(Not) Yet Another Matcher

Fabien Duchateau*, Rémi Coletta*, Zohra Bellahsène*, Renée J. Miller ø
* LIRMM, Université Montpellier 2, ø University of Toronto

SCHEMA MATCHING

Context. Discovering correspondences between schema elements is a crucial task for data integration.

Applications of schema matching. Geographical and scientific information systems, B2B, Web Services composition, Semantic Web, etc.

Drawbacks of current schema matching tools:
- No flexibility (same match algorithm to combine similarity measures, i.e., aggregation function)
- Manual tuning
- Limited extensibility (in terms of similarity metrics or match algorithm)

Our contribution: ***YAM, a schema matcher factory***
- Automatic selection of a dedicated schema matcher for a given schema matching scenario and according to user inputs
- Automatic tuning of the dedicated schema matcher
- Capability for promoting either precision or recall

YAM IN DETAILS

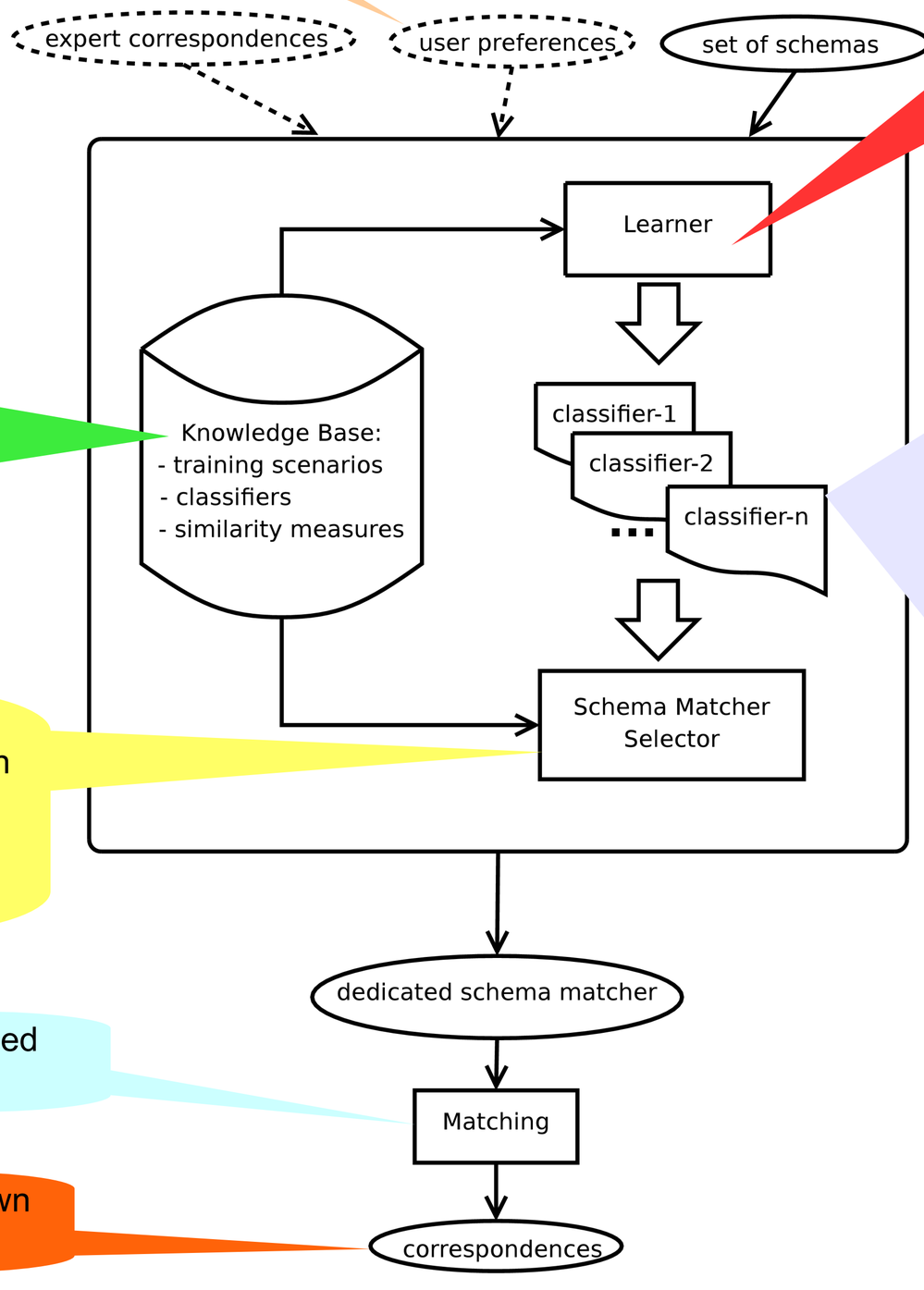
Schema matchers can be seen as machine learning classifiers (e.g. decision tree, aggregation functions, etc.) since they label each pair of schema elements either as a correspondence (match) or not.

In YAM, each schema matcher uses classifiers to combine similarity measures. As a schema matcher factory, it produces a set of schema matchers (based on different classifiers) and it selects the best one for a given scenario. This dedicated schema matcher is automatically tuned (in terms of thresholds, weights, etc.).

YAM's architecture

User inputs:
- Set of schemas (to be matched)
- User preferences, optional (e.g., preference between recall or precision)
- Expert correspondences, optional (between the schemas to be matched)

The **learner** trains all classifiers from the KB with training scenarios and expert correspondences (when provided). User preference between precision and recall is used by the learner.



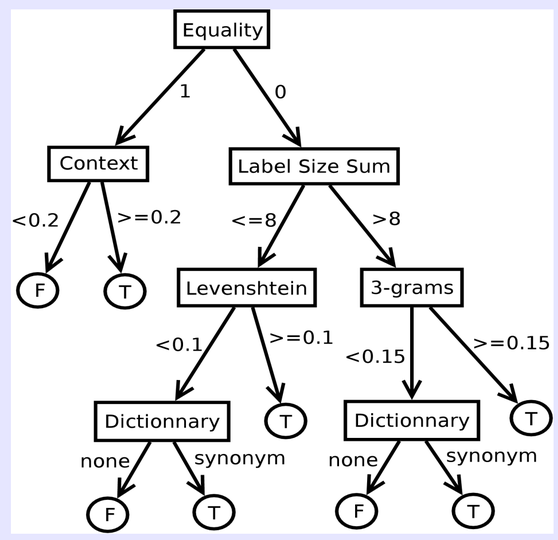
Knowledge Base currently contains:
- 20 classifiers from Weka
- 30 similarity measures (including those from Second String)
- 200 training scenarios (schemas with expert correspondences)

To **select** the dedicated schema matcher among all those learned, a cross-validation against training scenarios is performed. The classifier which obtains the highest f-measure is selected as the dedicated matcher.

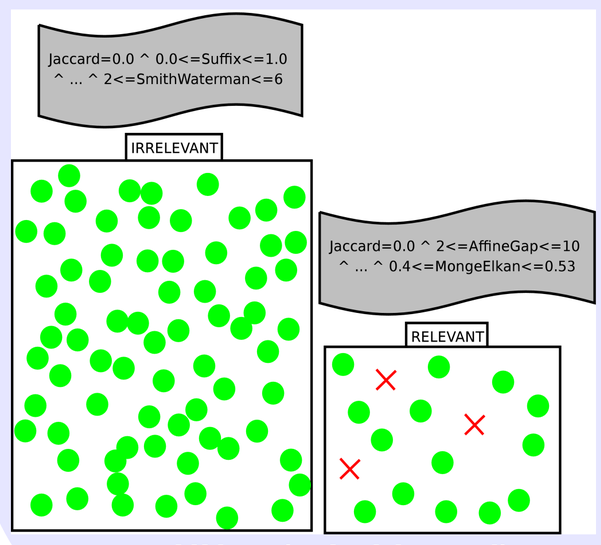
The dedicated schema matcher can be used for **matching** the input set of schemas.

Discovered correspondences are shown to the user thanks to a GUI.

Here are two examples of **trained (and tuned) classifiers**:



J48 decision tree



NNge (ruled-based)

(Not) Yet Another Matcher

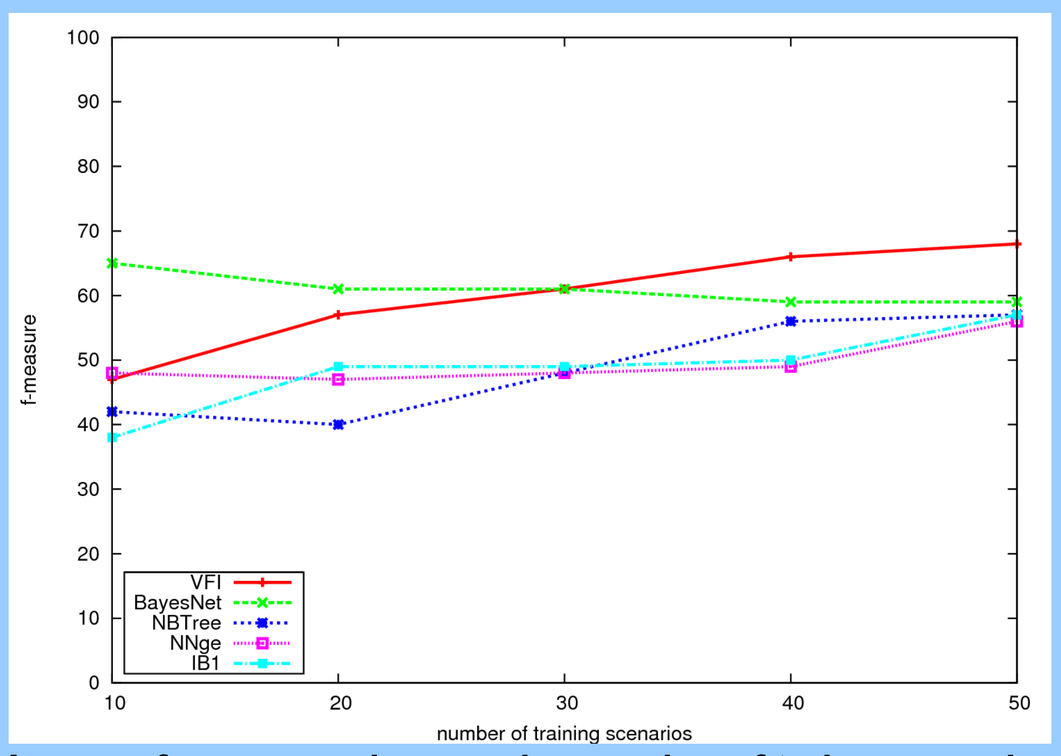
Laboratoire d'Informatique de Robotique et de Microélectronique de Montpellier (LIRMM)

Fabien Duchateau*, Rémi Coletta*, Zohra Bellahsène*, Renée J. Miller ø
* LIRMM, Université Montpellier 2, ø University of Toronto

University of Toronto

EXPERIMENTS REPORT

1) IMPACT OF TRAINING SCENARIOS

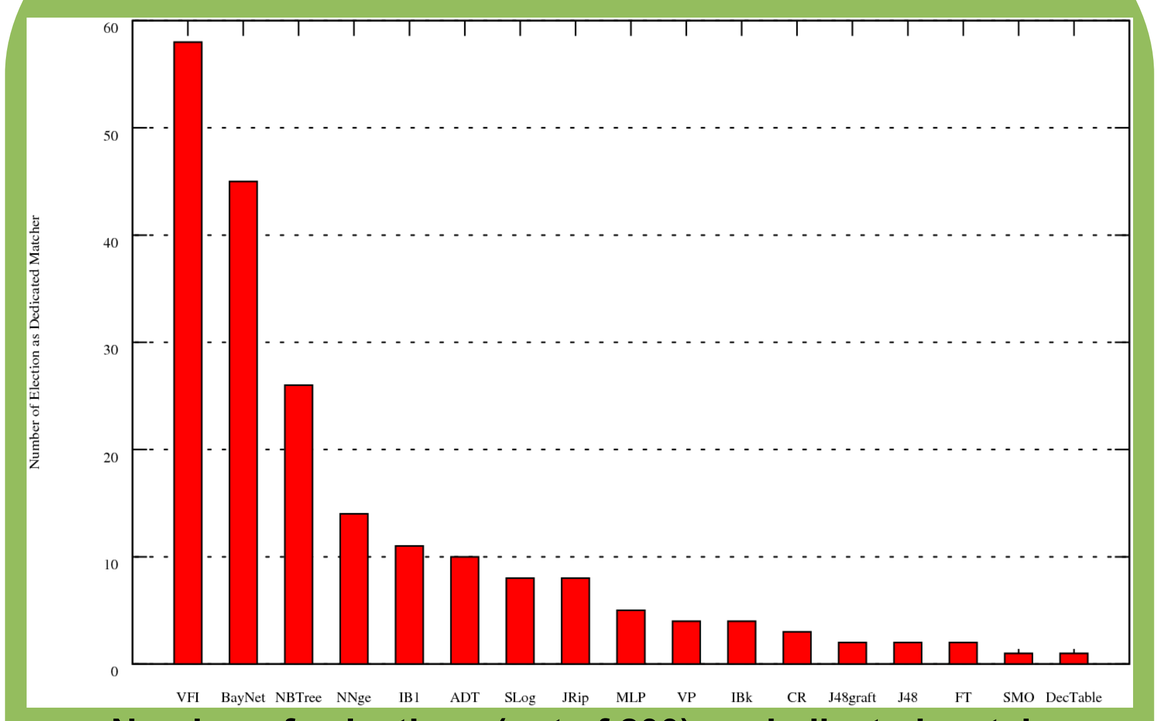


Average f-measure when varying number of traing scenarios

From empirical results (more than 11,000 experiment runs), we deduced the following table so that YAM is able to automatically select the number of training scenarios according to the classifier.

# training scenarios	Classifiers
20 and less	SLog, ADT, CR
20 to 30	J48, J48graft
30 to 50	NNge, JRip, DecTable, BayesNet, VP, FT
50 and more	VFI, IB1, IBk, SMO, NBTree, MLP

2) COMPARING GENERATED MATCHERS



Number of selections (out of 200) as dedicated matcher

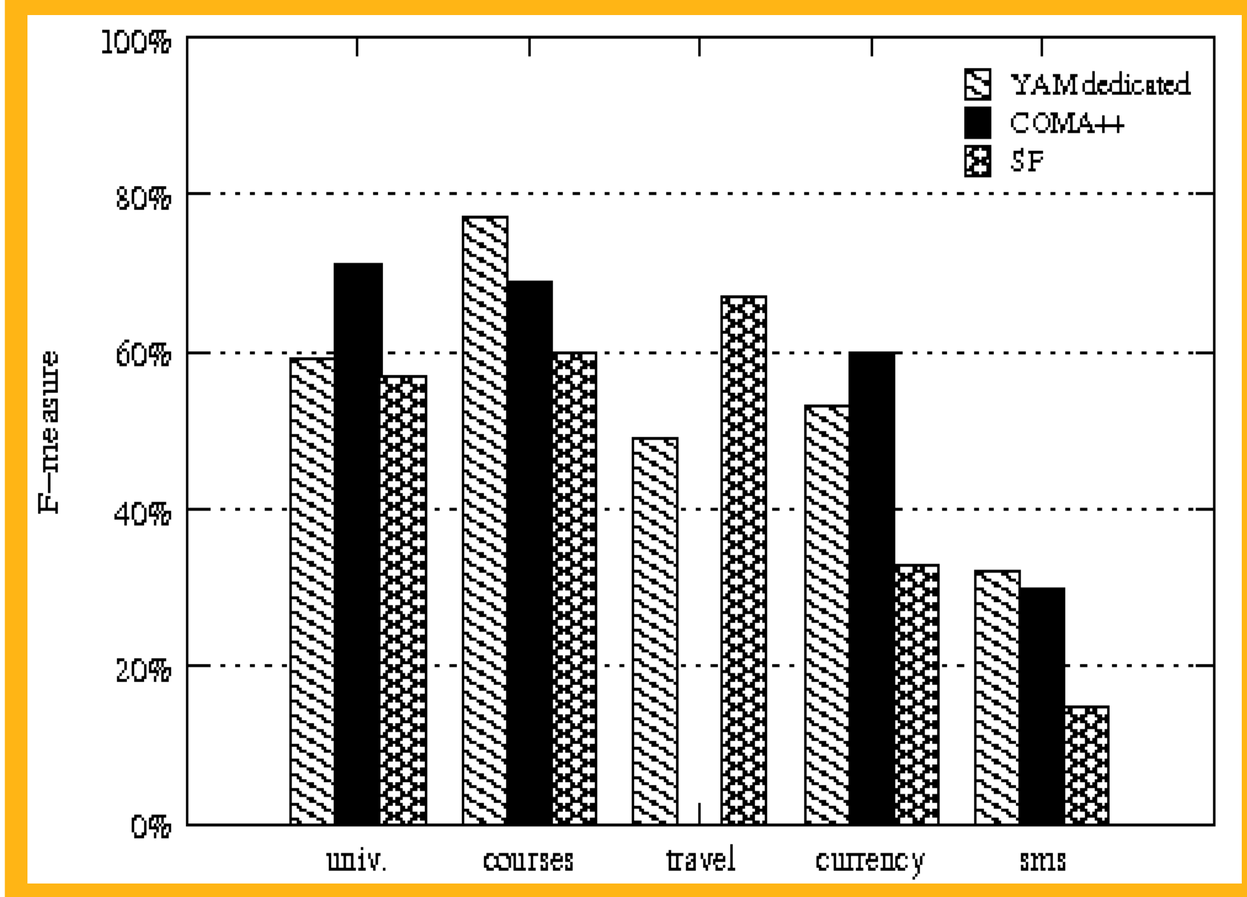
- VFI and BayesNet : selected in half of the scenarios
  => robust schema matchers
- CR or ADT : low average f-measure for the 200 scenarios but they are selected 3 and 10 times as dedicated matchers
  => effective for some specific scenarios
- SLog or MLP (aggregator functions) : only selected in a few scenarios
  => traditional schema matching tools based on these functions do not always provide the best matching quality

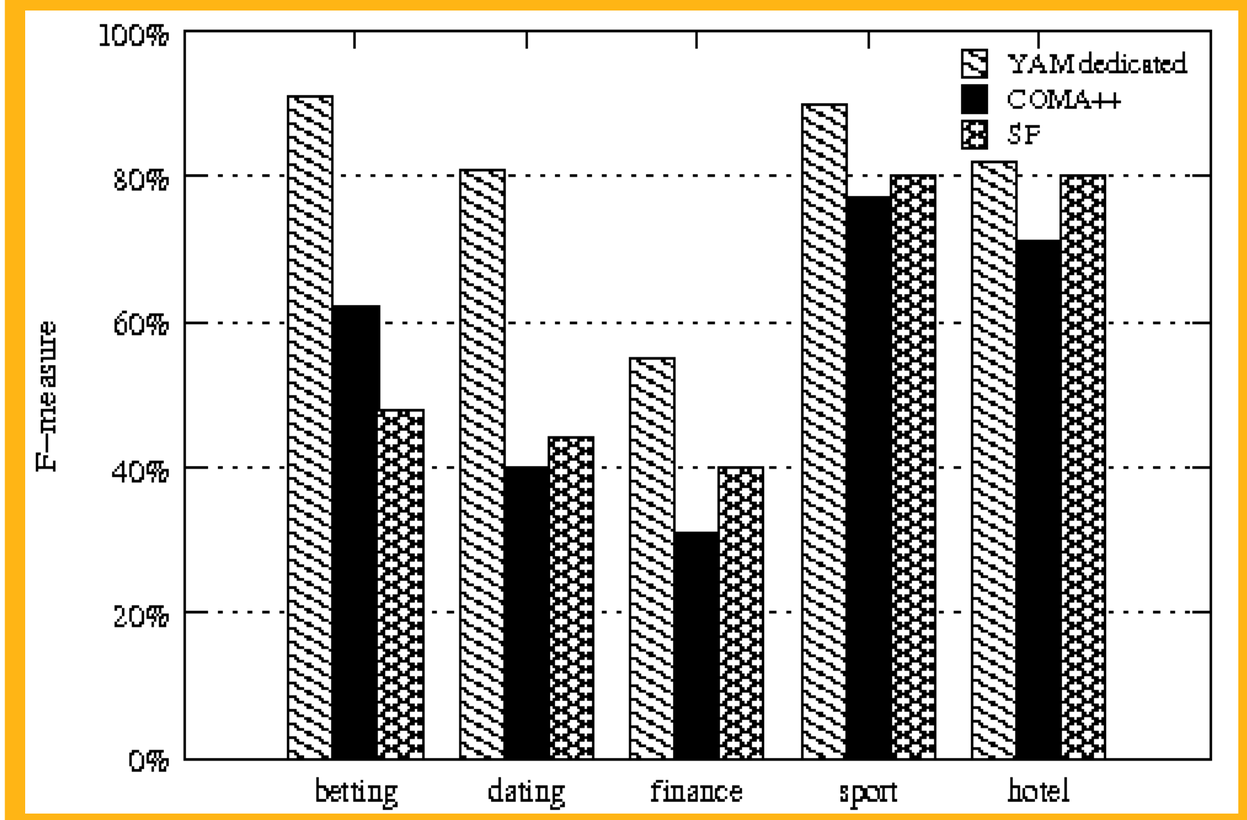
There is a real need for a schema matcher factory like YAM !

3) COMPARING WITH OTHER SCHEMA MATCHING TOOLS

Scenarios used for evaluation:
- *University courses* (Thalia benchmark)
- *Currency* and *sms* (web services)
- *University department* (literature)
- *Travel* (UIUC repository)
- *Betting*, *dating*, *finance*, *sport*, *hotel* (web forms)

Protocol:
- Training with webforms from the KB
- Average results from 200 runs
- Comparison with COMA++ and Similarity Flooding (SF)





F-measure achieved by schema matching tools for the 10 scenarios

- YAM obtains the highest f-measure in 7 scenarios
- YAM also achieves more than 8O% f-measure in 4 scenarios
- YAM obtains better results on webforms scenarios since it was trained with webforms
- Average f-measure for the 10 scenarios:
  - YAM = 67%
  - COMA++ = 51%
  - SF = 52%

YAM achieves better results since it has generated dedicated matchers based on various classifiers (VFI, BayesNet, J48, ...) while COMA++ and SF only rely on respectively an aggregation function and propagation