U-MAP : A System for Usage-Based Schema Matching and Mapping

Hazem Elmeleegy
AT&T Labs-Research

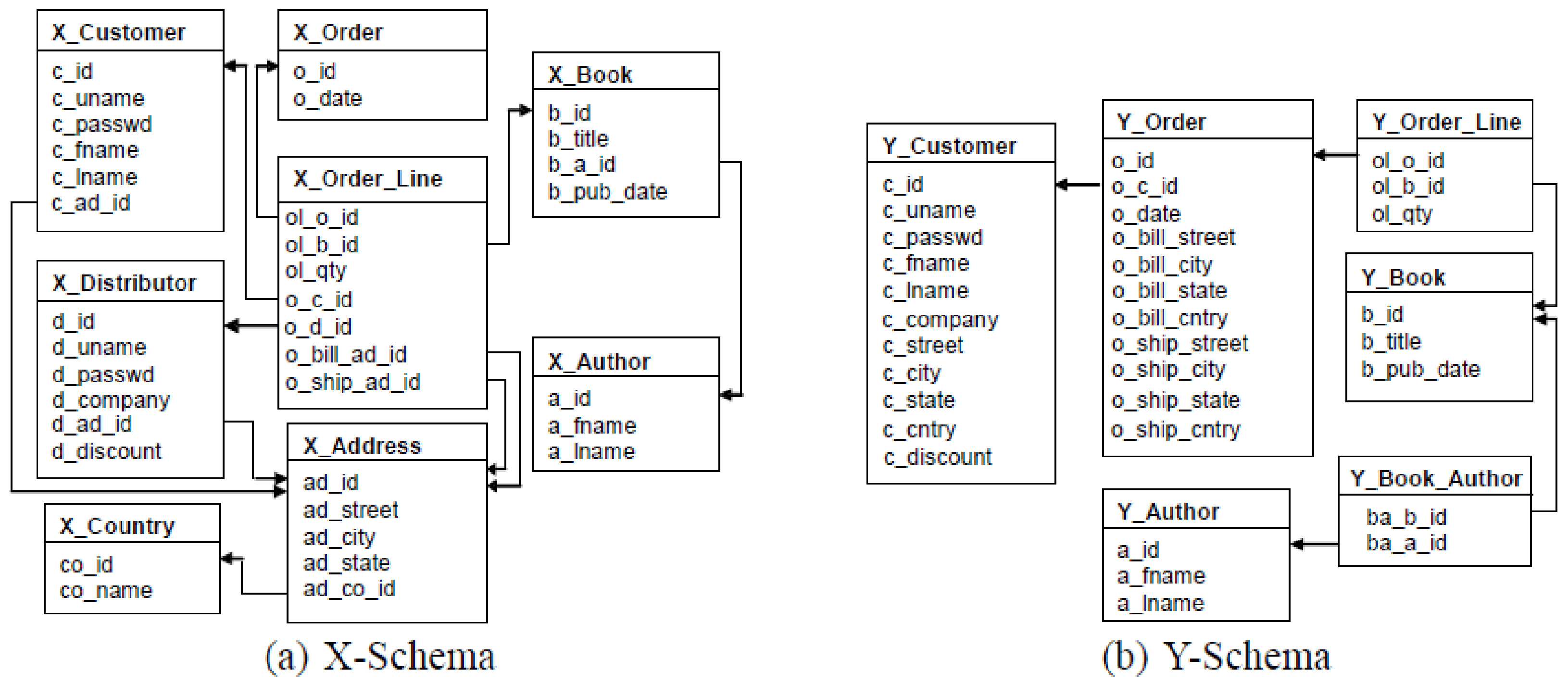
Jaewoo Lee and Elkindi Rezig
Purdue University

Mourad Ouzzani and Ahmed K. Elmagarmid
Qatar Computing Research Institute

Overview

- U-MAP is the first system that uses usage information buried in database query logs to tackle data integration and exchange challenges. U-MAP generates correspondences between the attributes of the source and target schemas, and then complex mapping rules to transform data records from one schema to another using these query logs.

Illustrative Example - Bookstores



(a) X-Schema

(b) Y-Schema

Basic Approach

1. **Generating correspondences:** Match schemas to generate the set of attribute correspondences (using any of a variety of techniques).
2. **Constructing logical relations:** Find all meaningful associations between schema attributes – using the chase algorithm.
3. **Creating mappings:** For each pair of source and target logical relations with corresponding attributes, create a new mapping.
4. **Optimizing mappings:** Eliminate redundant mappings, merge selected mappings together, and so on to ultimately get a "higher quality" set of mappings.
5. **Compiling mappings:** Generate an executable script (e.g., SQL queries) to transform source data records into target data records.

Mapping Example

Mapping M1 (X_Order_Line_LR_2_Y_Order_Line_LR):

```
FOR ol in X_Order_Line, o in X_Order, b in X_Book, a1 in X_Address, a2 in X_Address, …
WHERE ol.o_id = o.o_id AND ol.ol_b_id = b.b_id AND ol.o_bill_addr_id = a1.ad_id
        AND ol.o_ship_addr_id = a2.ad_id AND …
EXISTS ol' in Y_Order_Line, o' in Y_Order, c' in Y_Customer, b' in Y_Book
WHERE ol'.o_id = o'.o_id AND o'.c_id = c'.c_id AND ol'.b_id = b'.b_id
WITH ol'.ol_qty = ol.ol_qty AND o'.o_date = o.o_date AND …
```

}source logical relation

}target logical relation

}correspondences

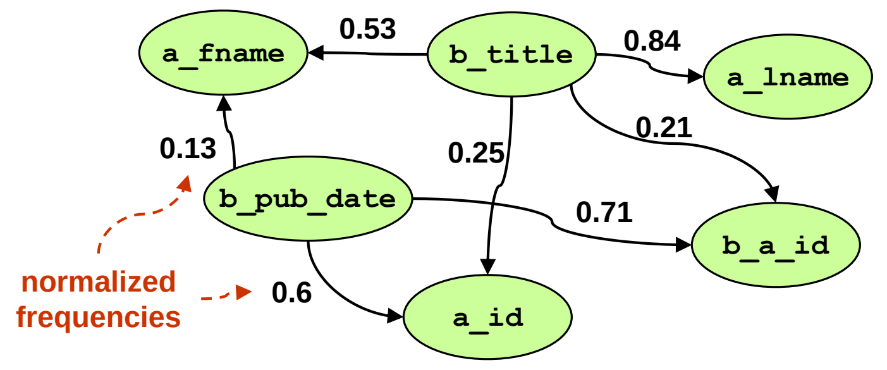
The U-MAP Approach

- Build upon existing approaches for schema matching and mapping.
- Analyze the query log to find guidance on how to handle some of the key unresolved issues in the existing approaches.

Generating Correspondences

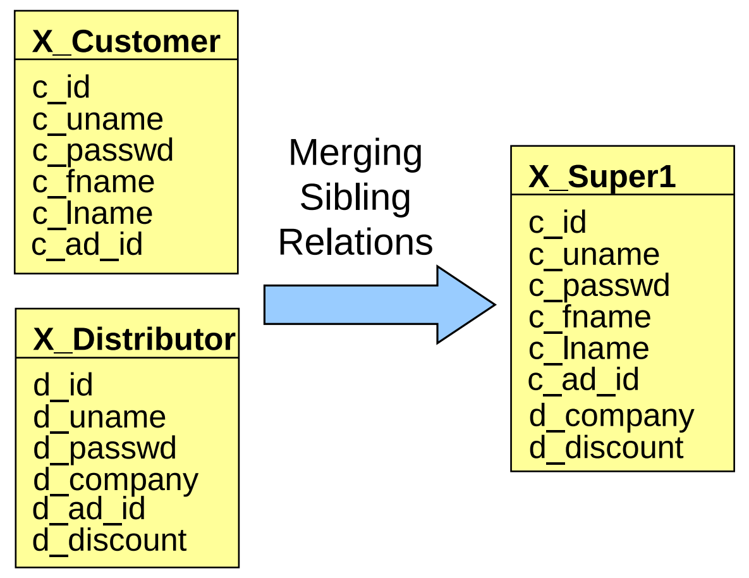
- **Problem:**
  - ➢ The sources of evidence currently used for schema matching (e.g. name similarity, structure similarity, data similarity, etc) are not always available or sufficiently reliable.
- **Solution:**
  - ➢ Introduce a new source of evidence: usage info from query logs
    - ❑ Collect co-occurrence statistics for pairs of attributes in the query log, and match the resulting co-occurrence graphs across the source and target.
    - ❑ Track and compare the usage of attributes with aggregate functions and in join predicates

sample co-occurrence graph:



Managing IS-A Relationships

- **Problem:**
  - ➢ Current tools cannot detect and handle overlapping sibling relations (subclasses of the same super class).
  - ➢ Each sibling relation is mapped separately leading to duplicate records in the target.
- **Solution:**
  - ➢ Detecting candidate sibling relations
    - ❑ Check for shared attributes
  - ➢ Detecting overlapping sibling relations
    - ❑ Check for queries looking for the overlap
  - ➢ Merging overlapping sibling relations
    - ❑ Outer Join the two relations and merge their attributes



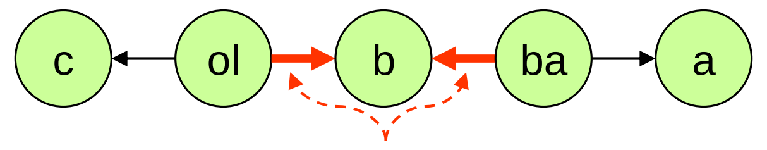
Aggressive Chase

- **Problem:**
  - ➢ Current tools can miss interesting attribute associations because the chase operation is limited to the forward direction.
  - ➢ Example: the association between the order line and the author name in the Y-Schema will be missed.
- **Solution:**
  - ➢ Discover all meaningful pairs of opposite references by analyzing the query log.
  - ➢ Allow chasing in the reverse direction as guided by the discovered pairs in the query log.

Log Query (Find all books by 'Jim Gray' ordered by 'John Smith'):

```
select b_title
from X_Customer c, X_Order_Line ol, X_Book b,
X_Book_Author ba, X_Author a
where c.c_id=ol.o_c_id and ol.ol_b_id=b.b_id
and b.b_id=ba.ba_b_id and ba.ba_a_id=a.a_id
and c_fname='John' and c_lname='Smith'
and a_fname='Jim' and a_lname='Gray'
```

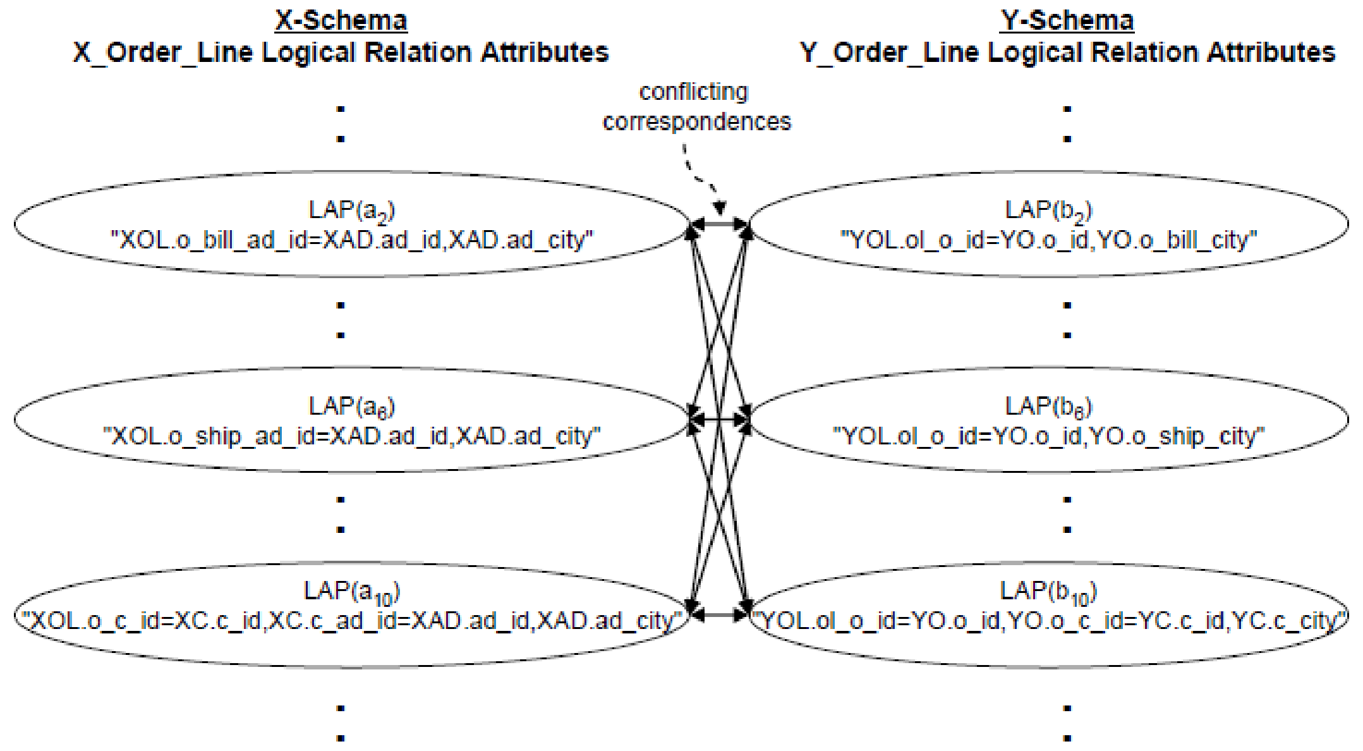
Query Graph:



An example of a join expression containing a meaningful pair of opposite references, which can be used during the aggressive chase

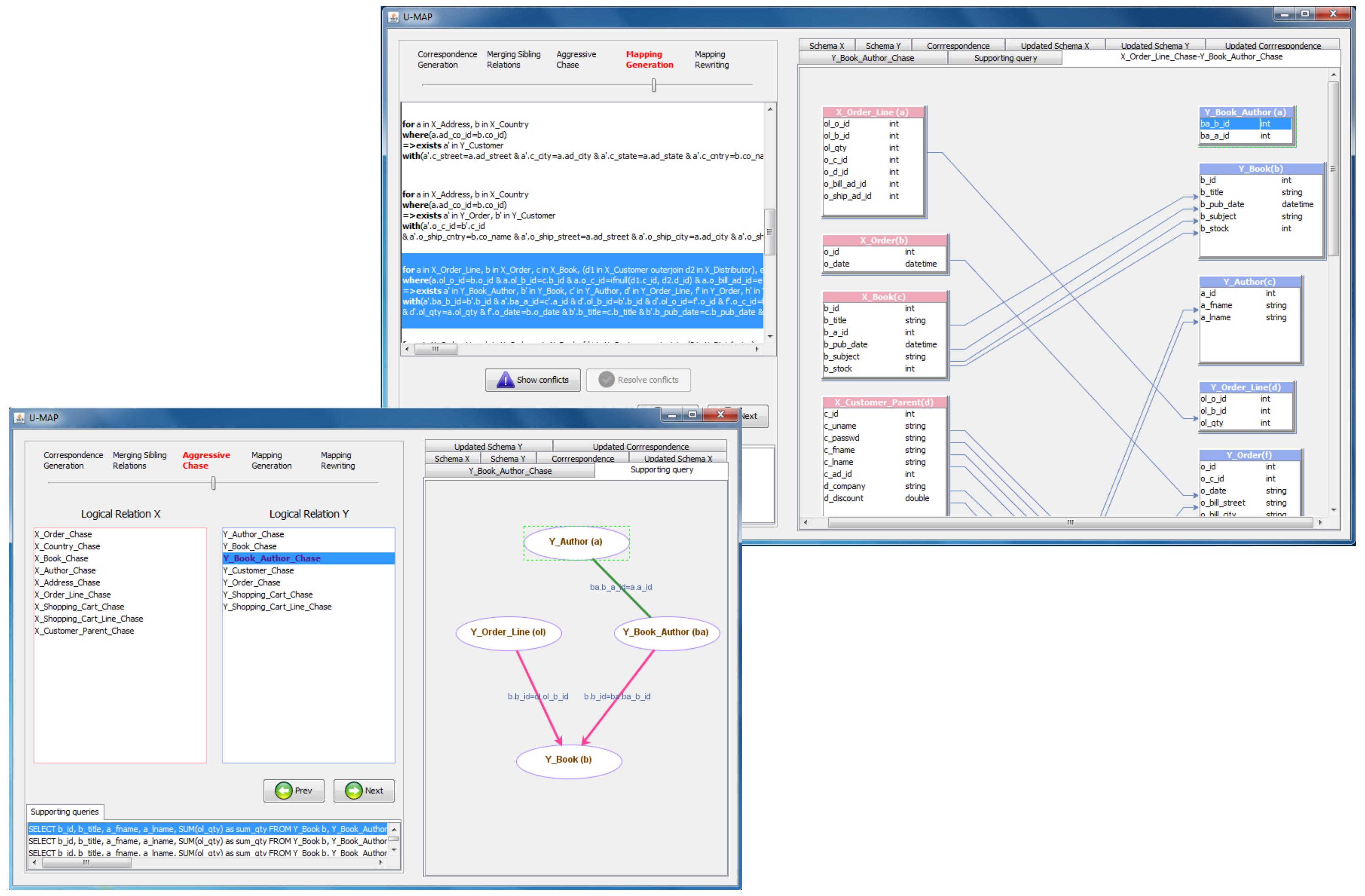
Resolving Correspondence Conflicts

- **Problem:**
  - ➢ Current tools cannot automatically resolve the potential correspondence conflicts across the attributes of the source and target logical relations – during mapping generation.
  - ➢ Existing solutions vary from to generating all possible resolutions to requesting the user to manually resolve the conflicts one attribute at a time.


X-Schema
X_Order_Line Logical Relation Attributes
Y-Schema
Y_Order_Line Logical Relation Attributes
conflicting correspondences
LAP(a2)
"XOL.o_bill_ad_id=XAD.ad_id,XAD.ad_city"
LAP(a6)
"XOL.o_ship_ad_id=XAD.ad_id,XAD.ad_city"
LAP(a10)
"XOL.o_c_id=XC.c_id,XC.c_ad_id=XAD.ad_id,XAD.ad_city"
LAP(b2)
"YOL.ol_o_id=YO.o_id,YO.o_bill_city"
LAP(b6)
"YOL.ol_o_id=YO.o_id,YO.o_ship_city"
LAP(b10)
"YOL.ol_o_id=YO.o_id,YO.o_c_id=YC.c_id,YC.c_city"

- **Solution:**
  - ➢ Treat the problem as a schema matching problem (across the two logical relations).
  - ➢ Build upon the usage-based schema matching approach
  - ➢ Collect the statistics separately for each *version* of the same attribute present in the logical relation.
    - ❑ E.g., three versions of ad_city can represent billing, shipping, and customer cities.
    - ❑ Match attribute *contexts* across the logical relation and the queries in the log to achieve the separation in statistics collection.
  - ➢ Group attributes to limit the number of possible resolutions.

U-MAP System



```
SELECT b_id, b_title, a_fname, a_lname, SUM(ol_qty) as sum_qty FROM Y_Book b, Y_Book_Author
SELECT b_id, b_title, a_fname, a_lname, SUM(ol_qty) as sum_qty FROM Y_Book b, Y_Book_Author
SELECT b_id, b_title, a_fname, a_lname, SUM(ol_qty) as sum_qty FROM Y_Book b, Y_Book_Author
```

Supported by the Qatar Computing Research Institute and
the National Science Foundation under Grants No. IIS-0916614 and IIS-0811954