YAM++ - A combination of graph matching and machine learning approach to ontology alignment task

DuyHoa Ngo, Zohra Bellahsene

Amir Naseri
Knowledge Engineering Group

Introduction

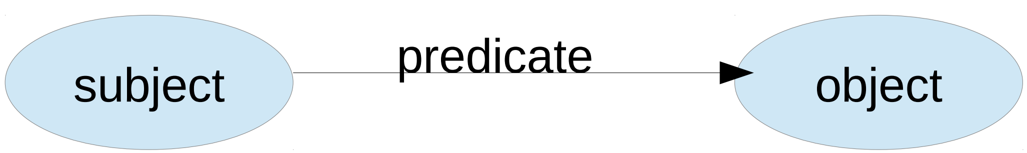
An Ontology is a

formal specification	→	**machine processable**
of a shared	→	**has reached a consensus**
conceptualization	→	**describes terms**
of a domain of interest	→	**of a certain topic**

(Gruber 1993)

An ontology can be represented as an RDF graph

- A set of triples in the following form:



Introduction

Providing semantic vocabularies

- Which make domain knowledge available to be exchanged and interpreted among information systems

Heterogeneity of ontologies

- Decentralized nature of the semantic web
- Different developer created ontologies describing the same domain differently
  - In domain of organizing conferences:
    - Participant (in confOf.owl)
    - Conference_Participant (in ekaw.owl)
    - Attendee (in edas.owl)
- An explosion in number of ontologies

Introduction

TECHNISCHE UNIVERSITÄT DARMSTADT

The heterogeneity consequences

- Terms variations
- Ambiguity in entity interpretation

Finding correspondences within different ontologies (ontology matching) as the solution

- Reaching a homogeneous view
- Enabling information systems to work effectively

Background

Formal definition of ontology

- O = <C, P, T, I, Hc, Hp, A>
- C: set of classes (concepts)
- P: set of properties consisting of object properties (OP) and data properties (DP)
- T: set of datatypes
- I: set of instances (individuals)
- Hc: defines the hierarchical relationshpis between classes
- Hp: defines the hierarchical relationshpis between properties
- A: set of axioms describing the semantic information, such as logical definition and interpretation of classes and properties

Background

Entities are the fundamental building blocks of OWL 2 ontologies

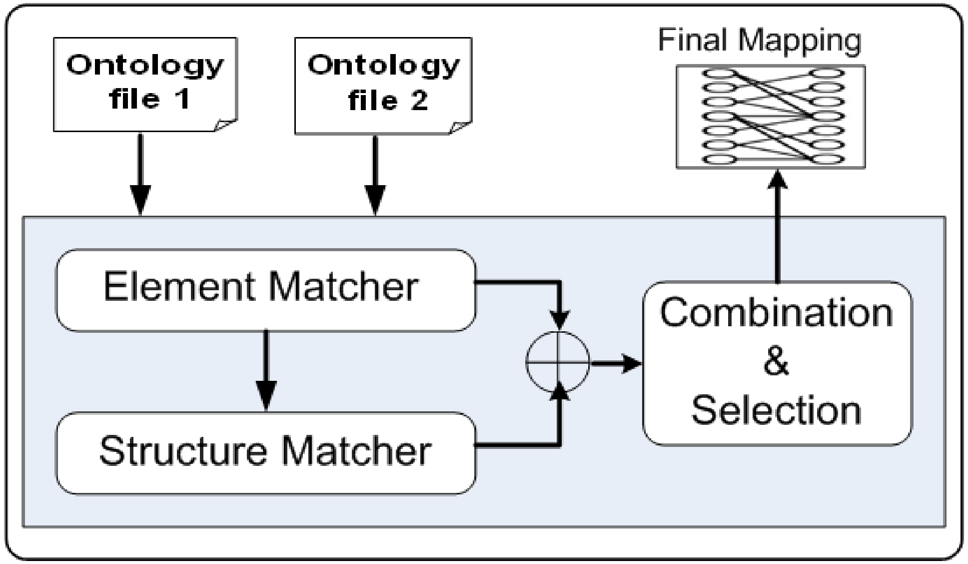
- Classes, object properties, data properties, and named individuals are entities
  - Scheme entities
    - Classes, object properties, and data properties
  - Data entities
    - The rest

A correspondence or a match m is defined

- m = <e, e', r, k>
  - e and e': entities in O and O'
  - r: relation (equivalent for match)
  - k: degree of confidence of relation (k → [0, 1] : 1 means we have a match)

An alignment is a set of correspondences between two or more ontologies

YAM++ Approach



Element matcher uses terminological feature (textual info)

Structure matcher uses structural feature

Combination & selection generates the final mappings

Motivating Example

TECHNISCHE UNIVERSITÄT DARMSTADT

Two university ontologies, namely, source.owl and target.owl

Concept hierarchies

object properties

data properties

Element Matcher

Machine learning approach to combine the selected metrics

- Each pair of entities as a learning object X
- Each similarity metric as X's attribute
- Each similarity score as attribute value
- Generating training data from gold standard dataset
  - Gold standard data are a pair of ontologies with an alignment provided by domain experts

Freeing user from setting the parameters to combine different similarity metrics

Element Matcher

Similarity metric groups related to different types of terminological heterogeneity

- Edit-based group
  - Considering two labels without dividing them into tokens
  - Suitable for cases such as: “firstname” vs. “First.Name”
- Token-based group
  - Splitting labels into set of tokens and computing the similarity between those sets
  - Suitable for cases such as: “Chair_PC” vs. “PC_chair”
- Hybrid-based group
  - An extension of the token-based, each internal similarity metric as a combination of an edit- and a language-based metric
  - Ignoring stop words
  - Suitable for cases such as: “ConferenceDinner” vs. “Conference_Banquet”

Element Matcher

Profile-based

- For each entity 3 types of context profile are produced
    1. Individual: all annotation (labels, comments) of an entity
    2. Semantic: combination of individual profile of an entity with its parents, children, domain, etc.
    3. External: combination of textual annotation (labels, comments and properties' value) of all instances belonging to an entity

Group Name	List of Metrics
Edit-based	Levenstein, ISUB
Token-based	Qgrams, TokLev
Hybrid-based	HybLinISUB, HybWPLev
Profile-based	MaxContext

Element Matcher

TECHNISCHE UNIVERSITÄT DARMSTADT

Employing a decision tree model (J48) for classification

- J48 is reused from the data mining framework Weka

Classification problem for the motivating example

- Training data is the gold standard datasets from Benchmark 2009
- Classification metrics are Levenstein, Qgrams, and HybLinISUB

Instances	Hyb.	Lev.	QGs	Class
Researcher | Researcheur	0.00	0.91	0.80	?
Teacher | Lecturer	0.77	0.37	0.21	?
Manager | Director	1.00	0.13	0.10	?
Teach | teaching	1.00	0.63	0.59	?

Element Matcher

Non-leaf nodes are similarity metrics

Leaves, illustrated with round rectangles, are 0 or 1, implying whether there is a match or not

For example Researcher | Researcheur:

- 1 → 3 → 5 → 6 → 8 → 10 → leaf (1.0)

Hyb.	Lev.	QGs	Class
0.00	0.91	0.80	?

Classifier output

```
Instances:     7959
Attributes:    4
               HybLinISUB
               Levenshtein
               QGrams
               CLASS
Test mode:     10-fold cross-validation

=== Classifier model (full training set) ===

J48 pruned tree
------------------

01  HybLinISUB    <= 0.891794
02  |   QGrams <= 0.258065: 0.0
03  |   QGrams > 0.258065
04  |   |   QGrams <= 0.645161: 0.0
05  |   |   QGrams > 0.645161
06  |   |   |   HybLinISUB    <= 0.576275
07  |   |   |   |   QGrams <= 0.7: 1.0
08  |   |   |   |   QGrams > 0.7
09  |   |   |   |   |   Levenshtein <= 0.888889: 0.0
10  |   |   |   |   |   Levenshtein > 0.888889: 1.0
11  |   |   |   HybLinISUB    > 0.576275: 0.0
12  HybLinISUB    > 0.891794
13  |   QGrams <= 0.785714
14  |   |   Levenshtein <= 0.111111: 0.0
15  |   |   Levenshtein > 0.111111
16  |   |   |   Levenshtein <= 0.785714: 1.0
17  |   |   |   Levenshtein > 0.785714: 0.0
18  |   QGrams > 0.785714: 1.0

Number of Leaves  :     10

Size of the tree :      19
```

Structure Matcher

Making use of similarity propagation (SP) method

- Inspired by flooding algorithm

Transformation of ontologies into directed labeled graph, with edges in the following format (1. and 2. row in algorithm 1):

- <sourceNode, edgeLabel, targetNode>

Generating a pairwise connectivity graph (PCG) by merging edges with the same labels (3. row in algorithm 1)

- Suppose G1 and G2 are two graphs after the transformation
  - $((x, y), p, (x', y')) \in PCG \quad \Leftrightarrow \quad (x, p, x') \in G1 \;\&\; (y, p, y') \in G2$
  - A part of the similarity of two nodes is propagated to their neighbors which are connected by the same relation

Structure Matcher

Algorithm 1: SP

- Input: O_1, O_2: ontologies

  $M_0 = \{(e_1, e_2, \equiv, w_0)\}$: initial mappings
- Output: $M = \{(e_1, e_2, \equiv, w_1)\}$: result mappings

1. $G_1 \leftarrow$ Transform (O_1)
2. $G_2 \leftarrow$ Transform (O_2)
3. PCG $\leftarrow$ Merge (G_1, G_2)
4. IPG $\leftarrow$ Initiate (PCG, Weighted, M_0)
5. Propagation (IPG, Normalized)
6. M $\leftarrow$ Filter (IPG, θ_s)

Structure Matcher

Edges in the PCG obtain weight values from the Weighted function

Nodes are assigned similarity values from initial mapping M_0

After initiating PCG becomes an induced propagation graph (IPG) (4. row in algorithm 1)

In the Propagation method (5. row in algorithm 1), similarity scores in nodes are updated, whereas the weights of edges are not changed

At the end, a filter with threshold θ_s is used to produce the final result

Structure Matcher

Concentration on the transformation of an ontology, represented as an RDF graph, into directed labeled graph

Disadvantages of RDF graphs

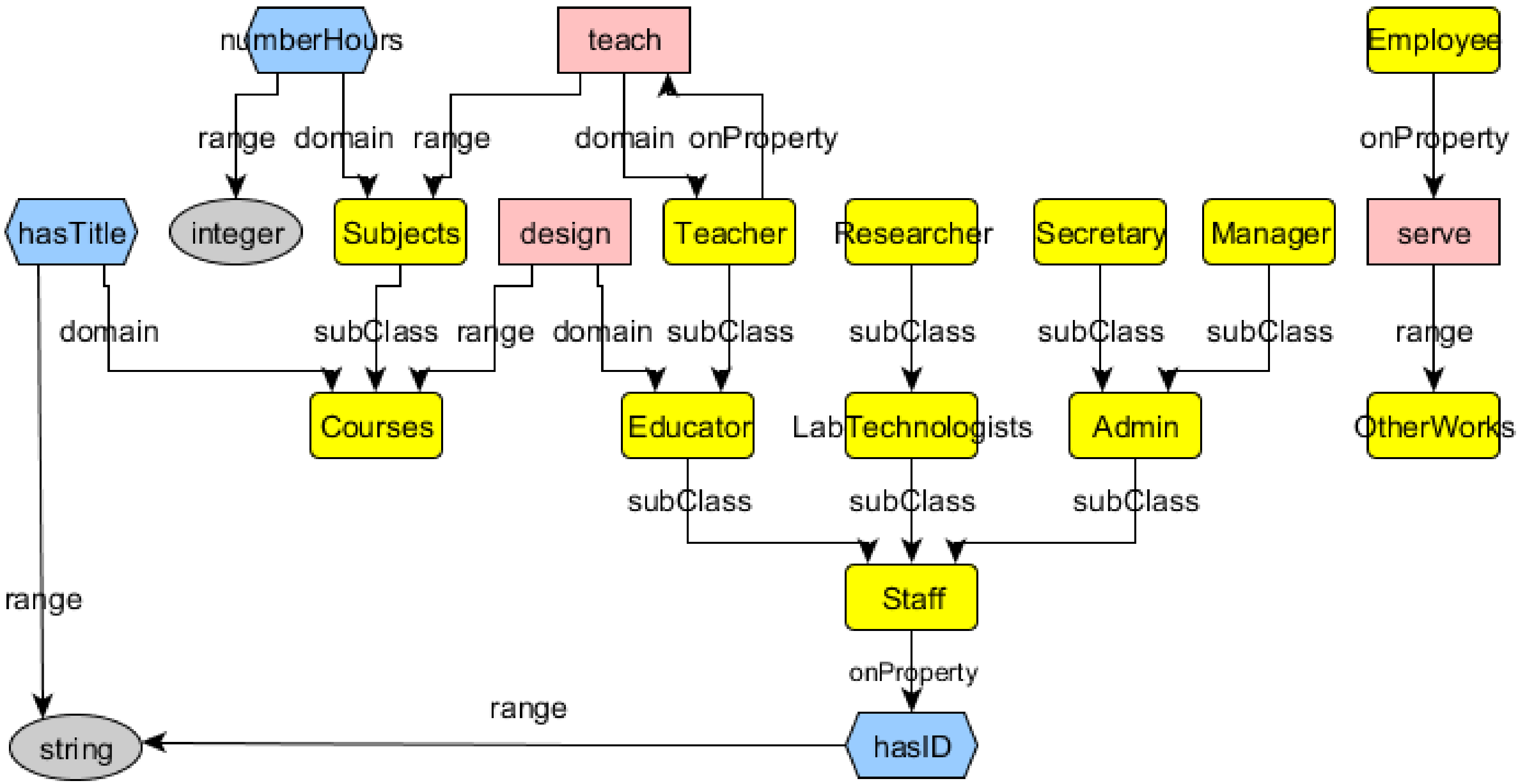
- Generating redundant nodes in PCG
  - e.g., with the label rdf : type, we will have many node compounds of the concept in the first ontology connected with the properties of the second one
- Generating incorrect mapping candidates
  - e.g., <Courses, rdf : type, Class> with <Director, rdf : type, Class>
- Problem of having anonymous (blank) nodes in the RDF graphs, since the similarity between those nodes cannot be calculated

Structure Matcher

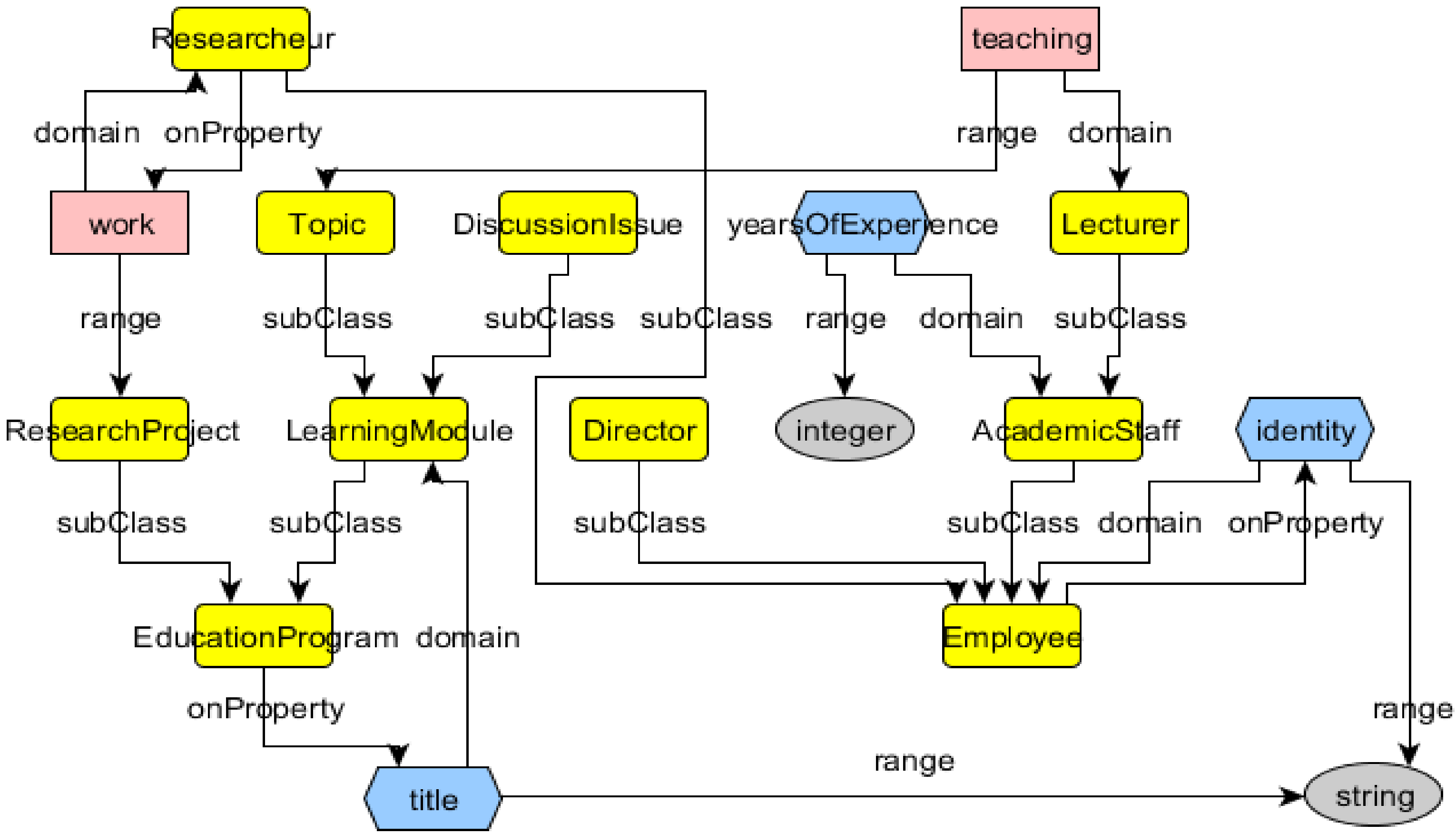
Employed approach for transformation into directed labeled graph

- Conversion of each semantic relation between entities to a directed edge with a predefined label
- Source and target node are ontology entities or primitive data types
- Semantic meaning of an edge is illustrated by the edge label belonging to one of the five types:
  - subClass, subProperty, onProperty, domain, range

Structure Matcher



Structure Matcher



Mappings Combination

Element matcher

- Names (labels) of entities

Structure matcher

- Semantic relation of an entity with other entities

Assumption

- Results of element and structure matcher are complement

$M_{element}$ and $M_{structure}$ are set of mappings found by element and structure matcher respectively (inputs of algorithm 2)

Mappings Combination

Algorithm 2: Produce Final Mappings

- Input: $M_{element} = \{(e_i, e_j, \equiv, 1)\}$

  $M_{structure} = \{(e_p, e_q, \equiv, c_s)\ ,\ \ c_s \in (\theta_s, 1]\}$
- Output: $M_{final} = \{(e_1, e_2, \equiv, c)\ ,\ \ c \in [0, 1]\}$

1. $\theta \leftarrow \min(m.c_s) : \quad m \in M_{structure} \cap M_{element}$
2. $M \leftarrow \text{WeightedSum}\ (M_{element}, \theta, M_{structure}, (1 - \theta))$
3. Threshold $\leftarrow \theta$
4. $M_{final} \leftarrow$ GreedySelection (M, threshold)
5. RemoveInconsistent (M_{final})
6. Return M_{final}

Mappings Combination

$M_{overlap}$ **= {se1, se2, se3}**

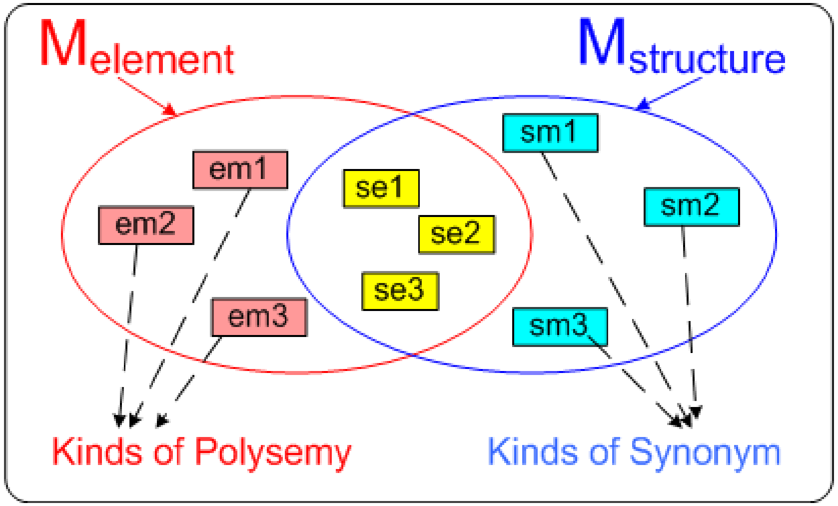
- *The most desired mapping*

$M_{structure}$ **= {sm1, sm2, sm3}**

- *Entities with different names, but similar semantic relations*

$M_{element}$ **= {em1, em2, em3}**

- *Entities with similar names, but different semantic relations*



Mappings Combination

Threshold θ is the minimum value of the structural similarity (1. row in algorithm 2)

- Assumption: all mappings with a higher similarity value than θ are considered as correct

The probability of correctness of mappings in $M_{element}$ is smaller than the probability of correctness of mappings in $M_{structure}$

WeightedSum's output is the union of mappings in $M_{element}$ and $M_{structure}$ with updated similarity scores *(2. row in algorithm 2)*

Mappings Combination

TECHNISCHE UNIVERSITÄT DARMSTADT

Greedy selection

- Sorting the mappings in descending order of the confidence value
- In each iteration, extracting the first (with highest score) mapping
- If the extracted mapping greater than or equal to threshold
  - Adding it to the final mappings
- Else
  - Return the final mappings

- Finding all mappings in M (output of weighted sum), whose source or target entities are the same with ones in the extracted mapping

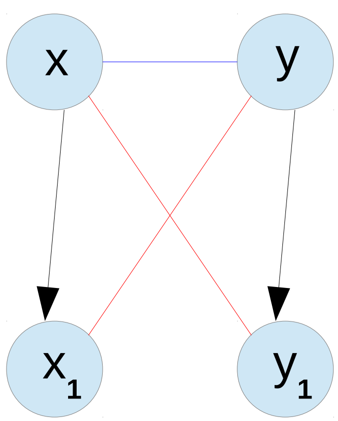
Mappings Combination

Mapping refinement

- If $\{(x, y), (x, y_1), (x_1, y)\} \in A$ and $x_1 \in Desc(x)$, $y_1 \in Desc(y) \rightarrow$

  (x, y_1), (x_1, y) are inconsistent and will be removed

  - Desc (e): all descendants of entity e
  - Criss-cross mappings



Mappings Combination

Mapping refinement

- If $(p_1, p_2) \in A$ and $\{ \text{Doms}(p_1) \times \text{Doms}(p_2) \cap A = \emptyset \}$ and $\{ \text{Rans}(p_1) \times \text{Rans}(p_2) \cap A = \emptyset \} \rightarrow$

  (p_1, p_2) is inconsistent and will be removed

  - Doms (p): all domains of property p
  - Rans (p): all ranges of property p
  - Some pairs of concepts are in greedy selection removed
    - Some properties lost their domain and range

Evaluation

Five experiments

- Comparison of matching performance of the ML combination vs. other combination methods
- Comparison of matching performance of the SP method vs. other structural methods
- Comparison of matching performance of the dynamic weighted sum (DWS) method vs. other element and structure combination methods
- Study the effect of mapping refinement
- Comparison of matching performance of YAM++ approach vs. other participants in OAEI competition

Evaluation

Comparison of matching performance of ML vs. other combination methods

- Weighted average with local confidence (LC) used in AgreementMaker
- Harmony-based adaptive weighted aggregation (HW)
  - Far better other aggregation functions like, max, min, and average
- Four individual matcher in four different groups with the best results

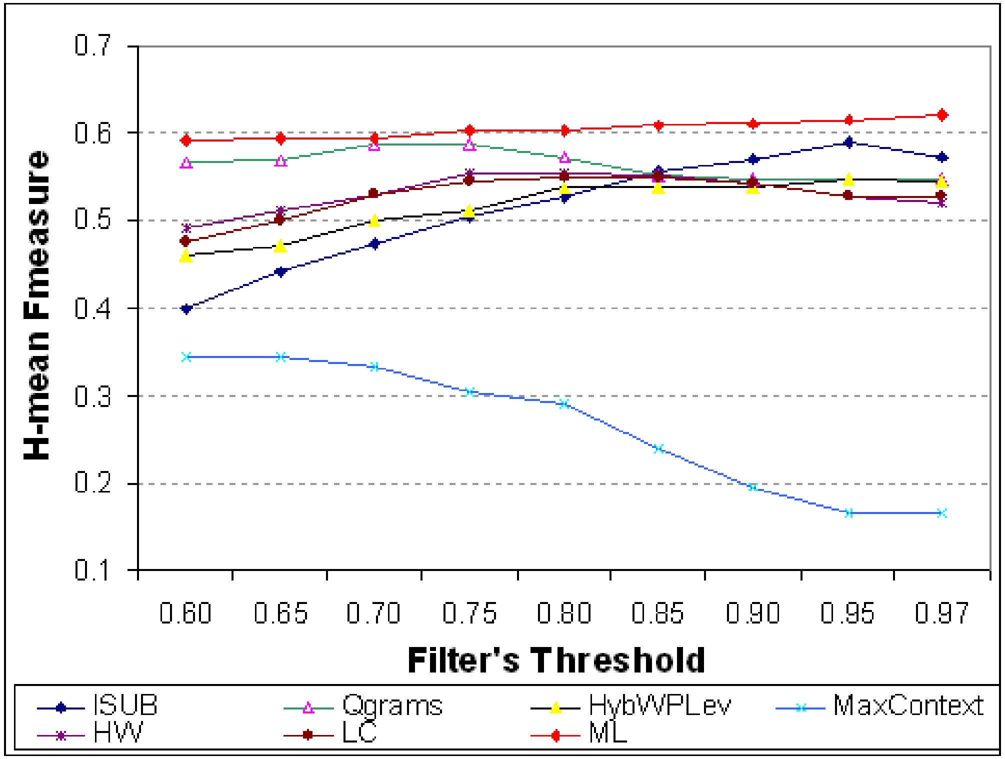
- Conference dataset with 15 real world ontologies in conference organization domain
- ML, freeing user from setting the threshold

Evaluation

$H(p) = (\Sigma |C_i|) \;/\; (\Sigma |A_i|)$,

$H(r) = (\Sigma |C_i|) \;/\; (\Sigma |R_i|)$,

$H(f_m) = (2 * H_p * H_r) \;/\; (H_p + H_r)$.



$|C_i|$: number of correct mappings

$|A_i|$: total number of mappings of a matching system

$|R_i|$: number of reference mappings produced by an expert domain

Evaluation

Usage of gold standard data set

- Ensuring the independence of training and test data
- 10 times with different data sets for having different training data
- Sorting H-mean values of 10 executions

ML better than HW and LC, since

- Does not employ linear arithmetic function, instead finding combination rules and constraint from training data
- Recognizing (Co-author ≡ Contribution_co_author), since
  - Finding similar pattern in training data, like (payment ≡ means_of_payment)

ML better than individual matchers

- Make use of more features

Evaluation

Comparison of matching performance of DWS vs other combination methods

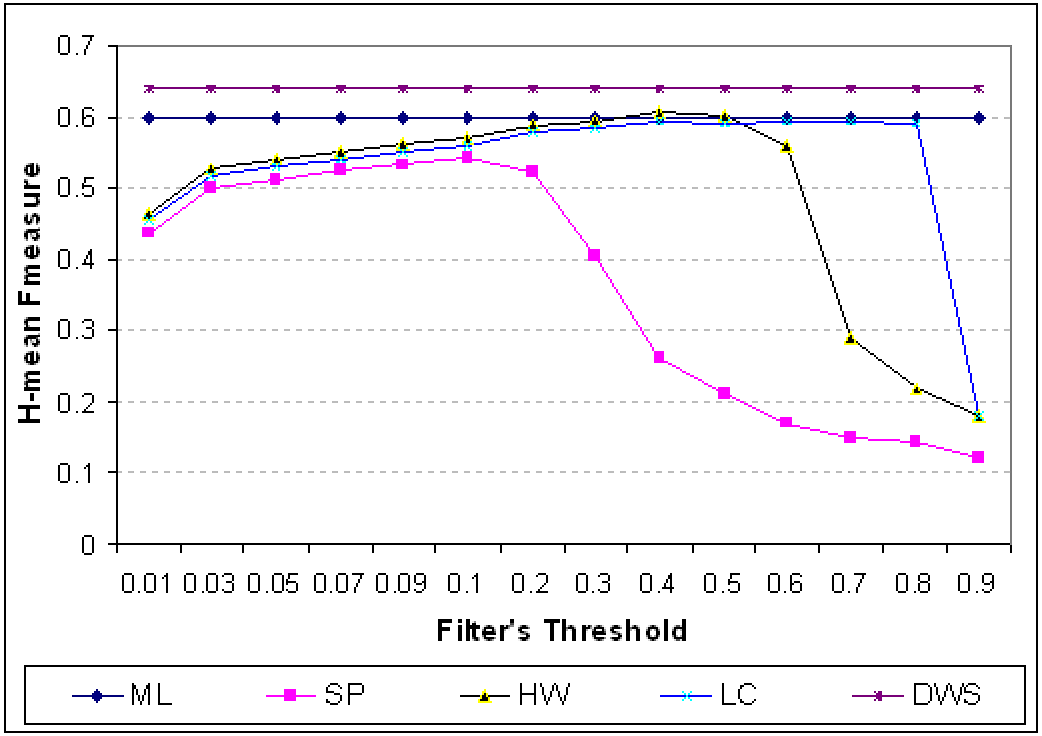
- Element matcher generates a matching result (ML)
- Structure matcher uses ML and generates another matching result (SP)
- Three weighted sum methods HW, LC and DWS combine ML and SP
- Make use of 21 real test cases of Conference data set
  - Ontologies of theses test cases are very different in terminology and structure
- A filter's threshold is used to select the final mappings for SP, HW and LC
- Similarity scores in ML are 1
- DWS computes automatically the threshold

Evaluation

TECHNISCHE UNIVERSITÄT DARMSTADT

SP covers many incorrect mappings (threshold 0.1)

DWS advantage of dynamic setting of weights and filter's threshold



Evaluation

Comparison with OAEI participants

- OAEI campaign in 2011, Benchmark track

	2010		2011							
	original		original		biblio		ekaw		finance	
System	Fmeas.	Top-5	Fmeas.	Top-5	Fmeas.	Top-5	Fmeas.	Top-5	Fmeas.	Top-5
ASMOV	.93	√								
AgrMaker	.89	√	.88	√	x		.71		.78	
Aroma	.59		.78		.76	√	.68		.70	√
CSA			.84	√	.83	√	.73	√	.79	√
CIDER			.76		.74		.70		.67	√
CODI	.55		.80	√	.75	√	.73	√	x	
edna	.51		.52		.51		.51		.50	
LDOA			.47		.46		.52		T	
Lily			.76		.77	√	.70		T	
LogMap			.60		.57		.66		x	
MaasMtch			.59		.58		.61		.61	√
MapEVO			.41		.37		.33		.20	
MapPSO	.61		.50		.48		.63		.14	
MapSSS			.84	√	x		.78	√	T	
Optima			.64		.65		.56		T	
YAM++			.87	√	.86	√	.75	√	T	

Evaluation

In Conference track, computation of $F_{measure}$ in 3 ways

- $F_{0.5}$: recall more important than precision
- F_1: recall and precision equally important
- F_2: precision more important than recall

Matcher	Prec.	$F_{0.5}$Meas.	Rec.	Prec.	F_1Meas.	Rec.	Prec.	F_2Meas.	Rec.
YAM++	.8	.73	.53	.78	**.65**	.56	.78	.59	.56
CODI	.74	.7	.57	.74	.64	.57	.74	.6	.57
LogMap	.85	**.75**	.5	.84	.63	.5	.84	.54	.5
AgrMaker	.8	.69	.44	.65	.62	.59	.58	**.61**	.62
$BaseLine_2$	**.79**	**.7**	**.47**	**.79**	**.59**	**.47**	**.79**	**.51**	**.47**
MaasMtch	.83	.69	.42	.83	.56	.42	.83	.47	.42
$BaseLine_1$	**.8**	**.68**	**.43**	**.8**	**.56**	**.43**	**.8**	**.47**	**.43**
CSA	.61	.58	.47	.5	.55	.6	.5	.58	.6
CIDER	.67	.61	.44	.64	.53	.45	.38	.48	.51
MapSSS	.55	.53	.47	.55	.51	.47	.55	.48	.47
Lily	.48	.42	.27	.36	.41	.47	.37	.45	.47
AROMA	.35	.37	.46	.35	.4	.46	.35	.43	.46
Optima	.25	.28	.57	.25	.35	.57	.25	.45	.57
MapPSO	.28	.25	.17	.21	.23	.25	.12	.26	.36
LDOA	.1	.12	.56	.1	.17	.56	.1	.29	.56
MapEVO	.27	.08	.02	.15	.04	.02	.02	.02	.02

Conclusion and Future work

Element matcher

- Combining terminological similarity metrics using ML (decision tree)

Structure matcher

- Similarity propagation method
- Using element matcher's output as input

Combination module

- Dynamic weighted sum
- Combining element and structure matcher results

Conclusion and Future work

Issues

- Dependency on gold standard dataset for classification in the element matcher
  - Gold standard dataset not always available
  - Gold standard dataset enough?!!

- High complexity in memory consuming
  - Graph-based matching method in the structure matcher
  - Large scale ontologies

Solutions

- Creating a new gold standard data set from another resource
- Partitioning large scale ontologies into sub-ontologies

Questions?

Thank you for your attention!