Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier

LIRMM

CNRS CENTRE NATIONAL DE LA RECHERCHE SCIENTIFIQUE

UM2 UNIVERSITÉ MONTPELLIER 2

ETH Eidgenössische Technische Hochschule Zürich Swiss Federal Institute of Technology Zurich

globis

XBenchMatch: a Benchmark for XML Schema Matching Tools

Fabien Duchateau[1], Zohra Bellahsene[1] and Ela Hunt[2]

[1]LIRMM, Univ. Montpellier 2-CNRS, [2]ETH Zurich

XBenchMatch: a Benchmark for XML Schema Matching Tool

XBenchMatch uses as

- **Input : the result of a schema matching algorithm (set of mappings and/or an integrated schema)**

- **Output : statistics about the quality of this input and the performance of the matching tool.**

- **A demo version of the prototype is available at**

***http://www.lirmm.fr/duchatea/XBenchMatch*.**

GOALS:

extensibility, portability, simplicity (ease of use), scalability, genericity, completeness

XBenchMatch FEATURES

- **Extensibility.**
  The benchmark should be able to be extended to include new measures and new format

- **Portability**.
  The benchmark should be OS-independent,

- **Simplicity.**
  since both end-users and schema matching experts are targeted by this benchmark tool.

- **Scalability** on two aspects
  creating new benchmark scenarii is an easy task. And a benchmark composed of many scenarii should be easy to build and evaluate.

- **Genericity.**
  It should work with most of the available matchers.

KIND OF EVALUATION

- **Quality of Mappings**
  - Measures (precison, recall, f-mesure)

- **Quality of Integrated Schema**
  - based on the use of the metrics

- **Performance of Matching Algorithms (time)**

MAPPING QUALITY MEASURES

- ***Given T****map* a set of derived mappings
- ***Given T****ex* a set of expert mappings

Precision = $|\boldsymbol{T}map \cap \boldsymbol{T}ex| \,/\, |\boldsymbol{T}map|$

Recall = $|\boldsymbol{T}map \cap \boldsymbol{T}ex| \,/\, |\boldsymbol{T}ex|$

Fmeasure = $(2 \cdot \text{precision} \cdot \text{recall}) \,/\, (\text{precision} + \text{recall})$

Integrated Schema Quality Measures

- Given an integrated schema Si, and an input schema Sg:

- **Backbone measure, BM,**
  - computes the size of the largest common subtree of Sg and Si (measured in nodes), seen against the background of the integrated schema Si.

    BM = | LCSub(Si, Sg) | / | Si |

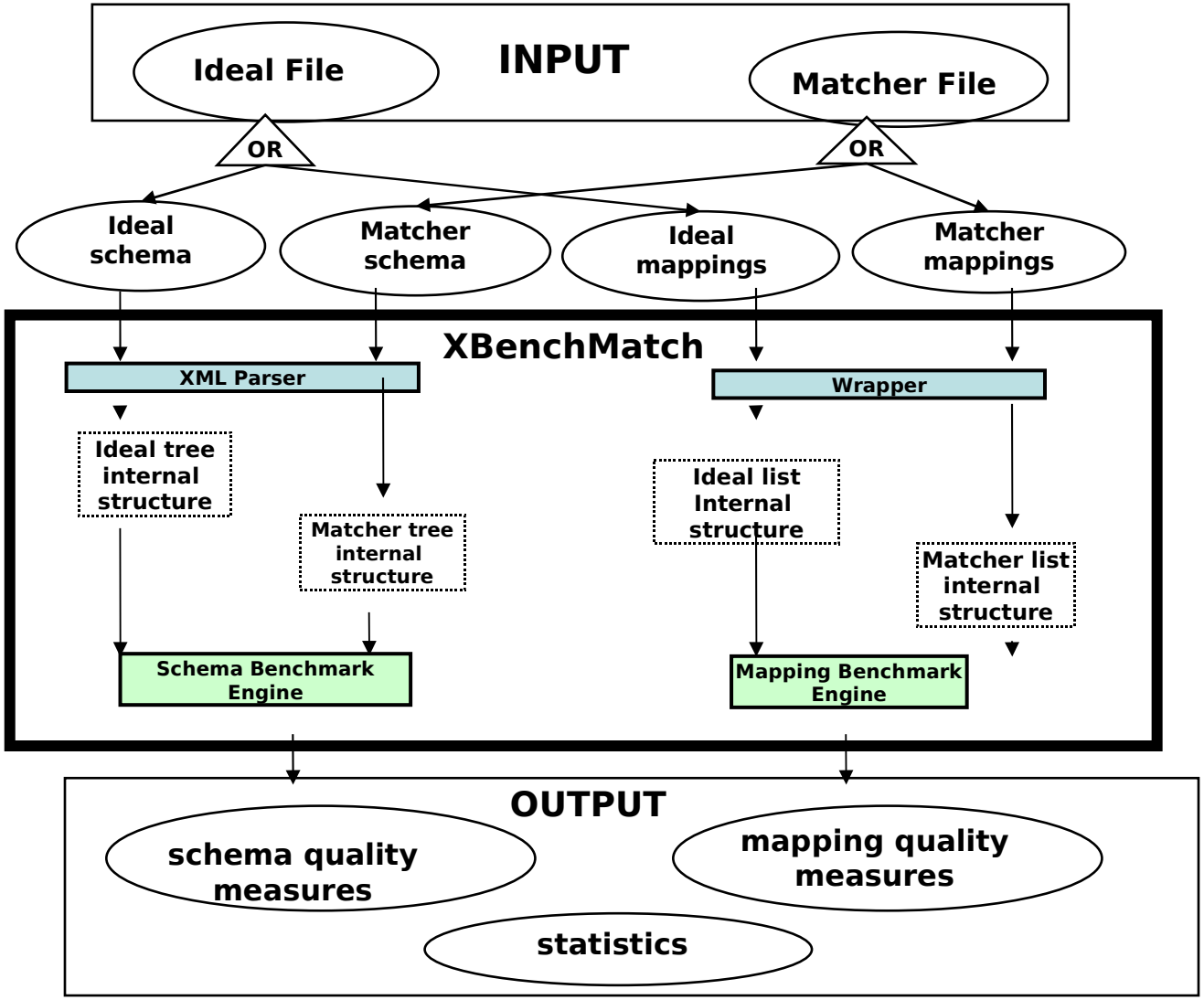
- **Structural overlap**
  - computes the number of nodes shared by Si and Sg and included in a common subtree. **Sub** is the set of all disjoint subtrees (each containing a minimum of two nodes) common to Si and Sg.
  - **kSub** is the total number of elements of all subtrees in Sub.

    StructuralOverlap = kSub / |Si|

- **Structural proximity**
- computes the number of subtrees common to Si and Sg.
- **o** is the number of elements in Si that are not included in any common subtree, **o = | Si | - kSub**.

  StructuralProximity = kSub / sqrt(|Si|x|Sub| + o)

XBenchMatch Prototype



Scenarii of schemas

- **SCHEMAS**
- Person schemas are small and strongly heterogeneous.
- Purchase orders, XCBL collection 3, demonstrate matching of a large schema to a smaller one.
- University course schemas are from Thalia [4].
- Biological schemas correspond to Uniprot protein DB, and GeneCards integrate data from over 100 databases.
- **TESTED MATCHERS**
- Porsche, COMA++ and Similarity Flooding.

Similarity Flooding (SF)

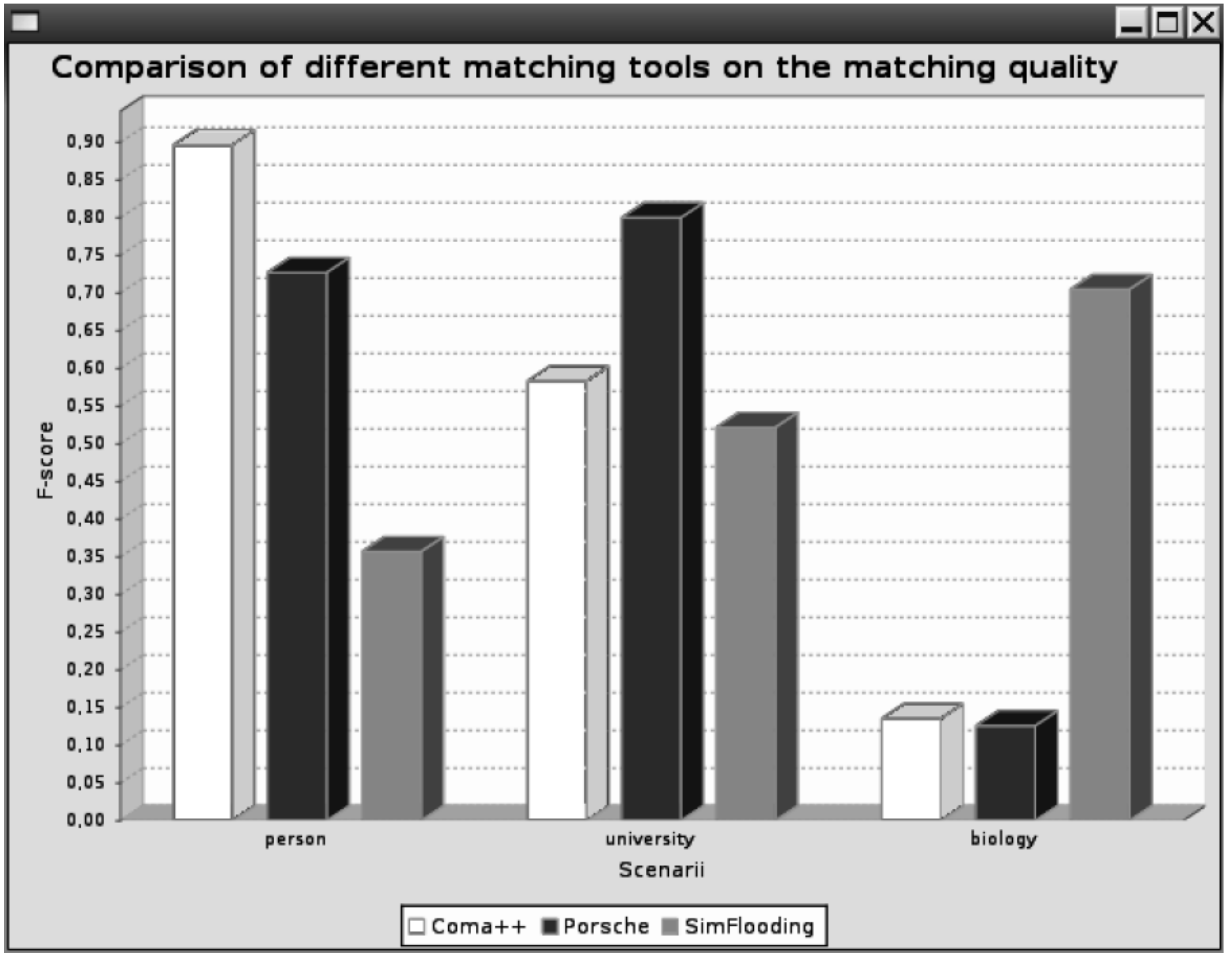
- Based on structural approaches.
- Input schemas are converted into directed labeled graphs and the aim is to find relationships between those graphs.
- Structural rule: two nodes from different schemas are considered similar if their adjacent neighbours are similar.
- When similar nodes are discovered, this similarity is then propagated to the adjacent nodes until there is no changes anymore.
- This algorithm mainly exploits the labels with some semantic-based algorithms, like String Matching, to determine the nodes to which it should propagate.
- Similarity Flooding does not give good results when labels are often identical, especially for polysemic terms. Thus involving wrong mappings to be discovered by propagation

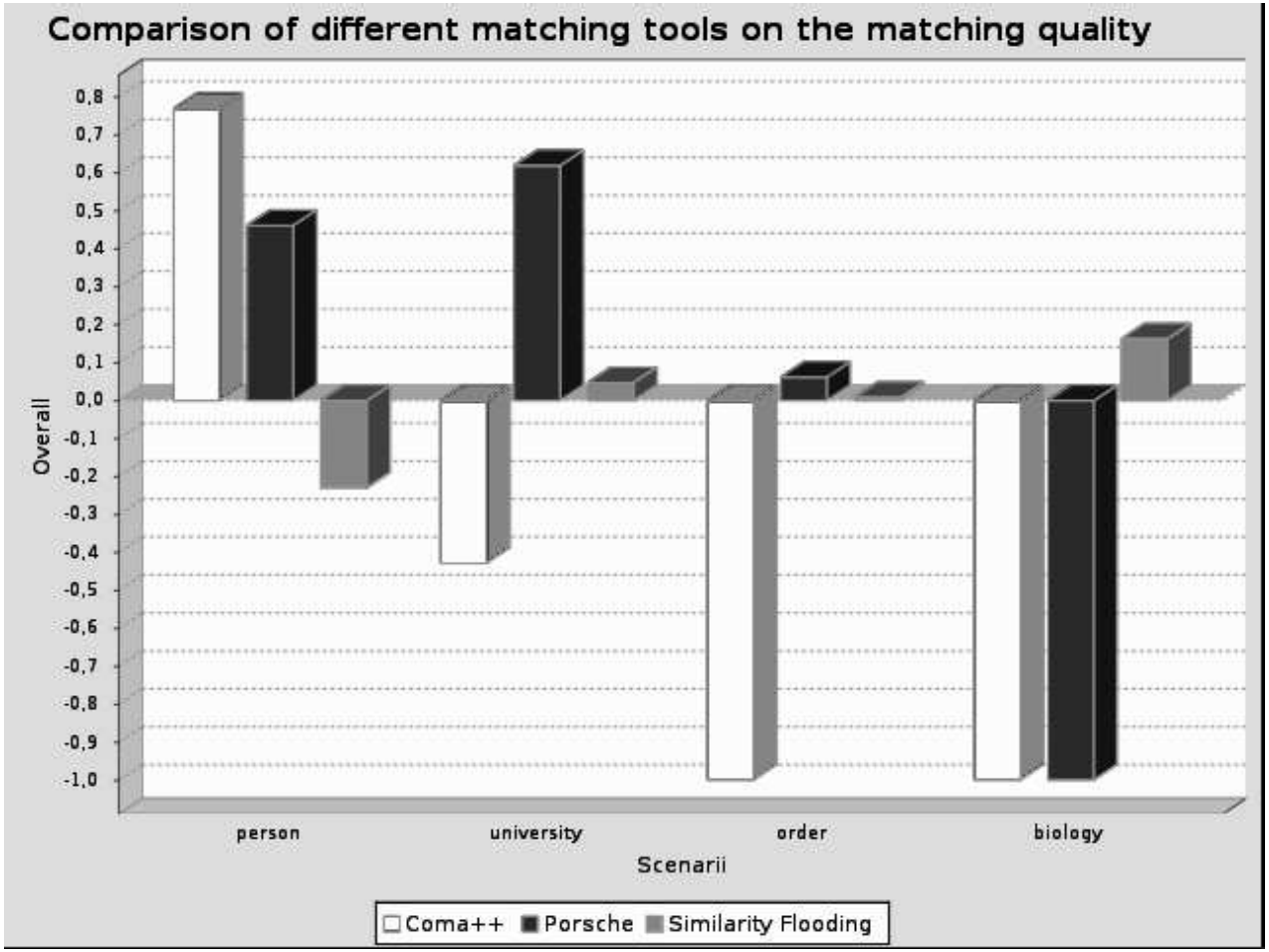
COMA/COMA++

- A generic, composite matcher
- It can process the relational, XML, RDF schemas as well as ontologies. Internally it converts the input schemas as trees for structural matching.
- For linguistic matching, it utilizes a user defined synonym and abbreviation tables like CUPID, along with n-gram name matchers.
- Similarity of pairs of elements is calculated into a similarity matrix.
- Uses 17 element level matchers. For each source element, elements with similarity higher then than threshold are displayed to the user for final selection.

	Person	University	Order	Biology
NB nodes (S_1 / S_2)	11 / 10	18 / 18	20 / 844	719 / 80
Avg NB of nodes	11	18	432	400
Max depth (S_1 / S_2)	4 / 4	5 / 3	3 / 3	7 / 3
NB of Mappings	5	15	10	57

Table 1: Summary of four evaluation scenarios.


Comparison of different matching tools on the matching quality
F-score
0,90
0,85
0,80
0,75
0,70
0,65
0,60
0,55
0,50
0,45
0,40
0,35
0,30
0,25
0,20
0,15
0,10
0,05
0,00
person
university
biology
Scenarii
Coma++
Porsche
SimFlooding


Comparison of different matching tools on the matching quality
Overall
0,8
0,7
0,6
0,5
0,4
0,3
0,2
0,1
0,0
-0,1
-0,2
-0,3
-0,4
-0,5
-0,6
-0,7
-0,8
-0,9
-1,0
person
university
order
biology
Scenarii
Coma++
Porsche
Similarity Flooding

Performances Results

	Person	University	Order	Biology
NB nodes (S1/S2)	11/10	18/18	20/844	719/80
BMatch	< 1	<1	<1	2
COMA++	< 1	<1	3	4
SF	<1	<1	2	4
PORSCHE	<1	<1	<1	<1