COMA – A system for flexible combination of schema matching approaches

Hong-Hai Do, Erhard Rahm
University of Leipzig, Germany
dbs.uni-leipzig.de
Content

- Motivation
- The COMA approach
  - Comprehensive matcher library
  - Flexible combination scheme
  - Novel reuse-oriented match approach
- Evaluation setup and results
- Conclusions and future work
Motivation

- **Schema matching**: Finding semantic correspondences between two schemas
- **Crucial step in many applications**
  - Data integration: mediators, data warehouses
  - E-Business: XML message mapping
  - ...
- **Currently manual, time-consuming, tedious**
  - Need for approaches to automate the task as much as possible

PO1.ShipTo.shipToCity ↔ PO2.DeliverTo.Address.City
Individual Match Approaches

- **Schema-based**
  - Element
    - Linguistic
      - Names
      - Descriptions
    - Constraint-based
      - Types
      - Keys
  - Structure
    - Constraint-based
      - Parents
      - Children
      - Leaves

- **Instance-based**
  - Element
    - Linguistic
      - IR (word frequencies, key terms)
    - Constraint-based
      - Value pattern and ranges

- **Reuse-oriented**
  - Element
    - Dictionaries
    - Thesauries
  - Structure
    - Previous match results

Survey paper [Rahm, Bernstein - VLDB Journal’01]
Combining Match Approaches

- **Combination of match algorithms**
  - *Hybrid*: fixed combination, difficult to extend and improve
    - currently most common: Cupid, SemInt, SimilarityFlooding, DIKE, MOMIS, TranScm
  - *Composite*: combination of the results of independently executed matchers
    - currently only for machine learning-based techniques: LSD, GLUE

- **COMA**: Framework for flexible **COmbination of MAtch algorithms**
  - Extensible matcher library
  - Combination scheme with various combination strategies
System Architecture

Schema Import

Matcher Library

Match Iteration

Combination Scheme

User Interaction (optional)

User Feedback

Matcher 1

Matcher 2

Matcher 3

Combination of match results

Mapping

Similarity cube

Combination Scheme

Match Iteration

Matcher execution

Combination Scheme

$S1 \rightarrow S2$

$S2 \rightarrow S1$
Combination Scheme

1. Aggregation of matcher-specific results
   - Matchers: S1, S2
   - Similarity cube

2. Match direction
   - Match results
     - S1→S2
     - S2→S1

3. Selection of match candidates
   - Match candidates
     - MaxN (Max1), Threshold, MaxDelta, Threshold+MaxN, Threshold+MaxDelta

4. Computation of combined similarity
   - Dice, Average
   - Combined similarity: [S1, S2, 0.7]
Match Processing: Example

1. Aggregation

2. Direction

3. Selection

**LargeSmall**
(Match candidates for smaller schema S2)

**Max1**

**Threshold(0.5)**
## Matcher Library

<table>
<thead>
<tr>
<th>Type</th>
<th>Matcher</th>
<th>Schema Info</th>
<th>Auxiliary Info</th>
<th>Constituent Matchers</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple</td>
<td>Affix</td>
<td>Element names</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>n-gram</td>
<td>Element names</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Soundex</td>
<td>Element names</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>EditDistance</td>
<td>Element names</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>Synonym</td>
<td>Element names</td>
<td>External dictionaries</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>DataType</td>
<td>Data types</td>
<td>Data type compatibility table</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>UserFeedback</td>
<td>–</td>
<td>User-specified (mis-) matches</td>
<td>–</td>
</tr>
<tr>
<td>Hybrid</td>
<td>Name</td>
<td>Element names</td>
<td>–</td>
<td>Affix, 3-Gram, Synonym</td>
</tr>
<tr>
<td></td>
<td>TypeName</td>
<td>Data Types+Names</td>
<td>–</td>
<td>DataType, Name</td>
</tr>
<tr>
<td></td>
<td>NamePath</td>
<td>Names+Paths</td>
<td>–</td>
<td>Name</td>
</tr>
<tr>
<td></td>
<td>Children</td>
<td>Child elements</td>
<td>–</td>
<td>TypeName</td>
</tr>
<tr>
<td></td>
<td>Leaves</td>
<td>Leaf elements</td>
<td>–</td>
<td>TypeName</td>
</tr>
<tr>
<td>Reuse-oriented</td>
<td>Schema</td>
<td>–</td>
<td>Existing schema-level match results</td>
<td>–</td>
</tr>
</tbody>
</table>
**Reuse-oriented Matching**

- **The `MatchCompose` operation:** Transitivity of element similarity
  - Composition of similarity relationships

- **Reuse of multiple match correspondences**
  - vs. reuse of single element-level correspondences from synonym tables, thesauries
Schema-level Reuse

The Schema matcher:

- Reuse complete match results at the schema level
- Exploit all possible reuse opportunities
- Limit negative effects of transitivity
Real-world Evaluation

- 5 real-world schemas (XML - Purchase order), 10 match tasks
  - CIDX, Excel, Noris, Paragon, Apertum from biztalk.org
  - 40-145 elements

- Systematic evaluation (automatic mode)
  - 1 Series = 10 Experiments: Test of 1 configuration of (Matcher, Aggregation, Direction, Selection, Combined similarity) with 10 match tasks
  - 12,312 series = 123,120 experiments

<table>
<thead>
<tr>
<th>Matchers</th>
<th>Aggregation</th>
<th>Direction</th>
<th>Selection</th>
<th>Combined Sim</th>
</tr>
</thead>
<tbody>
<tr>
<td>No reuse</td>
<td>5 single</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>11 combinations</td>
<td>-Max -Average -Min</td>
<td>-LargeSmall -SmallLarge -Both</td>
<td>-MaxN(1-4) -Delta(0.01-0.1) -Threshold(0.3-1.0) -Threshold(0.5)+MaxN(1-4) -Threshold(0.5)+Delta(0.01-0.1)</td>
</tr>
<tr>
<td>Reuse</td>
<td>2 single</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>12 combinations</td>
<td>-Max -Average -Min</td>
<td>-LargeSmall -SmallLarge -Both</td>
<td>-MaxN(1-4) -Delta(0.01-0.1) -Threshold(0.3-1.0) -Threshold(0.5)+MaxN(1-4) -Threshold(0.5)+Delta(0.01-0.1)</td>
</tr>
<tr>
<td>Σ</td>
<td>16 + 14</td>
<td>3</td>
<td>3</td>
<td>36</td>
</tr>
</tbody>
</table>
Match Quality Measures

- **Comparison of automatically with manually (i.e. real) derived match correspondences**

  ![Venn Diagram]

  - **A**: Missed matches
  - **B**: Correct matches
  - **C**: False matches

- **Quality measures:**

  \[
  \text{Precision} = \frac{|B|}{|B+C|} \quad \text{Recall} = \frac{|B|}{|A+B|}
  \]

  \[
  \text{SimilarityFlooding [ICDE02]}: \quad \text{Overall} = 1 - \frac{|A+C|}{|A+B|} = \frac{|B-C|}{|A+B|} = \text{Recall} \times \left(2 - \frac{1}{\text{Precision}}\right)
  \]

  - **Overall**: post-match effort to add missed and to remove false matches; negative Overall → no gain
  - Computed for single experiments and averaged over 10 experiments for each series (average Overall, etc.)
Results: Combination Strategies (1)

- Most no-reuse series have negative average Overall

- "Good" matcher/strategy:
  - Positive average Overall
  - High presence in higher Overall ranges

- Aggregation: Average (compensating)

![](image1.png)

![Graph showing distribution of series with overall values ranging from 0.0 to 0.8.](image2.png)

- Average is used by all series with average Overall > 0.6

- Aggregation (2376 series/strategy):

  - Min
  - Max
  - Average

#All Series = 8208

Overall Series share

- Min 0.0 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8

- #Series

- Aggregation: Average (compensating)
Results: Combination Strategies (2)

- **Direction**: Both (considering both directions)
- **Selection**: Threshold+Delta (above threshold + within tolerance)
- **Combined similarity**: Average (pessimistic)
- **Matcher**: All (combination of all hybrid matchers)
Results: Single Matchers

Instability of some single (hybrid) matchers (negative *Overall*) because of shared elements
- E.g. *DeliverTo.Address* and *BillTo.Address*

Considering hierarchical names (*NamePath*) more accurate

Schema-level reuse very effective:
- Essential improvement over no-reuse hybrid matchers
- Reusing approved match results better than automatically derived match results

**SchemaM**: Schema with *manually* derived (real) match results

**SchemaA**: Schema with match results *automatically* derived using the default match operation
Results: Combined Match Approaches

- **Reuse matchers outperform no-reuse matchers**
  - Best no-reuse: All: 0.73 average Overall (Precision 0.95, Recall 0.78)
  - Best reuse: All+SchemaM: 0.82 average Overall (Precision 0.93, Recall 0.89)

- **Combinations outperform single hybrid matchers**
  - Combined matchers, e.g. All, consider many aspects at the same time
  - NamePath+Leaves: effective scheme, considering paths to identify context of shared elements, and leaves to cope with structural conflicts
Results: Match Sensitivity

- Impact of schema characteristics:
  - Degrading match quality with increase of schema size

- Best combinations: no-reuse All and reuse-oriented All+Schema
  - High stability across different match tasks
  - Little tuning effort for the default match operation
Conclusions and Future Work

■ The COMA framework
  ■ Extensible matcher library, including novel reuse approach
  ■ Powerful combination scheme for both specifying match operations and constructing new matchers from existing ones

■ Comprehensive evaluation on real-world schemas
  ■ High effectiveness on large schemas
  ■ Reuse: essential improvement over no-reuse
  ■ Composite approach as THE solution for matcher combination

■ Future work
  ■ Matchers: more powerful reuse strategies, instance-based matchers
  ■ More intelligent combination strategies
  ■ Application to more real-world scenarios, esp. in bioinformatics