INRIA

http://gemo.saclay.inria.fr

Indiana

http://pierre.senellart.com

coordinates

http://gemo.saclay.inria.fr/events/sigmod10contest/

Splitted Desktop

http://gemo.saclay.inria.fr

possibleCollaborator

http://www.splitted-desktop.com/fr/

French researchers involved in ACM conference organization?
Possible collaborators for a data management project in the clouds?
Distributed hash tables
Distributed hash tables
Distributed hash tables
Distributed hash tables

\[ \text{put}(k_1, v_1) \]
Distributed hash tables

\[ \text{put}(k_1, v_1) \]
Distributed hash tables

\[ (k_1, v_1) \]

\[ \text{put}(k_1, v_1) \]

\[ \text{put}(k_1, v_2) \]
Distributed hash tables

\( (k_1, \{ v_1, v_2 \}) \)

- \( p_1 \) to \( p_3 \) to \( p_5 \) to \( p_2 \) to \( p_3 \) to \( p_6 \) to \( p_7 \) to \( p_8 \) to \( p_1 \)

- \( p_8 \rightarrow \text{put}(k_1, v_1) \)

- \( p_6 \rightarrow \text{put}(k_1, v_2) \)
Distributed hash tables

\[
(k_1, \{v_1, v_2\})
\]

- \text{put}(k_1, v_1)
- \text{put}(k_1, v_2)
- \text{get}(k_1)
Distributed hash tables

\[
(k_1, \{v_1, v_2\})
\]

- \(p_1\), \(p_2\), \(p_3\), \(p_4\), \(p_5\), \(p_6\), \(p_7\), \(p_8\)

- \(\text{put}(k_1, v_1)\)
- \(\text{put}(k_1, v_2)\)
- \(\text{get}(k_1)\)
Distributed hash tables

\begin{align*}
\text{put}(k_1, v_1) \\
\text{put}(k_1, v_2) \\
(k_1, \{v_1, v_2\}) \\
\text{get}(k_1) \\
\{v_1, v_2\}
\end{align*}
What distributed hash tables provide

Dynamic peer networks
- each peer is assigned an id $\Rightarrow$ address range
- bound of $\log_2(N)$ hops to route a message to peer
- network re-adjustment when peers join or leave: peers’ address spaces stretch and contract
What distributed hash tables provide

Dynamic peer networks
- each peer is assigned an id $\Rightarrow$ address range
- bound of $\log_2(N)$ hops to route a message to peer
- network re-adjustment when peers join or leave: peers’ address spaces stretch and contract

(Key, value) stores = basis for content sharing
- index the resources by keys
- look up resources by keys
What distributed hash tables provide

Dynamic peer networks

- each peer is assigned an id ⇒ address range
- bound of $\log_2(N)$ hops to route a message to peer
- network re-adjustment when peers join or leave: peers’ address spaces stretch and contract

(Key, value) stores = basis for content sharing

- index the resources by keys
- look up resources by keys

Main DHT functionality

Small shared memory ⇒ resource catalog
From DHTs to distributed data management

Functionalities to add

- data indexing
- storage for application data and even DHT index data
- local query processing
- distributed query processing: operators, including data transfers, optimization . . .
From DHTs to distributed data management

Functionalities to add

- data indexing
- storage for application data and even DHT index data
- local query processing
- distributed query processing: operators, including data transfers, optimization . . .

Reliability

- provided a peer will eventually answer at each address some (key, value) replication to handle peer failures (broadcast to $k$ replicas)
- to do resilience of all added functionalities to peer joining/leaving/failing
Querying distributed data on a DHT

Two types of operations:

1. Look up useful data sources in the DHT
   - Documents matching the query
   - Documents which *might* match the query
Two types of operations:

1. Look up useful data sources in the DHT
   - Documents matching the query
   - Documents which *might* match the query

DHT indexing is important here
Querying distributed data on a DHT

Two types of operations:

1. Look up useful data sources in the DHT
   - Documents matching the query
   - Documents which *might* match the query

   DHT indexing is important here

Index building and maintenance
Querying distributed data on a DHT

Two types of operations:

1. Look up useful data sources in the DHT
   - Documents matching the query
   - Documents which *might* match the query
   DHT indexing is important here
   Index building and maintenance

2. Remaining query evaluation steps
   - **evaluate** sub-queries to extract partial results
   - **combine** partial results
   - **transfer** results to query peer
Querying distributed data on a DHT

Two types of operations:

1. Look up useful data sources in the DHT
   - Documents matching the query
   - Documents which *might* match the query

   **DHT indexing is important here**
   - Index building and maintenance

2. Remaining query evaluation steps
   - **evaluate** sub-queries to extract partial results
   - **combine** partial results
   - **transfer** results to query peer

   **Distributed query processing “as usual”**
Trade-offs in DHT indexing and query processing

Level of detail of resource indexing
- lookup precision $\uparrow \Rightarrow$ execution time $\downarrow$
- data publication time $\uparrow$, possibly execution time $\uparrow$

Data re-placement or clustering
- fewer peers contacted for a query (message no. $\downarrow$, execution time $\uparrow$)
- data transfers in the absence of queries (message no. $\uparrow$, total message size $\uparrow$)
Part II

ViP2P: materialized views on DHTs
5 Credits

6 ViP2P overview

7 ViP2P views

8 ViP2P rewrites

9 The distributed platform
Joint work with

Students
- Konstantinos Karanasos, Asterios Katsifodimos, Spyros Zoupanos (PhD 2009)
- Martin Goodfellow (U. Strathclyde), Silviu Julean (UVT, Romania), Varunesh Mishra (IIT Kanpur), Alexandra Roatis (UVT, Romania)

Engineers
- Jesús Camacho-Rodriguez, Julien Leblay, Alin Tilea

Friends from abroad
- Angela Bonifati (CNR Italy), Vasilis Vassalos (AUEB, Greece)
Web data sharing in ViP2P

Web data:
- XML documents
- RDF annotations
Web data sharing in ViP2P

Web data:
- XML documents
- RDF annotations

Choices:
- peers retain control over the data they store/publish
- no global schema
- documents published independently
- annotations (triples) can freely connect content
- content is re-distributed: peers may accumulate results of long-running queries
- peers share their accumulated results with others
Web data sharing in ViP2P

Web data:
- XML documents
- RDF annotations

Choices:
- peers retain control over the data they store/publish
  - no global schema
  - documents published independently
  - annotations (triples) can freely connect content
- content is re-distributed: peers may accumulate results of long-running queries
- peers share their accumulated results with others
Behind the scene: distributed materialized views
Behind the scene: distributed materialized views

Long-running query = subscription = view

Declarative specification of results that the peer is interested in

- tree pattern $\Rightarrow$ matches an individual document
- tree patterns with value joins $\Rightarrow$ matches across distinct documents
- corresponding XQuery dialect

Data publication = incremental update of the view

Easier for tree pattern views

Query planning = rewriting queries using views

Find relevant views $\Rightarrow$ rewrite the query $\Rightarrow$ optimize the rewriting

Query execution = distributed query processing

With XML- and RDF-specific bells & whistles
Behind the scene: distributed materialized views

Long-running query = subscription = view

Declarative specification of results that the peer is interested in

- tree pattern $\Rightarrow$ matches an individual document
- tree patterns with value joins $\Rightarrow$ matches across distinct documents
- corresponding XQuery dialect
- flexibility
Behind the scene: distributed materialized views

Long-running query = subscription = view

Declarative specification of results that the peer is interested in

- tree pattern $\Rightarrow$ matches an individual document
- tree patterns with value joins $\Rightarrow$ matches across distinct documents
- corresponding XQuery dialect
- flexibility

Data publication = incremental update of the view

Easier for tree pattern views
Behind the scene: distributed materialized views

Long-running query = subscription = view

Declarative specification of results that the peer is interested in
- tree pattern $\Rightarrow$ matches an individual document
- tree patterns with value joins $\Rightarrow$ matches across distinct documents
- corresponding XQuery dialect
- flexibility

Data publication = incremental update of the view

Easier for tree pattern views

Query planning = rewriting queries using views

Find relevant views $\Rightarrow$ rewrite the query $\Rightarrow$ optimize the rewriting
Behind the scene: distributed materialized views

Long-running query = subscription = view
Declarative specification of results that the peer is interested in
  - tree pattern ⇒ matches an individual document
  - tree patterns with value joins ⇒ matches across distinct documents
  - corresponding XQuery dialect
  - flexibility

Data publication = incremental update of the view
Easier for tree pattern views

Query planning = rewriting queries using views
Find relevant views ⇒ rewrite the query ⇒ optimize the rewriting

Query execution = distributed query processing
With XML- and RDF-specific bells & whistles
Publishing and querying in ViP2P
The peers may store:
- documents,
- annotations
The peers may store:

- documents, annotations
- views
Publishing and querying in ViP2P

When $q$ arrives:
Publishing and querying in ViP2P

When $q$ arrives:
- view definition lookup
Publishing and querying in ViP2P

When $q$ arrives:
- view definition lookup
- rewriting
Publishing and querying in ViP2P

When \( q \) arrives:
- view definition
- lookup
- rewriting
- execution of physical plan
Publishing and querying in ViP2P

When \( d \) arrives:

\[
\text{search view definitions for which } v_i(d) \neq \emptyset
\]

compute \( v_i(d) \)

send results
When $d$ arrives:

- search view definitions for which $v_i(d) \neq \emptyset$
Publishing and querying in ViP2P

When $d$ arrives:

- search view definitions for which $v_i(d) \neq \emptyset$
- compute $v_i(d)$
Publishing and querying in ViP2P

When \( d \) arrives:
- search view definitions for which \( v_i(d) \neq \emptyset \)
- compute \( v_i(d) \)
- send results
**All article abstracts**

<table>
<thead>
<tr>
<th>URI</th>
<th>abstract</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="http://peer1.com/doc1.xml">http://peer1.com/doc1.xml</a></td>
<td>Databases…</td>
</tr>
<tr>
<td><a href="http://peer2.com/f2.xml">http://peer2.com/f2.xml</a></td>
<td>XML processing…</td>
</tr>
<tr>
<td>…</td>
<td>…</td>
</tr>
</tbody>
</table>
Sample views

Titles and abstracts of all articles

article

abstract  title

<table>
<thead>
<tr>
<th>URI</th>
<th>abstract</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Sample views

Titles of all articles having “P2P” in the abstract

```
<table>
<thead>
<tr>
<th>URI</th>
<th>title</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
```
Sample views

URIs of all documents containing articles having “P2P” in the abstract
Sample views

IDs and authors of all articles having “P2P” in the abstract

<table>
<thead>
<tr>
<th>URI</th>
<th>article.ID</th>
<th>author.val</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

Diagram:

```
article_ID
  abstract
  author_val
    P2P
```
Sample views

Homepages of all WebDam summer school organizers

![Diagram showing relationships between school, organizers,homepage, lab, name, URI1, URI2, homepage.cont]

<table>
<thead>
<tr>
<th>URI1</th>
<th>URI2</th>
<th>homepage.cont</th>
</tr>
</thead>
<tbody>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>
Sample rewritings

Views

\[ V_1 \]

\[ \text{article} \_ID \]

\[ \text{title} \_val \]

\[ \text{author} \_val \]

\[ V_2 \]

\[ \text{article} \_ID \]

Query

\[ \text{article} \]

\[ \text{title} \_val \]

\[ \text{author} \_val \]

Rewriting

\[ \nabla \text{article} \_ID \]

\[ V_1 \]

\[ V_2 \]
Sample rewritings

Views

$v_1$

proceedings

$v_2$

proceedings

article_{ID}

title_{ID, val}

author_{val}

Query

proceedings

article

title_{val}

author_{val}

Rewriting

▷ article_{ID} ◁

title_{ID}

$v_1$

$v_2$
Sample rewritings

Views

Query

Rewriting

$v$

proceedings

$v$

proceedings$_{cont}$

article

$v$

title$_{val}$

author$_{val}$

$v$

nav

article

$v$

title$_{val}$

author$_{val}$
Sample physical rewriting plan

project@
bordereau-25.bordeaux.grid5000.fr

receive@
bordereau-25.bordeaux.grid5000.fr

hashJoin@
pastel-79.toulouse.grid5000.fr

scan(4nodesView)@
pastel-79.toulouse.grid5000.fr

receive@
pastel-79.toulouse.grid5000.fr

scan(3nodesView)@
griffon-92.nancy.grid5000.fr
ViP2P platform

- 70,000 Java lines (330 classes) and decreasing :)
- Uses Berkeley DB (version 3.3.75) to store view data
- Uses FreePastry (version 2.1) as our DHT network
- Experiments carried on Grid5000 using 250 machines
- Deployed on 1000 ViP2P peers
ViP2P platform

- 70,000 Java lines (330 classes) and decreasing :)
- Uses Berkeley DB (version 3.3.75) to store view data
- Uses FreePastry (version 2.1) as our DHT network
- Experiments carried on Grid5000 using 250 machines
- Deployed on 1000 ViP2P peers

Algorithms inside:
- SAX-based custom processor for computing $v(d) +$ identifiers
- Iterator-based execution engine, (nested) tuples
- Send/receive for distributed execution + URI compression
- Many query optimization strategies
Indexing views for view materialization

View indexing

For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$
Indexing views for view materialization

View indexing

For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$

View lookup

For each document $d$ and node label $l$ in $d$, get($l$)
Indexing views for view materialization

View indexing
For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$

View lookup
For each document $d$ and node label $l$ in $d$, get($l$)
Filter views thus obtained:

\[
\text{if } v \text{ uses a tag/keyword not present in } d \\
\text{then } \text{discard } v
\]
Indexing views for view materialization

View indexing
For each view \( v \) and node label \( l \) in \( v \), put \( (l, v) \)
For each view \( v \) and constant \( k \) in \( v \), put \( (k, v) \)

View lookup
For each document \( d \) and node label \( l \) in \( d \), get(\( l \))
Filter views thus obtained:

\[
\text{if } v \text{ uses a tag/keyword not present in } d \\
\text{then } \text{discard } v
\]

Evaluate the filtered view set on \( d \)
View materialization

View + documents = few results

![Graph showing extraction time, BDB storage time, and data exchange time as functions of document size.](image-url)
View materialization

View + documents = many results

Extraction Time

BDB Storage Time

Data exchange Time

Size of Document (MB)

Extraction time (ms)
Indexing views for query rewriting

View indexing

For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$
Indexing views for query rewriting

**View indexing**

For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$

**View lookup**

For each query $q$ and label $l$ in $q$, get($l$)
For each query $q$ and constant $k$ in $q$, get ($k$)
Indexing views for query rewriting

View indexing
For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$

View lookup
For each query $q$ and label $l$ in $q$, get($l$)
For each query $q$ and constant $k$ in $q$, get ($k$)
Filter the views thus obtained:

if $v$ cannot be embedded in $q$
then discard $v$
Indexing views for query rewriting

View indexing

For each view $v$ and node label $l$ in $v$, put $(l, v)$
For each view $v$ and constant $k$ in $v$, put $(k, v)$

View lookup

For each query $q$ and label $l$ in $q$, get($l$)
For each query $q$ and constant $k$ in $q$, get ($k$)
Filter the views thus obtained:

$$\text{if } v \text{ cannot be embedded in } q$$
$$\text{then } \text{discard } v$$

Rewrite $q$ with the filtered view set
View look up performance

1440 views related to a 32-nodes query $q$

**View definitions retrieved**

<table>
<thead>
<tr>
<th>View indexing strategy</th>
<th>Number of views</th>
</tr>
</thead>
<tbody>
<tr>
<td>LI</td>
<td>1500</td>
</tr>
<tr>
<td>RLI</td>
<td>1400</td>
</tr>
<tr>
<td>LPI</td>
<td>1200</td>
</tr>
<tr>
<td>RPI</td>
<td>1100</td>
</tr>
</tbody>
</table>

- **View definitions retrieved**
- **Useful view definitions**
View rewriting performance

We find **minimal** rewritings
NP-hard problem; heuristics for finding some rewriting fast
The interest of view-based rewriting

Almost-XMark queries

![Bar chart showing execution time (ms) for direct views and views]

<table>
<thead>
<tr>
<th>Query</th>
<th>Execution Time (ms)</th>
</tr>
</thead>
<tbody>
<tr>
<td>TPQ1</td>
<td>10000</td>
</tr>
<tr>
<td>TPQ2</td>
<td>10000</td>
</tr>
<tr>
<td>TPQ3</td>
<td>10000</td>
</tr>
<tr>
<td>TPQ4</td>
<td>10000</td>
</tr>
<tr>
<td>JTPQ1</td>
<td>12000</td>
</tr>
<tr>
<td>JTPQ2</td>
<td>12000</td>
</tr>
<tr>
<td>JTPQ3</td>
<td>12000</td>
</tr>
<tr>
<td>JTPQ4</td>
<td>12000</td>
</tr>
</tbody>
</table>

Ioana Manolescu (Leo, Saclay)
Part III

Summary
We have
A distributed engine for sharing XML data based on materialized views
Perspectives

We have
A distributed engine for sharing XML data based on materialized views

We are working on

1. XML+RDF management (K. Karanasos, F. Goasdoué)
3. Incremental view maintenance (M. Goodfellow, A. Bonifati)
4. Query optimization (A. Roatis)
We have
A distributed engine for sharing XML data based on materialized views

We are working on
1. XML+RDF management (K. Karanasos, F. Goasdoué)
3. Incremental view maintenance (M. Goodfellow, A. Bonifati)
4. Query optimization (A. Roatis)

Blue sky
1. Pure RDF management in P2P
2. Moving the back-end into the clouds (EnergieNet)
Closest related works

DHT-based sharing of relations [LHSH04]
DHT-based XML indexing [GWJD03, BC06, SHA05, AMP+08]
DHT-based shared XML caches [LP08]
XPath query rewriting [BOB+04, XO05, CDO08, TYÖ+08]
  - XPath: wildcard *, union
  - Rewritings: intersection, navigations, joins

Rewriting with structural constraints [ABMP07]
  - Centralized setting
  - Dataguide [GW97] constraints

Layered architecture for Web content warehousing [AAC+08]
RDF querying and reasoning on DHT [KMK08, LIK06]
Thank you!

http://vip2p.saclay.inria.fr


[BC06] Angela Bonifati and Alfredo Cuzzocrea.
Storing and retrieving XPath fragments in structured P2P networks.
*Data Knowl. Eng.*, 59(2), 2006.

**[BOB+04]** A. Balmin, F. Ozcan, K. Beyer, R. Cochrane, and H. Pirahesh.
A framework for using materialized XPath views in XML query processing.

**[CDO08]** Bogdan Cautis, Alin Deutsch, and Nicola Onose.
XPath rewriting using multiple views: Achieving completeness and efficiency.

**[GW97]** Roy Goldman and Jennifer Widom.
Dataguides: Enabling query formulation and optimization in semistructured databases.


<table>
<thead>
<tr>
<th>Reference</th>
<th>Authors</th>
<th>Title</th>
<th>Conference</th>
<th>Pages</th>
<th>Year</th>
</tr>
</thead>
<tbody>
<tr>
<td>[SHA05]</td>
<td>Gleb Skobeltsyn, Manfred Hauswirth, and Karl Aberer.</td>
<td>Efficient processing of XPath queries with structured overlay networks.</td>
<td>OTM Conferences (2)</td>
<td></td>
<td>2005</td>
</tr>
<tr>
<td>[XO05]</td>
<td>W. Xu and M. Ozsoyoglu.</td>
<td>Rewriting XPath queries using materialized views.</td>
<td>VLDB</td>
<td></td>
<td>2005</td>
</tr>
</tbody>
</table>