A Context-based Measure for Discovering Approximate Semantic Matching between Schema Elements

Fabien Duchateau, Zohra Bellahsène and Mathieu Roche

Laboratoire d'Informatique, de Robotique et de Microélectronique de Montpellier
Université Montpellier II, France

RCIS'07
Ouarzazate, Morocco

Table of Content

- Finding semantic correspondences between 2 schemas still a challenging issue
- Semi automatic matchers available based on several approaches (combination of terminological measures, structural rules, ...)

Motivations

Terminological measures are not sufficient, for example:

- mouse (computer device) and mouse (animal) ⇒ polysemia
- university and faculty ⇒ totally dissimilar labels

Structural measures have some drawbacks:

- propagating the benefit of irrelevant discovered matches to the neighbour nodes increases the discovering of more irrelevant matches
- not efficient with small schemas

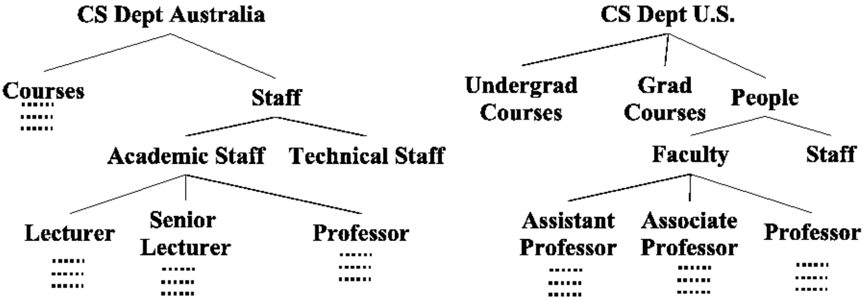

Figure: Two schemas from the university domain.

Our approach: Approxivect

Based on the work of [1], Approxivect evaluates the similarity between two terms from different schema trees. It has the following properties:

- it is based on the combination of terminological measures (Levenhstein and n-grams) and structural measures (cosine measure applied to contexts)
- it is both automatic and not language-dependent
- it does not rely on dictionaries or ontologies
- it provides an acceptable matching quality

Figure: XML schemas relative to university.

- 3grams(Courses, GradCourses) $= 0.2$
- Lev(Courses, GradCourses) $= 0.42$

$\Rightarrow$ **StringMatching(Courses, GradCourses) = 0.31**

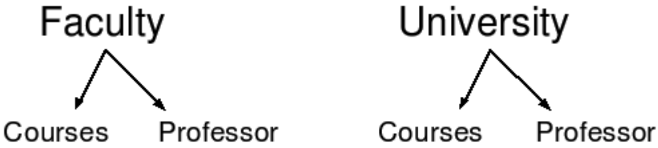

Figure: In the second schema, *Courses* replaces *GradCourses* due to StringMatching value.

- StringMatching(Faculty, University) = 0.002

- Context(Faculty) = Faculty, Courses, Professor
- Context(University) = University, Courses, Professor

⇒ **CosineMeasure(Context(Faculty), Context(University)) = 0.37**

Context of node n_c

- represents the most important neighbour nodes n_i for n_c
- each neighbour n_i is assigned a weight depending on the relationship n_c

$$\omega(n_c, n_i) = 1 + \frac{K}{\Delta d + |level(n_c) - level(n_a)| + |level(n_i) - level(n_a)|}$$

String Matching is the average between

- Levenhstein distance
- 3-grams

Discovering semantic similarities:

- String Matching between 2 node labels
- if above a given threshold, replacement of one of the label by the other.

Cosine Measure using context:

- due to replacements, the contexts of two nodes can be very similar

Similarity between two nodes

It is the best value between String Matching and Cosine Measure.

- NB_LEVELS restricts the context by limiting the number of levels
- MIN_WEIGHT restricts the context by keeping only nodes with a weight above this threshold
- REPLACE_THRESHOLD if the StringMatching between two node labels is above this replacement threshold, then one label is replaced by the other
- K represents the importance given to the context

Flexibility

These parameters allow more flexibility. Tuning them is required in some specific scenarii.

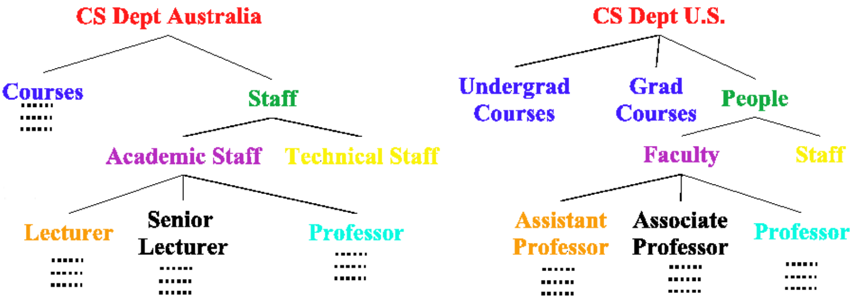

Figure: Mappings discovered by an expert between the schemas.

Element from schema 1	Element from schema 2	Similarity value	Relevance
Professor	Professor	1.0	+
CS Dept Australia	People	0.46	
Courses	Grad Courses	0.41	+
CS Dept Australia	CS Dept U.S.	0.36	+
Courses	Undergrad Courses	0.28	+
Academic Staff	Faculty	0.25	+
Staff	People	0.23	+
Technical Staff	Staff	0.21	+
Senior Lecturer	Associate Professor	0.16	+
...	...	...	...

Table: Approxivect similarity ranking between the two schemas

Element from schema 1	Element from schema 2	Similarity value	Relevance
Professor	Professor	0.53545463	+
Technical Staff	Staff	0.5300107	+
CS Dept Australia	CS Dept U.S.	0.52305263	+
Courses	Grad Courses	0.5041725	+
Courses	Undergrad Courses	0.5041725	+

Table: COMA++ discovered mappings between the two schemas

	Precision	Recall	F-measure
COMA++	1	0.56	0.72
Approxivect	0.62	0.89	0.73

Table: Results of COMA++ and Approxivect on the XML schemas

Note that Approxivect parameters are set to default. An optimal configuration enables to obtain a 0.82 F-measure.

COMA++ [2]

- combination of many terminological measures and a user-defined synonym table
- a matrix is built for each couple of elements and for each measure
- a strategy is applied to select the mappings
- mappings are modified and/or validated by the user

Similarity Flooding [3]

- a simple string matching algorithm to provide initial matchings
- structural rules and propagation to refine the matchings
- mappings are modified and/or validated by the user

An automatic schema matching approach

- based on the combination of terminological and structural measures
- with an acceptable quality of matching
- flexible thanks to the parameters

However

- tuning is not automatic, but some tools could handle this step (eTuner)
- more heterogeneity in the experiments

Ongoing work

- performance aspect

- T. YiFei, "Using contextual and lexical information to map terms of schemas," Master's thesis, Research Master - Université de Montpellier 2, 2006.
- D. Aumueller, H. Do, S. Massmann, and E. Rahm, "Schema and ontology matching with coma++," in *SIGMOD 2005*, 2005.
- S. Melnik, H. G. Molina, and E. Rahm, "Similarity flooding: A versatile graph matching algorithm and its application to schema matching," in *Proc. of the International Conference on Data Engineering (ICDE'02)*, 2002.