Towards a Benchmark for Instance Matching

Alfio Ferrara, Davide Lorusso, Stefano Montanelli, Gaia Varese

OM-2008
Karlsruhe, Germany – 26/10/2008
Summary

- Instance matching problem
  - Definition and issues
  - Applications

- The benchmark generation procedure
  - Overview of the procedure
  - Practical example
  - Heterogeneities classification and examples

- Benchmarks evaluation
  - Quality of the generated benchmarks

- Conclusions and future work
Instance Matching

- The problem
  - The goal is to detect instances that refer to the same real world entity
  - Mainly studied in the database literature
    ✓ Record linkage, entity recognition, merge-purge

- Applications
  - BOEMIE
    ✓ support for the population task
    ✓ help in the choice between different interpretations
  - OKKAM
    ✓ Web of entities, real world entities are univocally identified over the semantic web
Different issues

- Instance VS schema matching
  - Descriptions of the same entity VS concept with similar meaning
- Ontology VS database
  - More complex structures
  - Implicit data, need for reasoning techniques
  - Open world assumption

We developed an instance matching algorithm as a component of HMatch 2.0
Instance Matching Evaluation

- How to evaluate instance matching algorithms?
- Lack of evaluation data
  - Real data:
    - Need to find different descriptions of the same real-world objects
    - Need to find similar descriptions referred to different real-world objects
    - Need to manually create a mapping between all the couples of descriptions referred to the same real-world object
  - Artificial benchmark:
    - OAEI → Benchmark for concept matching
    - No benchmark for instance matching available
Our Solution

- Definition of a semi-automatic procedure for the generation of several different benchmarks
A real example: IMDb

- Reference ABox generation
  - Input:
    ✓ The reference TBox for the movie domain, built as a portion of the IMDb database
    ✓ A user query of the form: SELECT * FROM movies WHERE title LIKE ‘%Scarface%’

  - Automatic population:
    ✓ The selected data is extracted from IMDb and automatically translated as instances of the reference ABox

  - Output:
    ✓ The reference ABox contains 302 instances
The Modified ABoxes

*Modified ABoxes generation*

- **Input:**
  - The reference ABox
  - A user specification of all the modifications to be applied to the reference ABox for each modified ABox

- **Output:**
  - A set of modified ABoxes with expected alignments

*Each modified ABox simulates a different situation that can be found when comparing instances*

- We have defined three main classes of instance heterogeneities
Data Value Differences

- Errors in the data values
  - Typographical errors
    ✓ Scarface -> Scrface

- Values expressed with different formats
  - Dates
    ✓ 26/10/08 -> October 26th 2008
  - Person names
    ✓ Brian De Palma -> De Palma, B.
- Use of different levels of depth for properties representation
  ✓ I.E. The property value is designed as an independent instance
Structural Heterogeneity

- Use of different aggregation criteria for properties representation
  - I.E. different properties are concatenated or merged in a single property
- Missing values specification
  - I.E. one or more values are not defined
  - For the open world assumption we cannot consider the “null” value as a negative evidence in the comparison
Logical Heterogeneity

- **Instances of different subclasses of the same superclass**
  - *Tbox*: Movie ⊆ Item, Film ⊆ Item

- **Instances of different classes of a class hierarchy explicitly declared**
  - *Tbox*: Action ⊆ Movie

- **Instantiation on different classes of a class hierarchy implicitly declared**
Logical Heterogeneity

- Instances of disjoint classes
  - $Tbox$: Movie $\cap$ Product $\subseteq \bot$
  - $Ref. Abox$: movie$_1$ : Movie, $Mod. Abox$: movie$_1$ : Product

- Implicit values specification
  - $Ref. Abox$: movie$_1$ : Movie, (movie$_1$, “Scarface”) : HasTitle
  - $Mod. Abox$: movie$_1$ : Movie, movie$_1$ : ($\exists$HasTitle.“Scarface”)
How to evaluate the effectiveness of the generated benchmarks?
- We need a relevant number of different instance matching algorithms
- The quality of the benchmark is affected by
  - The source dataset: instances referring to different real world entities must not be too much similar
  - The level of modifications: the instance description must not be changed completely

The benchmark created from the IMDb dataset is available at http://islab.dico.unimi.it/iimb
Conclusion and Future Work

- A Semi-automatic procedure to create instance matching benchmarks
  - Doesn’t require to manually define the mappings
  - Can work with any domain and any dataset
  - Provides good flexibility with the combination of different classes of modifications

- Future work
  - Automatic population of the reference Abox through mappings between DB and Tbox
  - Easier interface to define the instance modifications