



Querying Heterogeneous P2P XML Databases

Angela Bonifati,
Icar CNR (Italy)

Joint work with Elaine Q.Chang,
Laks V.S.Lakshmanan and Terence Ho, UBC (Canada)



1

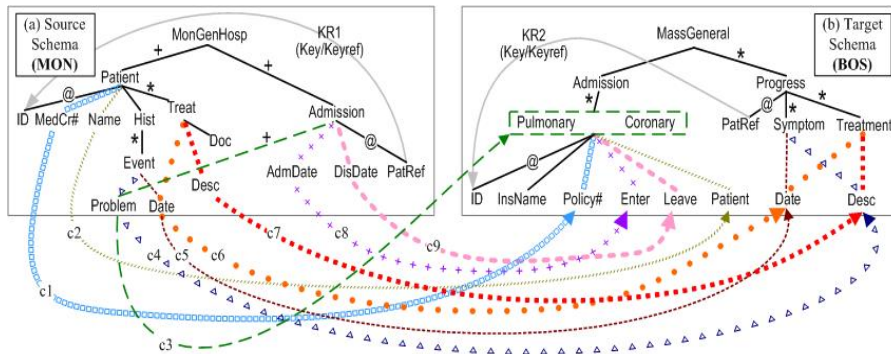
Motivating example

Consider two **schemas of medical centers**
containing **similar and related XML data**

Two hospital schemas

MON

BOS



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005

Motivating example

- **In one schema (MON),** *patients and admissions* appear under separate groups and are *connected by id/idref links*
- **In the second schema (BOS),** the *admissions are grouped by each symptom* and the *patients* appear underneath them

Angela Bonifati, ICAR CNR, Italy

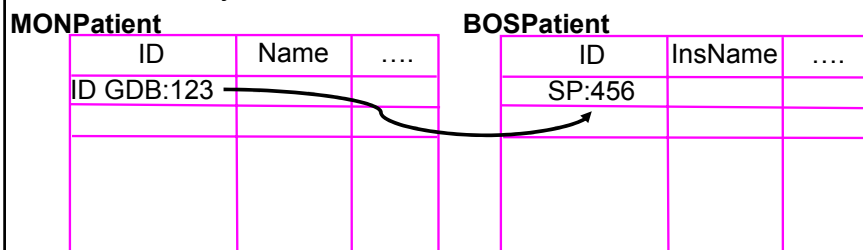
University of Leipzig, Nov.3 2005

4

Differences between the two schemas

- **Value** differences:

- ID GDB:123 in schema MON is SP:456 in schema BOS
- Addressed by [*Clio*, *Hyperion*] with mapping tables
- Affect relational tables as well as XML data, and many other formats



Angela Bonifati, ICAR CNR, Italy

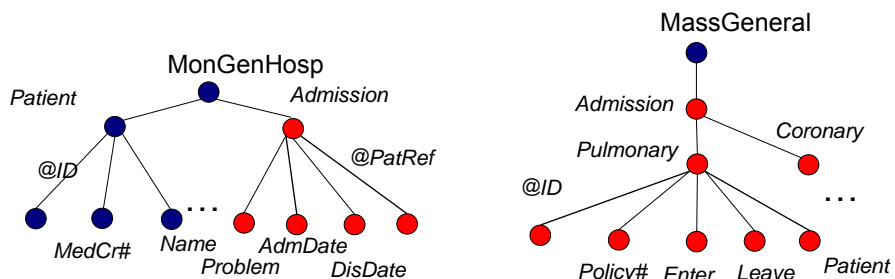
University of Leipzig, Nov.3 2005

5

Differences between the two schemas

- **Structure** differences:

- Admission subtree in schema MON is different from the admission subtree in BOS
- Affect all hierarchical data models (e.g.XML)
- Addressed in Clio [*Clio*], Piazza [*Piazza*]



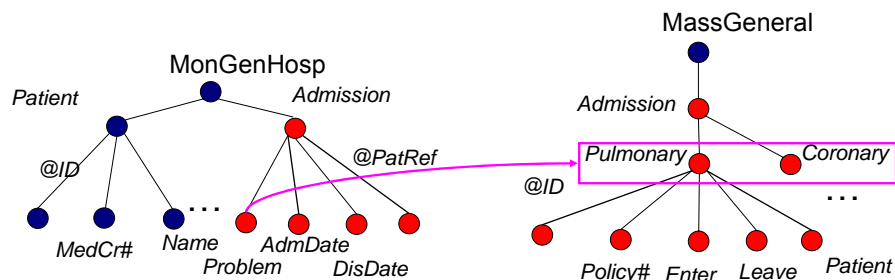
Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005

6

Differences between the two schemas

- **Data/metadata conflicts:**
 - Problem values in MON are represented as tags in BOS
 - *Not considered* in XML data integration thus far



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 7

Underlying P2P scenario

- **We consider heterogeneous XML data and schemas in P2P databases**
 - There is no common mediated schema
 - Each peer has a local schema *to be mapped* to the neighbors schemas (i.e. *acquaintances*)
 - The mapping process has to be kept simple:
 - Either aided by (semi-)automatic mapping tools (e.g. Cupid, COMA)
 - Or manually provided
 - In both cases, the user may want to check the correctness

Angela Bonifati, ICAR CNR, Italy

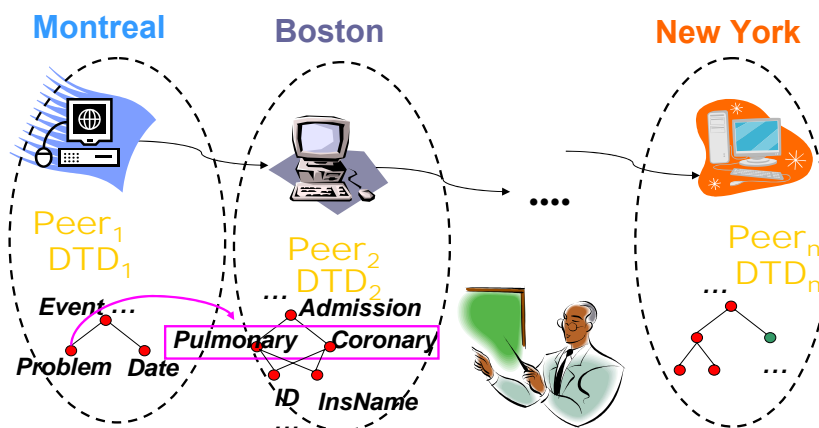
University of Leipzig, Nov.3 2005 8

Outline

- **Application to an Health Care P2P scenario**
- **Informal specification of mappings**
- **Novel rule inference algorithm**
- **Semantics of peer queries**
 - a novel *forward* translation
- **Query translation algorithm**
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- **Experimental evaluation of HePToX¹**
- **Conclusions and future work**

¹ HePToX stands for “Heterogeneous Peers Talk!”

A P2P Network of Heterogeneous Hospitals



Motivations for Health care/Medical scenario

- **Given that:**
 - (1) patients transfer between hospitals
 - (2) the transfers do not always happen between the same set of hospitals, and
 - (3) there is no global mediated schema
- **P2P is a natural choice for health care!**

Options for a patient

- Consider a P2P network of hospitals and an unfortunate patient moving among them:
 - **Option#1: the patient carries his/her own files and query translation is done manually**
 - error-prone
 - unfeasible with several moves and with frequent joins/leaves of peers



Options for a patient

– **Option#2: the hospital db admin has a very good insight of the two schemas and manually writes the mappings:**

- not easy to find a person who knows the rule machinery that well!



Options for a patient

– **Option#3: the hospital db admin provides informal arrows/boxes correspondences w.r.t. a set of acquaintances:**

- made simple for users/applications that do not know the underlying mappings machinery
- kept lightweight for a new peer entering the network



Crossing heterogeneous data

- **Besides patients....:**
 - **doctors:** track patients
 - **insurance companies:** define the policy for a set of patients
 - etc.
- **....also benefit from crossing heterogeneous data**



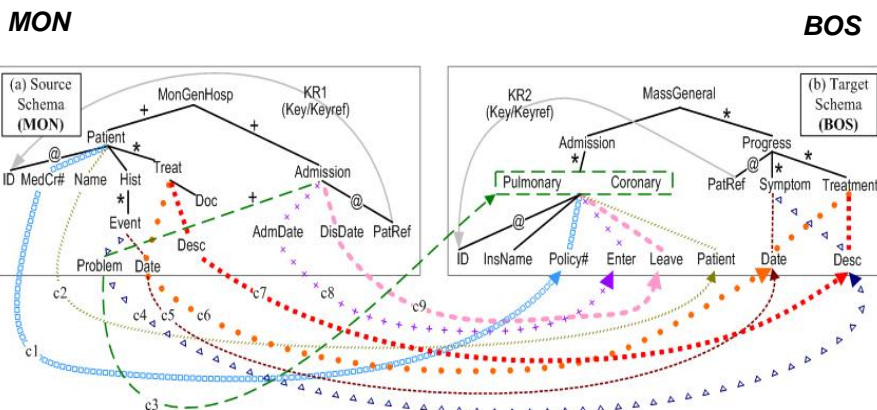
Outline

- **Application to an Health Care P2P scenario**
- **Informal specification of mappings**
- **Novel rule inference algorithm**
- **Semantics of peer queries**
 - a novel *forward* translation
- **Query translation algorithm**
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- **Experimental evaluation of HePToX**
- **Conclusions and future work**

A set of informal correspondences

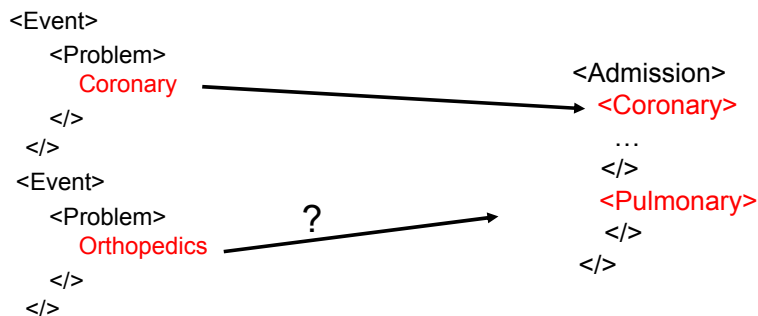
- We consider:
 - **Leaf-to-leaf arrows:**
 - *one-to-one arrows*
 - **Boxes:** semantically similar values are grouped together
 - **Leaf-to-boxes arrows:**
 - *special one-to-many arrows*
- We can handle arbitrary one-to-many or many-to-many correspondences in HePToX:
 - *e.g. union*

Two hospital schemas



Semantics of correspondences

- The arrows/boxes *do not tell* how the source schema is *physically* transformed into the target schema
- They tell how the data of the source is exchanged with the target [*Clio*, *Hyperion*]: **Data Exchange Semantics**
- As a consequence, there may be source data not represented in the target and viceversa



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 19

Disclaimer

- In HepToX, there are no special correspondences for *keys* or *referential constraints across schemas*; these could be encoded as *value correspondences with mapping tables* [*Clio*]
- *Underlying messages:*
 - **even an unfamiliar user can provide our intuitive correspondences**
 - **we can draw complex mappings (e.g. using boxes)**

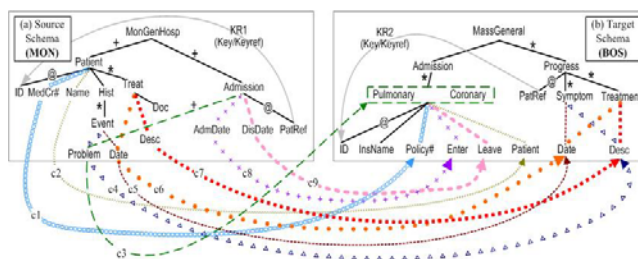
Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 20

Outline

- Application to an Health Care P2P scenario
- Informal specification of mappings
- Novel rule inference algorithm
- Semantics of peer queries
 - a novel *forward* translation
- Query translation algorithm
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- Experimental evaluation of HePToX
- Conclusions and future work

From arrows/boxes to actual mapping rules



From graphical notation to logical mapping rules

Rule #1,
Rule #2,
....
Rule #n

A rule inference algorithm

- *Algorithm Input:* Source and Target Schemas, A Set of Arrows/Boxes
- *Algorithm Output:* A set of *Datalog-like mapping rules*
- The Algorithm consists of the following steps:
 - (1) Determine source and target groups of nodes such that each group captures “units” of information
 - (2) For each source group, build the corresponding *tree expression*
 - (3) For each target group, identify *all minimal sets* of source groups that populate information into the target tree expressions and fill in the rules
- ❖ Can be split into a *group formation algorithm* and a *rule construction algorithm*

Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 23

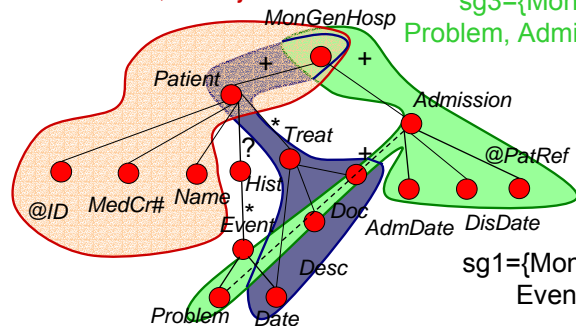
Gathering Nodes into Groups

- Groups are built by identifying *group nodes*

A *group node* is a node with incoming edge labeled with ?, *, + or, alternatively, an ancestor of such a node

sg4={MonGenHosp, Patient, @ID, MedCr#,Name}

sg3={MonGenHosp, Admission, Problem, AdminDate, DisDate, @PatRef}



sg1={MonGenHosp, Patient, Hist, Event, EProblem, EDate}

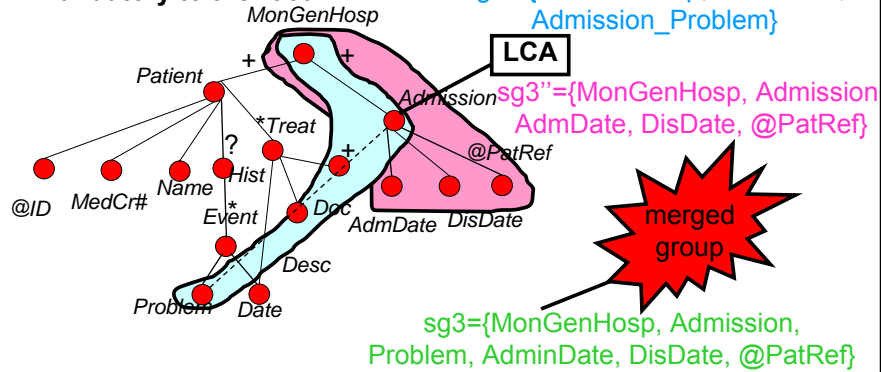
sg2={MonGenHosp, Patient, Treat, Date, Desc, Doc}

Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 24

Merging Groups (if applicable)

- Given two groups, let u be the least common ancestor, they can be merged
 - if all descendants of u in one group are mandatory single-valued relative to the root and all descendants of u in the other group are mandatory to the root

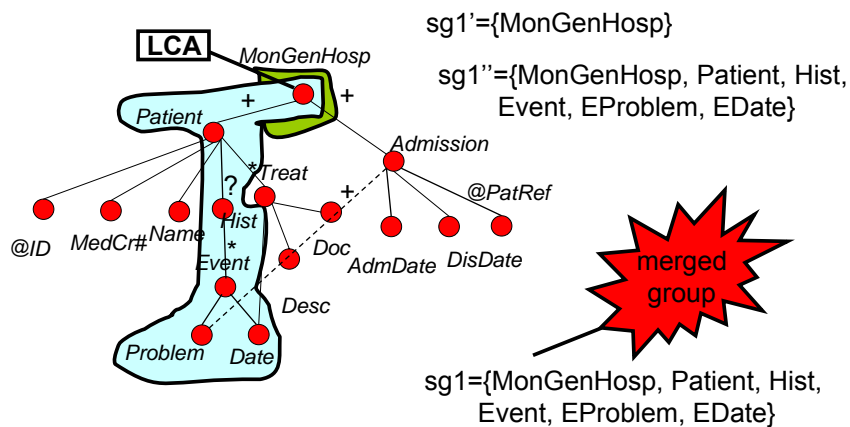


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 25

Merging Groups (if applicable)

- Or:
 - if all descendants of u in one group are mandatory single-valued relative to the root



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 26

Nr. Target Groups = Nr. Rules!

- Our group formation algorithm ensures that **the subgraphs induced by groups are connected**
- How **many target groups** you have determines **how many rules** you need for the mapping
- For the hospital example:
 - **sg1....., sg2 ..., sg3....., sg4.... (two are merged)**
 - **tg1 = {MassGeneral, Admission, Pulmonary, Coronary, @ID, InsName, Policy#, Enter, Leave, Patient}**
 - **tg2 = {MassGeneral, Progress, Symptom, SDate, SDesc}**
 - **tg3 = {MassGeneral, Progress, Treatment, TDate, TDesc}**
 - **tg4 = {MassGeneral, Progress, @PatRef}**
- Pairs of Groups Connected by Arrows:

(sg3, tg1), (sg4, tg1), (sg1, tg2), (sg2, tg3), (sg3, tg4)

From Groups to Tree Expressions

- Any source group induces a *tree*:
 - **Example:**
 - **MonGenHosp -> \$Montreal [Patient -> \$P [Hist -> \$H [Event -> \$E, EProblem -> \$P, EDate -> \$D]]]**
- *If it is a DAG*, we replicate the shared node as many times as necessary to create a tree
 - **does not apply to nodes in a box**
- Any target group induces a tree, but we do not know the node IDs until the rule body is actually written:
 - **Example:**
 - **MassGeneral -> ?? [Admission -> ?? [\$Tag -> ?? [@ID -> ?? , ..., Patient -> ???]]]**

How a mapping rule would look like

- The mapping language is an adaptation of **SchemaLog**, a syntactic high-order language for querying and restructuring relational data
- A mapping rule consists of *atoms* of the form:

<HeadOfRule>

←

Tag₁ -> id₁, [Tag₂ -> id₂
[Tag₃ -> id₃, ..., Tag_i -> id_i],
..., Tag_n -> id_n]

where id may be a variable \$v or a Skolem function f(\$v1,...\$vn)

Rule Construction algorithm

- *Algorithm Input*: Source Groups and Target Groups
- *Algorithm Output*: A set of *Datalog-like* rules
- The Algorithm consists of the following steps:
For each target group tg:
 - (1) Start with rule skeleton $TE(tg) \leftarrow TE(sg_1), \dots, TE(sg_j)$; fill in the variables that corresponds to leaf positions in tg
 - (2) For root and single-value descendants of it, assign their ids as distinct skolem functions of the root variable
 - (3) For each internal node, assign its id as skolem function of the single-value descendants. If any of these does not belong to tg, track back the arrow to find the sg_i that contains its corresponding node and add $TE(sg_i)$ to the rule **body**

Building the rule step by step

- Applying step (1) of Rule Construction Algorithm:

```
MassGeneral -> ??? [Admission -> ???  
  [$AP/text() -> ???  
    [@ID -> $ID, Policy -> $M,  
      Enter -> $AD, Leave -> $DD, Patient -> $N]]]
```

←

```
sg4 MonGenHosp -> $Montreal [Patient -> $P  
  [@ID -> $ID, MedCr# -> $M, Name -> $N],  
sg3 Admission -> $A [Problem -> $AP, AdmDate -> $AD,  
  DisDate -> $DD, @PatRef -> $PR]], $PR = $ID
```

Building the rule step by step

- Applying step (2) of Rule Construction Algorithm:

```
MassGeneral -> f1($Montreal) [Admission -> f2($Montreal)  
  [$AP/text() -> ???  
    [@ID -> $ID, Policy -> $M,  
      Enter -> $AD, Leave -> $DD, Patient -> $N]]]
```

←

```
MonGenHosp -> $Montreal [Patient -> $P  
  [@ID -> $ID, MedCr# -> $M, Name -> $N],  
  Admission -> $A [Problem -> $AP, AdmDate -> $AD,  
  DisDate -> $DD, @PatRef -> $PR]], $PR = $ID
```


Building the rule step by step

- Applying step (3) of Rule Construction Algorithm:

```
MassGeneral -> f1($Montreal) [Admission -> f2($Montreal)
[$AP/text() -> f3($AP/text(), $ID, $M, $AD, $DD, $N)
[@ID -> $ID, Policy -> $M,
Enter -> $AD, Leave -> $DD, Patient -> $N]]]
```

←

```
MonGenHosp -> $Montreal [Patient -> $P
[@ID -> $ID, MedCr# -> $M, Name -> $N],
Admission -> $A [Problem -> $AP, AdmDate -> $AD,
DisDate -> $DD, @PatRef -> $PR]], $PR = $ID
```

How a mapping rule would look like

- Here is one rule for the first target group (**sg3, tg1**), (**sg4, tg1**):

```
MassGeneral -> f1($Montreal)[Admission -> f2($Montreal)
[$AP/text() -> f3($AP/text(), $ID, $M, $AD, $DD, $N)
[@ID -> $ID, Policy -> $M,
Enter -> $AD, Leave -> $DD, Patient -> $N]]]
```

←

```
MonGenHosp -> $Montreal [Patient -> $P
[@ID -> $ID, MedCr# -> $M, Name -> $N],
Admission -> $A [Problem -> $AP, AdmDate -> $AD,
DisDate -> $DD, @PatRef -> $PR]], $PR = $ID
```

Components of mapping rules

- A Skolem function is bound to the mandatory single-value subelements of a node
- E.g. for the nodes in the box Pulmonary and Coronary:
f3(\$AP/text(), \$ID, \$M, \$AD, \$DD, \$N)
- If keys are known, we take advantage of them in the arguments of Skolem functions
- The equality \$PR=\$ID ensures that the rule is safe
- Binding variables across rules are 'glued together'

Components of mapping rules

MR#3: Boston -> f1(\$Montreal) [Progress -> f2(\$PR)
 [Treatment -> f3(\$TDate, \$TDesc)
 [Date -> \$TDate, Desc -> \$TDesc]]
 ←
 Montreal -> \$Montreal [Patient -> \$P
 [@ID -> \$ID, MedCr# -> \$M, Name -> \$N, Treat -> \$T
 [Date -> \$TDate, Desc -> \$TDesc]],
 Admission -> \$A
 [Problem -> \$AP, AdmDate -> \$AD,
 DisDate -> \$DD, @PatRef -> \$PR] \$PR = \$ID.

MR#4:
 Boston -> f1(\$Montreal) [Progress -> f2(\$ID) @PatRef -> \$PR]
 ←
 Montreal -> \$Montreal [Admission -> \$Adm
 [Problem -> \$Prob, AdmDate -> \$AD,
 DisDate -> \$DD, @PatRef -> \$PR] \$PR = \$ID

Outline

- Application to an Health Care P2P scenario
- Informal specification of mappings
- Novel rule inference algorithm
- **Semantics of peer queries**
 - a novel *forward* translation
- Query translation algorithm
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- Experimental evaluation of HePToX
- Conclusions and future work

Query Answering Semantics

Given a mapping $m: \Delta_1 \rightarrow \Delta_2$:

- Backward Query Translation:
 - means querying *against* the direction of the mapping m , e.g.
 - for each $D_1, Q_2^t(D_1) = Q_2(m(D_1))$
- Forward Query Translation:
 - means querying *along* the direction of the mapping m , e.g.
 - for each $D_2, Q_1^t(D_2) = \bigcap_k Q_1(m(D_1^k))$ all pre-images D_1^k such that $D_2 = m(D_1^k)$

Query Answering Semantics

- Backward Query Translation:
 - similar to view expansion
 - the easiest of the two
- Forward Query Translation:
 - a key contribution of HePToX
 - not easy since the mapping m may not be invertible

Outline

- Application to an Health Care P2P scenario
- Informal specification of mappings
- Novel rule inference algorithm
- Semantics of peer queries
 - a novel *forward* translation
- Query translation algorithm
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- Experimental evaluation of HePToX
- Conclusions and future work

XQuery Translation Algorithm

- On a subset of XQuery corresponding to Joins of Tree Patterns (TP), with return arguments equal to leaf nodes:

``Find all patients
with Admission/Problem = 'Coronary' whose Treatment
started Dec, 25th 2003"

Q1:

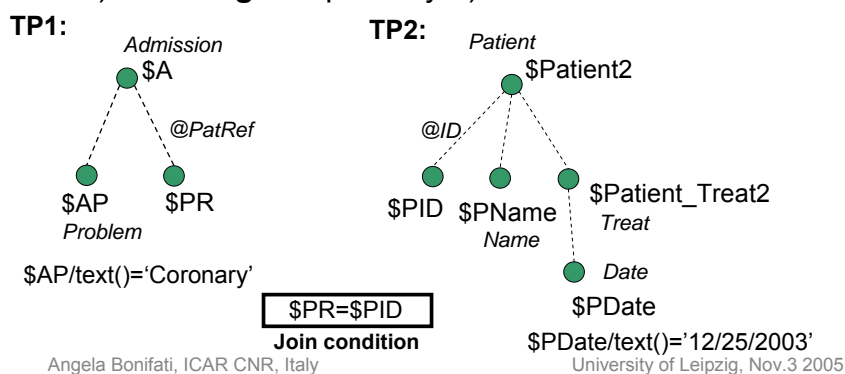
```
FOR $A IN //Admission,
    $P IN //Patient[@ID=$A/@PatRef]
WHERE $A/Problem="Coronary" AND
      $P/Treat/Date="12/25/2003"
RETURN {$P/Name}
```

Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 41

Steps of the translation algorithm (Forward)

- Each XQuery query is reduced to a join of TP
- Each TP is 1) **Expanded**, 2) **Translated w.r.t. each mapping rule**
- The obtained translated TPs then undergo the phases of 3) **Stitching** and possibly 4) **Contraction**

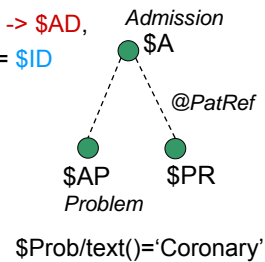


TP1 versus Mapping Rule 1

MassGeneral -> f1(\$Montreal)[Admission -> f2(\$Montreal)
 [\$AP/text() -> f3(\$AP/text(), \$ID, \$M, \$AD, \$DD, \$N)
 [@ID -> \$ID, Policy -> \$M,
 Enter -> \$AD, Leave -> \$DD, Patient -> \$N]]]

←

MonGenHosp -> \$Montreal [Patient -> \$P
 [@ID -> \$ID, MedCr# -> \$M, Name -> \$N],
 Admission -> \$Adm [Problem -> \$Prob, AdmDate -> \$AD,
 DisDate -> \$DD, @PatRef -> \$RefVal], \$RefVal = \$ID

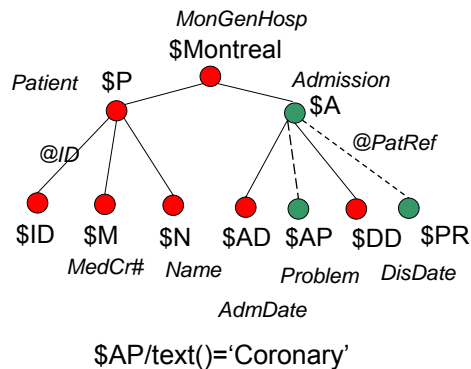


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 43

Expansion

- Expansion: the TP is enriched with the nodes present in the rule body (*red nodes are dummy nodes*)



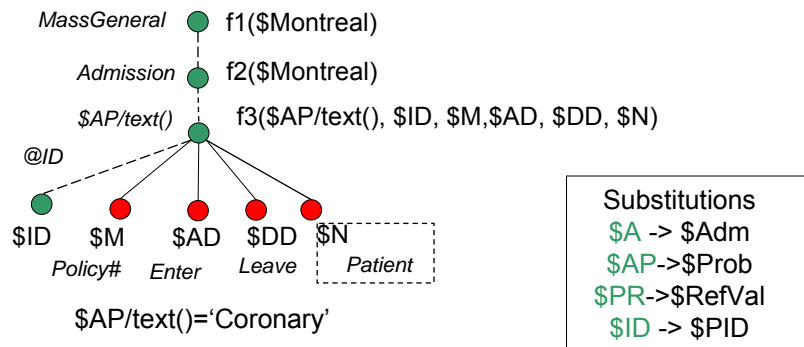
Substitutions
 \$A -> \$Adm
 \$AP -> \$Prob
 \$PR -> \$RefVal

Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 44

Translation

- Translation: the TP is translated against the rule head (while paying attention to dummy nodes!)

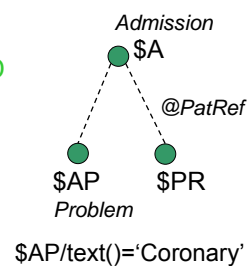


TP1 versus Mapping Rule 4

MassGeneral -> f1(\$Montreal) [Progress -> f2(\$ID) [@PatRef -> \$PR]]

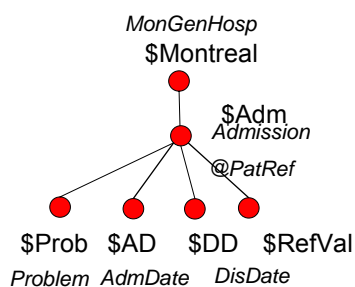
←

MonGenHosp -> \$Montreal [Admission -> \$Adm [Problem -> \$Prob, AdmDate -> \$AD, DisDate -> \$DD, @PatRef -> \$PR], \$PR = \$ID



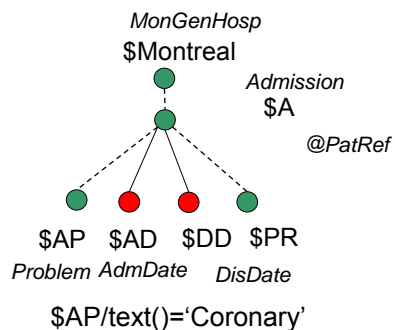
Rule Body (for Mapping Rule 4)

- The rule body builds a tree:



Expansion

- Expansion: the TP is enriched with the nodes present in the rule body (*red nodes are dummy nodes*)

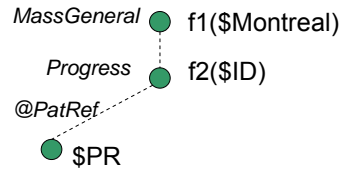


Substitutions:

\$A -> \$Adm
 \$AP->\$Prob
 \$PR->\$RefVal

Translation

- Translation: the TP is translated against the rule head
(there are no dummy nodes this time)

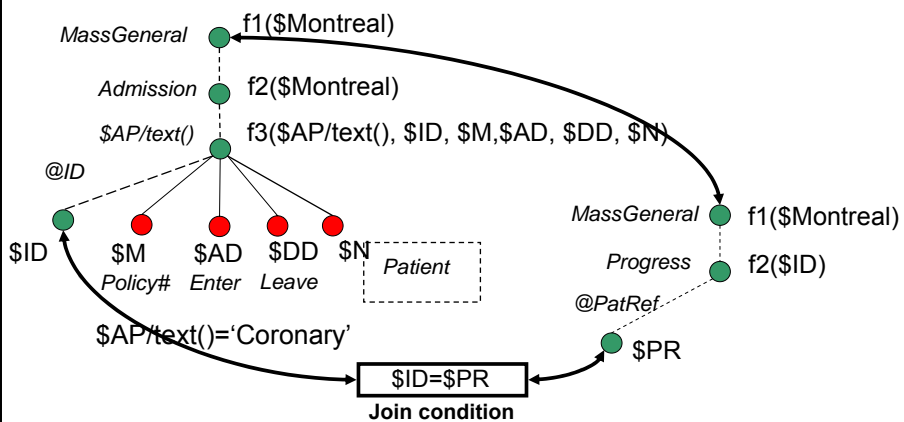


Substitutions:

\$A -> \$Adm
\$PR->\$RefVal

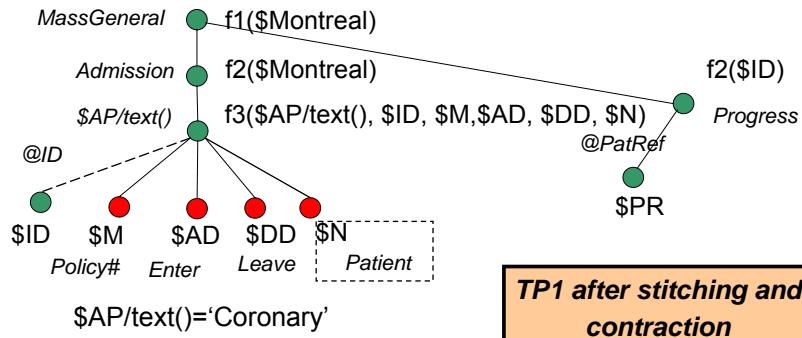
Stitching

- Stitching: the obtained translated pieces are unified;
unification is obtained via identity substitution



Contraction

- Contraction: leaf dummy nodes are dropped; internal dummy nodes are dropped if their children are all dummy

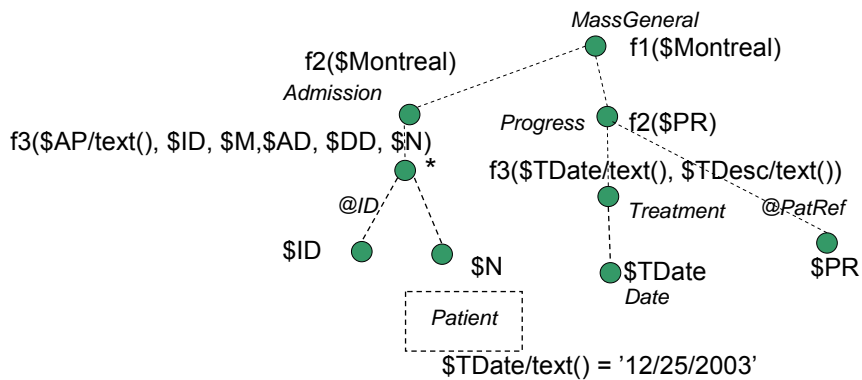


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 51

TP2 after Stitching and Contraction

- TP2 after translation against MR1, MR3 and MR4, stitching and contraction



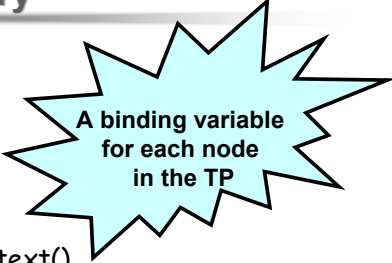
Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 52

The translated XQuery query

Trivial translation:

```
FOR $L1 IN /MassGeneral,  
  $L2 IN $L1/Progress,  
  $PatRef IN $L2/@PatRef,  
  $PDate IN $L2/Treatment/Date/text(),  
  $L3 IN $L1/Admission/Coronary,  
  $PName IN $L3/Patient/text(),  
  $PID IN $L3/@ID  
WHERE $PDate = "12/25/2003"  
  AND $PatRef = $PID  
RETURN $PName
```

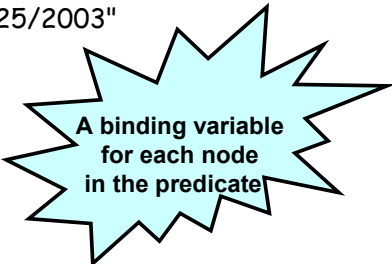


A binding variable
for each node
in the TP

The translated XQuery query

Smart translation:

```
FOR $C IN /MassGeneral/Admission/Coronary,  
  $P IN /MassGeneral/Progress  
WHERE $C/@ID=$P/PatRef AND  
  $P/Treatment/Date="12/25/2003"  
RETURN {$C/Patient}
```



A binding variable
for each node
in the predicate

Correctness of Query Translation

- We have proven that for the considered fragment of XQuery, the query translation algorithm is

correct w.r.t. the given semantics

Outline

- **Application to an Health Care P2P scenario**
- **Informal specification of mappings**
- **Novel rule inference algorithm**
- **Semantics of peer queries**
 - a novel *forward* translation
- **Query translation algorithm**
 - a significant subset of XQuery
 - works *against and along* the direction of rules
- **Experimental evaluation of HePToX**
- **Conclusions and future work**

Experimental guidelines

- Each peer is connected to a set of acquaintances
 - **Transitive mappings produce semantic paths**
- The queries are propagated from a peer to its acquaintances *and recursively on*
- If multiple paths connect a pair of peers, we preferably choose the *shortest* path
- Cycles are detected by marking queries with a unique global ID

Setup of experiments

- As a P2P routing algorithm we used DHT-based FreePastry
- We emulate a real network with Emulab:
 - **50 real machines with full resources preemption (as opposed to Planetlab)**
 - **delay and bandwidth to simulate geographical network behavior**
- As a query engine for XQuery, we use QIZX [QIZX]

Outline of experiments

- Rule Inference Algorithm
 - Query Translation Across the Mappings
 - Query Performance
 - System Scalability
- *All these experiments have been run for both synthetic and realistic datasets*
- **XMark documents (10 schema variations)**
 - **DBResearch documents (19 variants, same used in Piazza)**

Queries for XMark scheme

QMC=%of rules traversed by Qi / avg number of rules for each scheme pair

| Q | Query Description | QMC |
|-----|------------------------------------------------------------|-------|
| Q1 | <i>Selection with 1filter, Mich. QR3, etc.</i> | 11,4% |
| Q2 | <i>Selection with 2filters, Mich. QS5, etc.</i> | 13,7% |
| Q3 | <i>Selection with 3filters, Mich. QS16, etc.</i> | 20,7% |
| Q4 | <i>Selection with 2filters (1nested), Mich. QS18, etc.</i> | 11,5% |
| Q5 | <i>Selection with 3filters (2nested), Mich. QS34 etc.</i> | 11,8% |
| Q6 | <i>1Join, Selection with 2filters, Mich. QJ1, etc.</i> | 28,6% |
| Q7 | <i>1Join, Selection with 1filter, Mich. QJ3, etc.</i> | 28% |
| Q8 | <i>3Joins, Selection with 1filter, No corresp. Mich.</i> | 62,5% |
| Q9 | <i>6Joins, Selection with 1filter, No corresp. Mich.</i> | 100% |
| Q10 | <i>9Joins, Selection with 1filter, No corresp. Mich.</i> | 100% |

Rule Inference Algorithm Performance

- HepToX GUI allows the user to draw a few arrows/boxes and lets us generate the corresponding rules accordingly.

| <i>Dataset</i> | <i>XMark</i> | <i>DBResearch</i> |
|----------------------|--------------|-------------------|
| Avg Nr. Of Arrows | 65 | 7 |
| Avg Nr. Of Rules | 9 | 3 |
| Rule Generation Time | 45ms | 25ms |

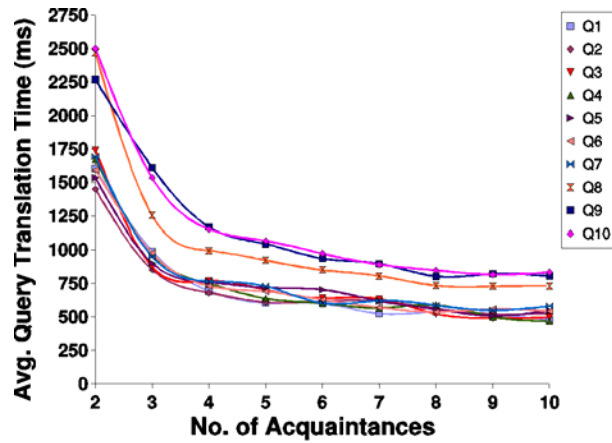
- *Tested on a P4machine, with 3.0GHz and 2GB memory.*

Query Translation in HePToX

- The query translation time is affected by 3 parameters:
 - **Avg number of acquaintances across all sets**
 - **'Degree of heterogeneity' within the acquaintances**
 - **Total number of distinct schemas in the network**

Avg Query Time VS. Nr. Of Acquaintances

In order to keep the number of translations reasonably low, we have to choose a number of acquaintances of (at least) 4.

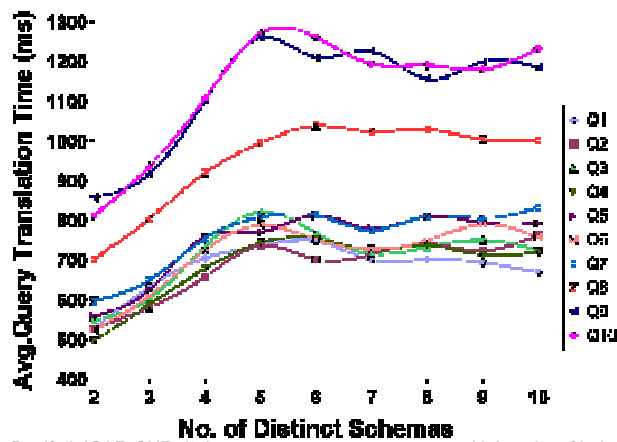


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 63

Query Translation Time

When number of acquaintances = 4, the query translation time reaches a stable point at almost the same No. of distinct schemas

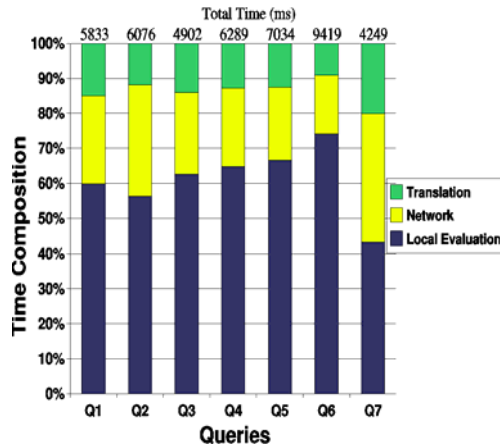


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 64

Query Composition for Querying XMark

Minimal overhead due to query translation; missing queries (Q8, Q9, Q10) did not actually terminate in the QIZX engine

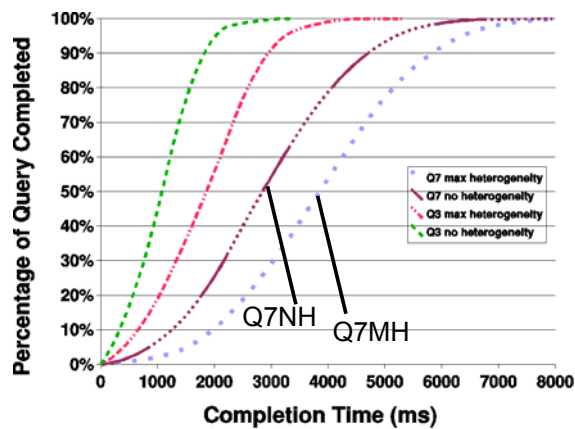


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 65

Query Completion VS. Timeout

The difference between one common schema and 10 different schemas is negligible

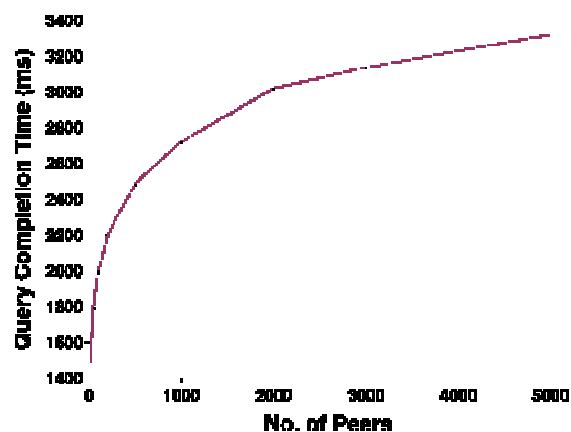


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 66

System Scalability

Query completion follows a regular logarithmic curve; this is with only Pastry (no Emulab)



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 67

About HePToX demo at last VLDB

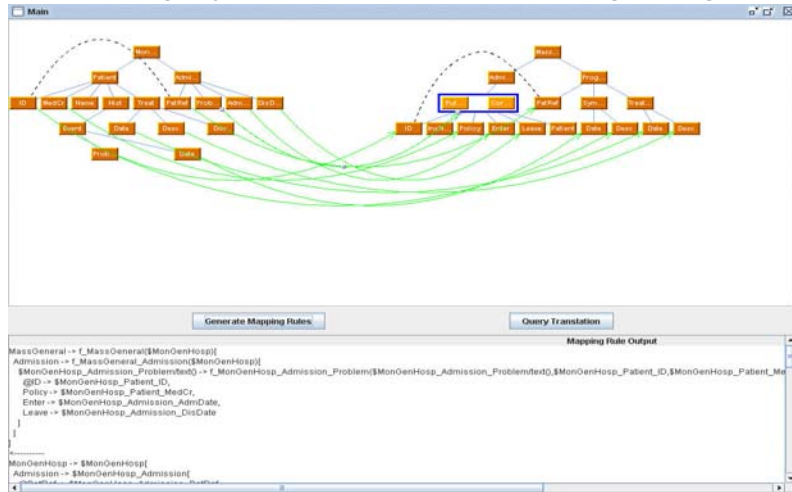
- **HePToX has been demonstrated at VLDB05**
- **The prototype has the following features:**
 - Draw mappings and show the generation of rules
 - Show the query translation algorithm at work
 - Show a real network emulation with Emulab
- **HePToX is implemented in Java:**
 - Uses **QizX** [QizX], **FreePastry** and **Emulab**
 - It consists of ~10,000 lines of code
- **Here are some snapshots....**

Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005

Demo Screenshots 1/2

Schema Mappings By Boxes/Arrows and Corresponding Datalog-like Rules

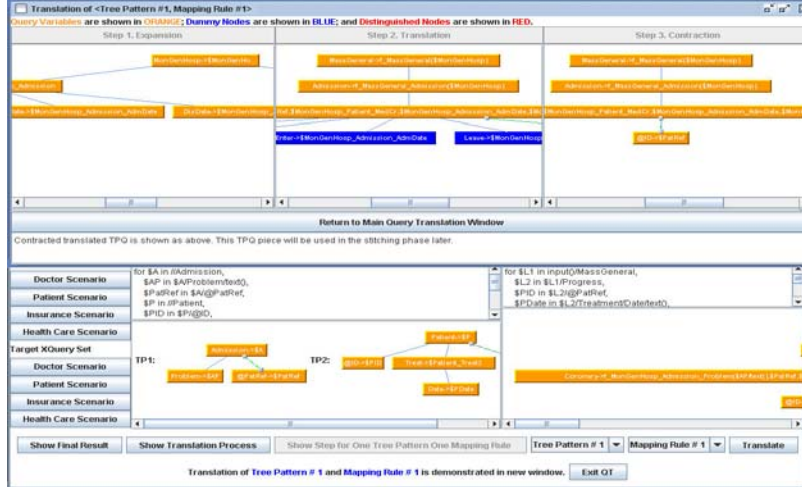


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005

Demo Screenshots 2/2

Details of Query Translation Algorithm (for each pair <TP, MR>)



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005

Outline

- **Application to an Health Care P2P scenario**
- **Informal specification of mappings**
- **Novel rule inference algorithm**
- **Semantics of peer queries**
 - a novel *forward* translation
- **Query translation algorithm**
 - a significant subset of XQuery
 - works against and along the direction of rules
- **Experimental evaluation of HePToX**
- **Conclusions and future work**

Wrapping up

- Previous work has not considered *data/metadata* interplay for XML data translation
- We have proposed a novel algorithm for inferring precise rules from arrows/boxes on DTDs
- We have devised an efficient query translation algorithm
- We have a new semantics for *forward direction of translation*
- We have conducted an *experimental study* to assess HePToX utility and scalability

Future Work

- Incorporating **keys**, or possibly broader functional dependencies
- Dealing with **GTPs**, i.e. extension of TP for XQuery queries
- Studying the **information-preservation** problem within our class of transformations



Grazie

References 1/2

- **The HepTox Web Site:** <http://www.cs.ubc.ca/~laks/heptox.html>
- **[HepTox Demo]** A. Bonifati et al. **HEPTOX: Marrying XML and Heterogeneity in Your P2P Databases.** In *Proc. of Vldb*, 2005.
- **[HepToxSubmitted]** A. Bonifati et al. **Schema Mapping and Query Translation in Heterogeneous P2P Databases.** *Submitted for publication, 2005.*
- **[QIZX]** <http://www.xfra.net/qizxopen/>
- **[Piazza]** A.Y. Halevy et al. **Schema Mediation in Peer Data Management Systems.** In *Proc. of Icde*, 2003.
- **[Lenzerini]** D. Calvanese et al. **Logical Foundations of Peer-To-Peer Data Integration.** In *Proc. of Pods*, 2004.
- **[Pods95]** A. Y. Levy et al. **Answering Queries Using Views.** In *Proc. of Pods*, 1995.

References 2/2

- **[Fagin]** R. Fagin et al. **Composing Schema Mappings: Second-Order Dependencies to the Rescue .** In *Proc. of Pods*, 2004.
- **[Clio]** L. Popa et al. **Translating Web Data.** In *Proc. of Vldb*, 2002.
- **[Hyperion]** Marcelo Arenas et al. **The Hyperion Project: From Data Integration to Data Coordination.** *SIGMOD Record*,32(3):53-58, 2003.
- **[Bohannon]** P. Bohannon et al. **Information Preserving XML Schema Embedding** In *Proc. of Vldb*, 2005.
- **[Benedikt]** M. Benedikt et al. **Capturing both Types and Constraints in Data Integration** In *Proc. of Sigmod*, 2003.
- **[COMA++]** D. Aumueller et al. **Schema and ontology matching with COMA++.** In *Proc. of Sigmod*, 2005.
- **[iFuice]** E. Rahm et al. **iFuice - Information Fusion utilizing Instance Correspondences and Peer Mappings.** In *Proc. of WebDB*, 2005.



If there is any time left...



77

HePToX versus [Clio]

- Clio is one of the *pioneers in schema mapping discovery*, translation and maintainance
- Operates on *both XML and relational*
- Moreover, under Hyperion, they introduce *mapping tables*

- Groups are similar to primary paths in Clio, which *are not* able to account for nodes in boxes
- Clio/Hyperion do not handle *data/metadata conflicts* as in HePToX
- Internal nodes with complex structures across schemas are nicely addressed in HePToX

HePToX versus [Piazza]

- Piazza defines *GAV/LAV-style mappings* across peers schemas
- The mapping language is *XQuery*, whereas in our case the mappings are fairly intuitive
- The forward query translation in HePToX resembles query answering using views, but leverages Skolem functions and the special nature of mappings in HePToX

- Piazza does not handle *data/metadata conflicts* as in HePToX

HePToX versus [Bohannon], [Benedikt]

- [Bohannon] and [Benedikt] can handle all *kinds of mappings* including recursive ones, but *not data/metadata* conflicts as in HePToX

- [Bohannon] has *node-to-path* mappings

HePToX versus [iFuice]

- [iFuice] includes a *mediator* offering an access to the sources and their mappings
- Mappings are *instance-driven*
- HePToX does not look at instances when building mappings

HePToX versus [COMA++]

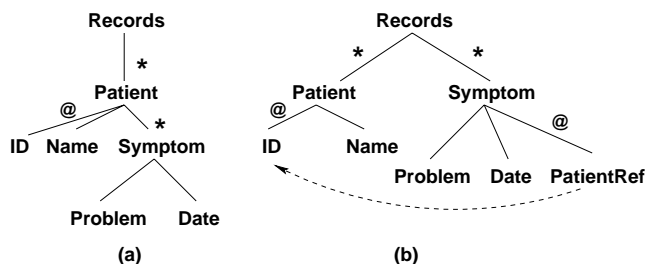
- **COMA++** is a comprehensive tool for schema and ontology mappings
 - features for *composing, merging and comparing* different mappings
- It is interesting to see whether
 - **COMA++ and HePToX could be cascaded, i.e.**
 - **mappings in COMA++ could be used in HePToX and if COMA++ can generate HePToX mappings**

How to get to the rules: the underlying algebra

- The rules are written by looking at the source and target groups
- What is important is that the rules capture a class of algebraic transformations:
 - **UnNest/Nest**
 - **Flip/Flop**
 - **Merge/Split**

Nesting/UnNesting XML trees

DTDs:

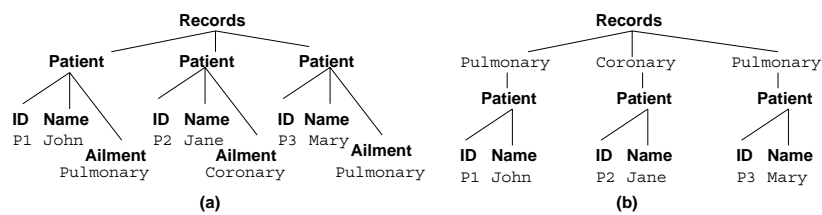


- A child/parent relationship is flattened by means of ID/IDREFs.
- Instances are modified accordingly.

Flipping/Flopping XML Trees

- Flip/Flop: Nodes values in a schema become tags into another schema:
 - notice that nodes can be pulled up the hierarchy at an arbitrary level

Instances:

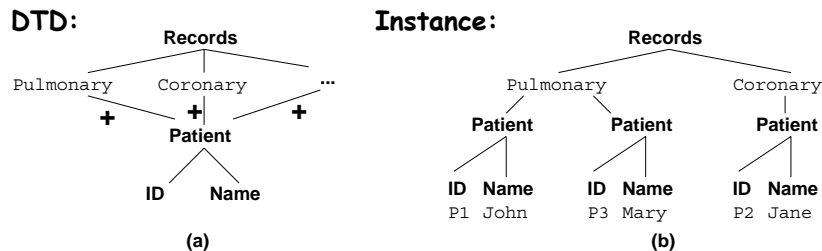


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 85

Splitting/Merging XML Trees

- Nodes having a common parent are grouped/ungrouped together:
 - notice that this corresponds to 'grouping by tag'



Angela Bonifati, ICAR CNR, Italy

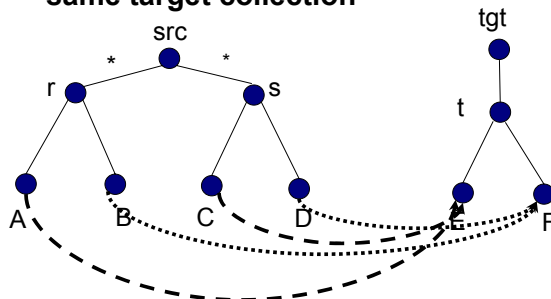
University of Leipzig, Nov.3 2005 86

HepToX vs. previous literature

- [*Clio*, *Hyperion*, *Piazza*, *Lenzerini*] all considered first-order language mappings
 - **Clio extended with XML and Piazza consider XML grouping but not data/metadata transformations as HepToX does**
- [*Fagin*] considered the problem of data exchange for FO rules and FO+aggregates/grouping (SQL)
 - **they do not say how to generate those higher-order mappings**

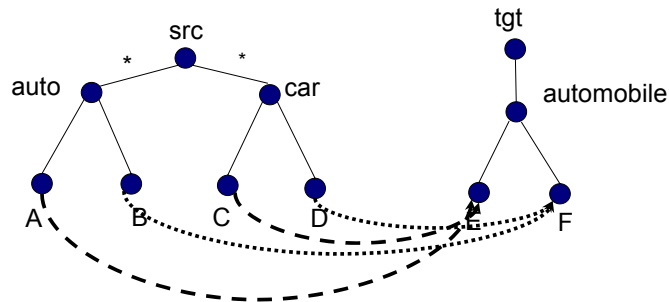
Capturing Union Rules

- HePToX supports union mappings:
 - **two or more source collections can be mapped to the same target collection**



Capturing Union Rules

- 1st scenario:
 - r is auto, s is car, t is automobile

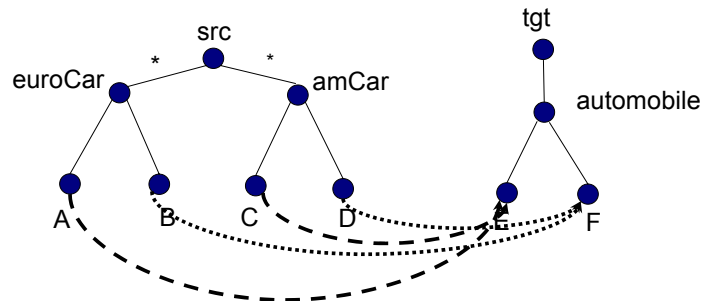


Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 89

Capturing Union Rules

- 2nd scenario:
 - r is euroCar, s is amCar, t is automobile



Angela Bonifati, ICAR CNR, Italy

University of Leipzig, Nov.3 2005 90

Capturing Union Rules

- We introduce the *label mapping tables*:

| <i>src schema label</i> | <i>op</i> | <i>tgt schema label</i> |
|-------------------------|-----------|-------------------------|
| auto | = | automobile |
| car | = | automobile |

| <i>src schema label</i> | <i>op</i> | <i>tgt schema label</i> |
|-------------------------|---------------|-------------------------|
| euroCar | \sqsubseteq | automobile |
| amCar | \sqsubseteq | automobile |