Viewing a World of Annotations through AnnoVIP

K. Karanasos  I. Manolescu  S. Zoupanos

Leo group
INRIA Saclay–Île-de-France and LRI, Université de Paris Sud-11

Codex Project Meeting, 19/3/2010
Overview

- Distributed Hash Tables (DHT)
- ViP2P
- AnnoVIP

We are not alone!

Asterios Katsifodimos
Jesús Camacho-Rodríguez
Alin Tilea
Martin Goodfellow
Overview

- Distributed Hash Tables (DHT)
- ViP2P
- AnnoVIP
Distributed hash tables
Distributed hash tables

put\((k_1, v_1)\)
Distributed hash tables

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

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

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

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

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

- put($k_1, v_1$)
- put($k_1, v_2$)
- get($k_1$)

$p_1 \rightarrow p_2 \rightarrow p_3 \rightarrow p_4 \rightarrow p_5 \rightarrow p_6 \rightarrow p_7 \rightarrow p_8$

$(k_1, \{v_1, v_2\})$
Distributed hash tables

get($k_1$) → put($k_1, v_1$) → p_1 → p_2 → p_3 → (k_1, {v_1, v_2}) → p_4 → p_5 

put($k_1, v_1$) → put($k_1, v_2$) → p_6 → p_7 → p_8
Distributed hash tables

- put($k_1, v_1$)
- put($k_1, v_2$)
- get($k_1$)
- ({$v_1, v_2$})
Overview

- Distributed Hash Tables (DHT)
- ViP2P
- AnnoVIP
ViP2P: views in peer-to-peer
ViP2P: views in peer-to-peer

The peers may store: documents
ViP2P: views in peer-to-peer

The peers may store:
- documents
- views
ViP2P: views in peer-to-peer

View materialization
ViP2P: views in peer-to-peer

View materialization

When $d$ arrives:

1. Search view definitions for which $v_i(d) \neq \emptyset$
2. Compute $v_i(d)$
3. Send results
ViP2P: views in peer-to-peer

View materialization

When $d$ arrives:
- search view definitions for which $v_i(d) \neq \emptyset$
View materialization

When $d$ arrives:
- search view definitions for which $v_i(d) \neq \emptyset$
- compute $v_i(d)$
ViP2P: views in peer-to-peer

View materialization

When $d$ arrives:

- search view definitions for which $v_i(d) \neq \emptyset$
- compute $v_i(d)$
- send results
ViP2P: views in peer-to-peer

Query answering
ViP2P: views in peer-to-peer

Query answering

When $q$ arrives:
ViP2P: views in peer-to-peer

Query answering

When $q$ arrives:
- view definition lookup
ViP2P: views in peer-to-peer

Query answering

When $q$ arrives:
- view definition lookup
- rewriting
ViP2P: views in peer-to-peer

Query answering

When $q$ arrives:
- view definition lookup
- rewriting
- execution of physical plan
Representing our queries and views.

- Each pattern node carries a label (element name or attribute name or word).
- Each pattern edge correspond to a /- or a ///</-relationship between nodes.
- A node may be annotated with zero or more among the following labels: id, cont and val.
- A node may be annotated with a predicate of the form \([val = c]\).
Let $q$ be a query and $\mathcal{V} = \{v_1, v_2, \ldots, v_k\}$ a set of views. A **rewriting** of $q$ using $\mathcal{V}$ is an algebraic expression $e(v_1, v_2, \ldots, v_k)$ such that $e(D) = q(D)$ for any document set $D$.

**Algebra operators**

- $\text{scan}(v)$
- $\pi_{\text{cols}}(op)$
- $\text{sort}_{\text{cols}}(op)$
- $\text{nav}_{i, np}(op)$ evaluates $np$ over the $\text{cont}$ attribute $op.i$
ViP2P: Rewriting example

$q$

$a$

$b$

$c_{\text{cont}}$

$e_{\text{val}}$

$d_{[\text{val}=5]}$

$v_1$

$v_2$

$v_3$

$a_{id}$

$b_{id,\text{cont}}$

$c_{id,\text{cont}}$

$a_{id}$

$d_{\text{val}}$
ViP2P: Rewriting example

\[ q \]

\[ \sigma_{a.id = a.id \land b.id < c.id \land a.id < b.id \land d.val = 5} \]

\[ \pi_{c.cont, e.val} \]

\[ \text{nav}_{b.cont, e.val} \]

\[ v_1 \]

\[ v_2 \]

\[ v_3 \]
Overview

- Distributed Hash Tables (DHT)
- ViP2P
- AnnoVIP
AnnoVIP: Annotations in ViP2P
The peers may store:
- documents (XML)
The peers may store:
- documents (XML)
- annotations (RDF)
The peers may store:
- documents (XML)
- annotations (RDF)
- views
AnnoVIP: Overview of the new features

- Extended pattern dialect for views and queries (**tree patterns with value joins** - subset of XQuery).
AnnoVIP: Overview of the new features

- Extended pattern dialect for views and queries (tree patterns with value joins - subset of XQuery).
- Modified view materialization.
  - each tree pattern of a view may contain data from different documents
  - each document is published independently
AnnoVIP: Overview of the new features

- Extended pattern dialect for views and queries (tree patterns with value joins - subset of XQuery).
- Modified view materialization.
  - each tree pattern of a view may contain data from different documents
  - each document is published independently
- Extended rewriting algorithm to deal with the new pattern dialect.
AnnoVIP: Overview of the new features

- Extended pattern dialect for views and queries (**tree patterns with value joins** - subset of XQuery).
- Modified **view materialization**.
  - each tree pattern of a view may contain data from different documents
  - each document is published independently
- Extended **rewriting algorithm** to deal with the new pattern dialect.
- Support for **namespaces** (needed especially for annotations)
AnnoVIP: Overview of the new features

- Extended pattern dialect for views and queries (tree patterns with value joins - subset of XQuery).
- Modified view materialization.
  - each tree pattern of a view may contain data from different documents
  - each document is published independently
- Extended rewriting algorithm to deal with the new pattern dialect.
- Support for namespaces (needed especially for annotations)
- Graphical User Interface (which you are about to see...)
Rewriting examples

$q$

$V_1$

$V_2$

K. Karanasos (Leo)

AnnoVIP

Codex, 19/3/2010 13 / 17
Rewriting examples

$q_2$

```
article
  author
  journal
  title
```

```
rdf:description
  rdf:docabout
  rdf:idabout
  rating
[\text{val='good'}]
```

$V_3$

```
book
  author
```

$V_4$

```
article
  author
  journal
  title
```

```
rdf:description
  rdf:docabout
  rdf:idabout
  rating
```

\text{K. Karanasos (Leo)}

\text{AnnoVIP}

\text{Codex, 19/3/2010}
Rewriting examples

\[ q_2 \]

\[
\text{article} \quad \text{rdf:docabout} \quad \text{rating}[\text{val}='\text{good}']
\]

\[
\text{book} \quad \text{article} \quad \text{rdf:docabout} \quad \text{rating}
\]

No rewriting!
Rewriting examples
Rewriting examples

q₃

article

| author val |
| journal val |
| title val |

rdf:description

| rdf:docabout |
| rdf:idabout |

author val

V₅

rdf:description

| rdf:docabout |
| rdf:idabout |

article

| author val |
| journal val |
| title val |

No rewriting!
Rewriting examples

Diagram:

- q4
- V6
- V7
- V8

Nodes and edges with labels and values.
It's demo time!

Thank you