UNIVERSITÀ DEGLI STUDI DI TRENTO

Ontology Matching and Alignment

Ilya Zaihrayeu

Slides adopted from a presentation of Pavel Shvaiko (http://www.dit.unitn.it/~pavel)

The 1st Asian Autumn School on the Semantic Web

6 November 2007, Busan, Korea

EAST WEB

Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

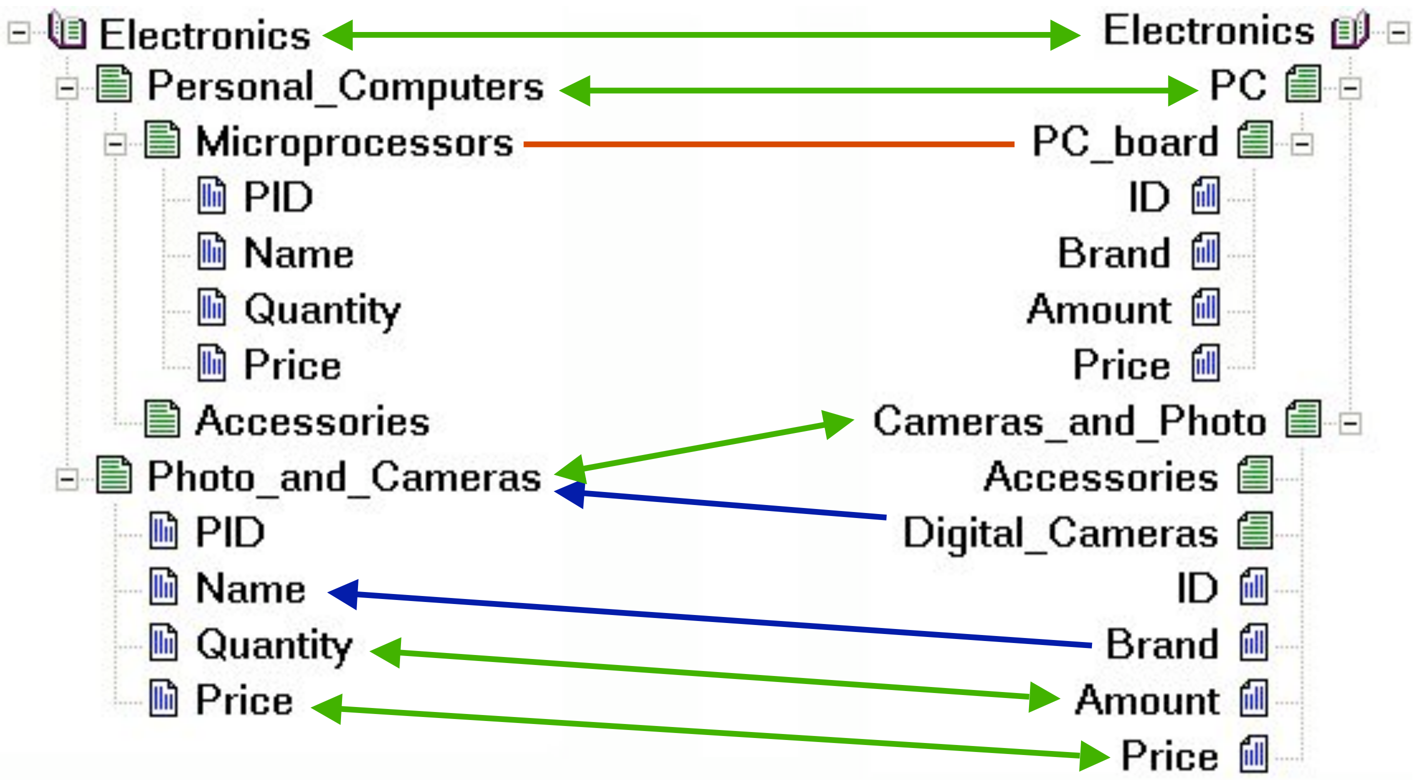
Outline

- Part I: The matching problem
  - Problem statement
  - Applications
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Matching operation

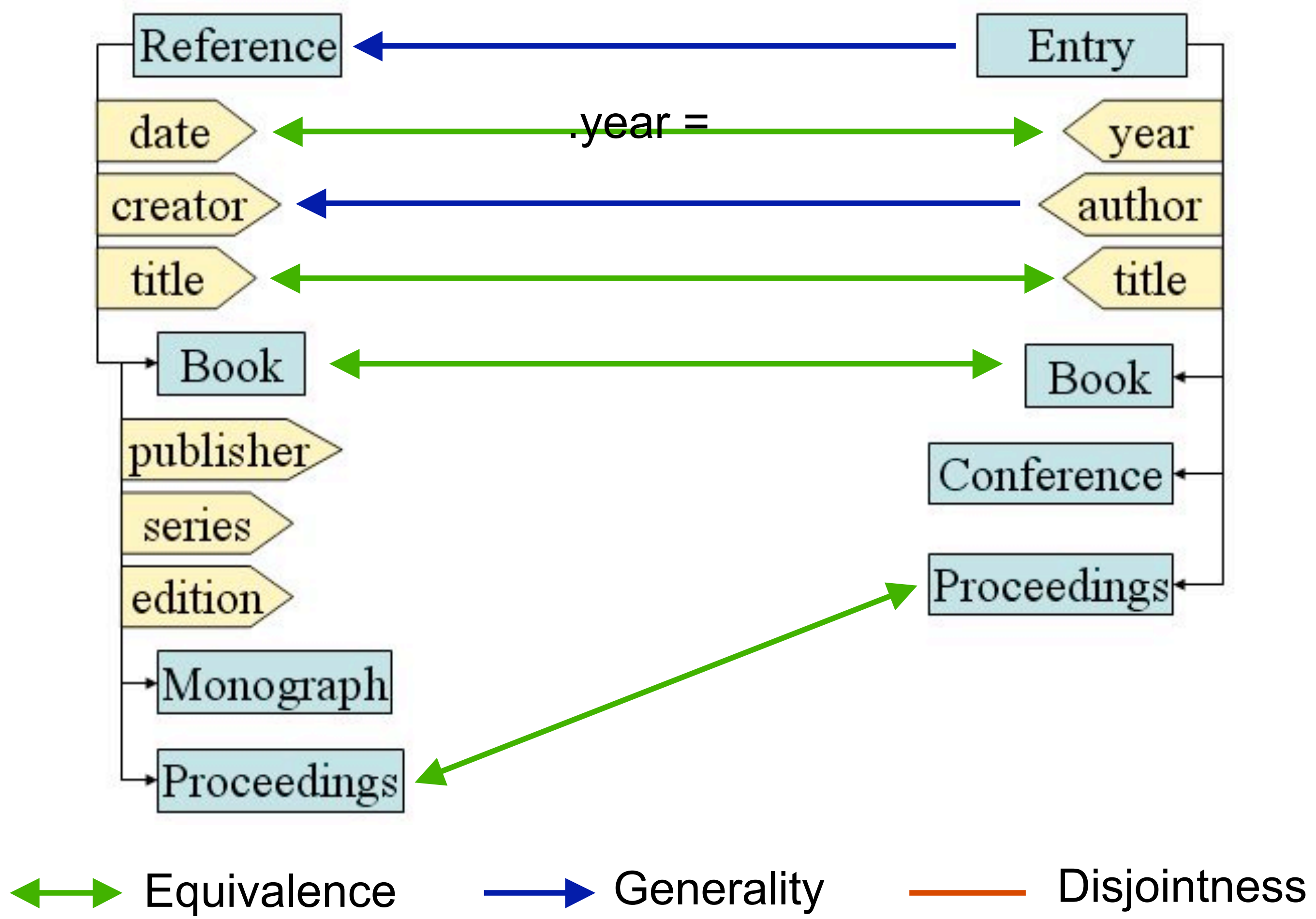
Matching operation takes as input ontologies, each consisting of a set of discrete entities (e.g., tables, XML elements, classes, properties) and determines as output the correspondences (e.g., equivalence, subsumption) that hold between these entities

Example: two XML schemas



Equivalence Generality Disjointness

Example: two ontologies



Statement of the problem

Scope

- Reducing heterogeneity can be performed in two steps:
  - Match, thereby determine the alignment
  - Process the alignment (merge, transform, translate...)

Statement of the problem

Correspondence is a 5-tuple ***<id, e1, e2, R, n>***

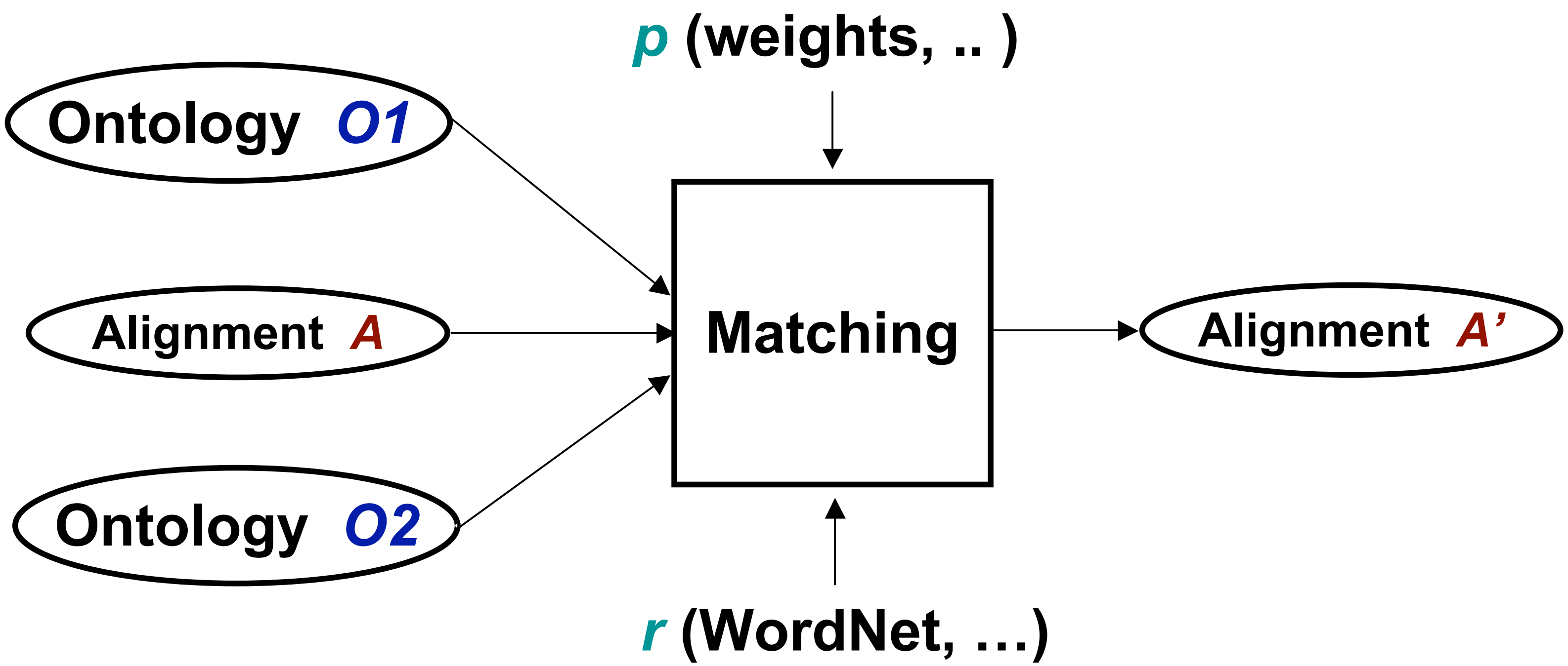
- ***id*** is a unique identifier of the correspondence
- ***e1*** and ***e2*** are entities (XML elements, classes,...)
- ***R*** is a relation (equivalence, more general, disjointness,...)
- ***n*** is a confidence measure, typically in the [0,1] range

Alignment (A) is a set of correspondences

- with some cardinality: 1-1, 1-n, ...
- some other properties (complete/partial)

Statement of the problem

Matching process



Applications

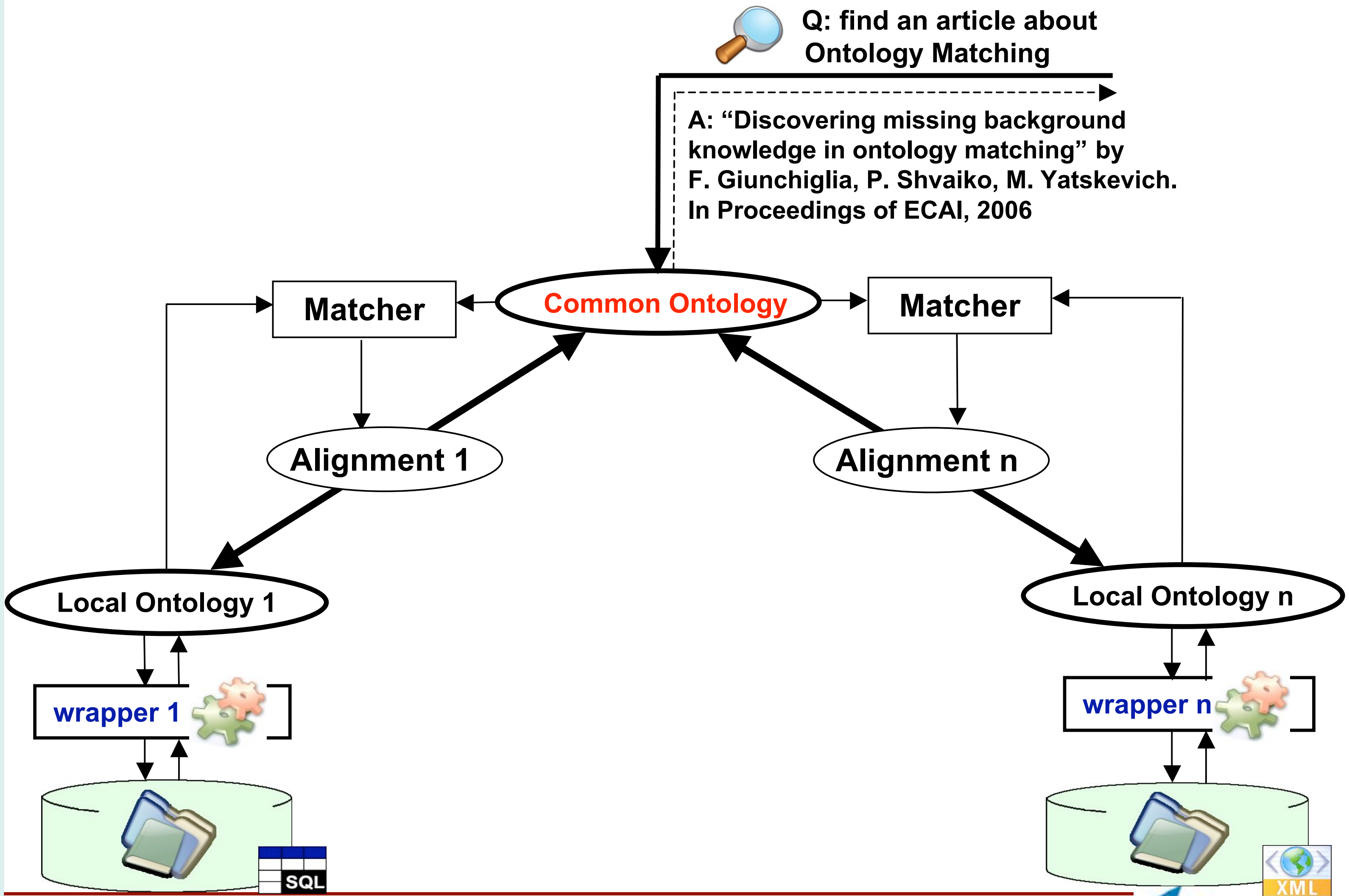
Traditional

- Ontology evolution
- Schema integration
- Catalog integration
- Data integration

Emergent

- P2P information sharing
- Web service composition
- Agent communication
- Query answering on the web

Applications: Information integration



Applications: summary

Application	instances	run time	automatic	correct	complete	operation
Ontology evolution	√			√	√	transformation
Schema integration	√			√	√	merging
Catalog integration	√			√	√	data translation
Data integration	√			√	√	query answering
P2P information sharing		√				query answering
Web service composition		√	√	√		data mediation
Multi-agent communication		√	√	√	√	data translation
Query answering	√	√		√		query reformulation

Outline

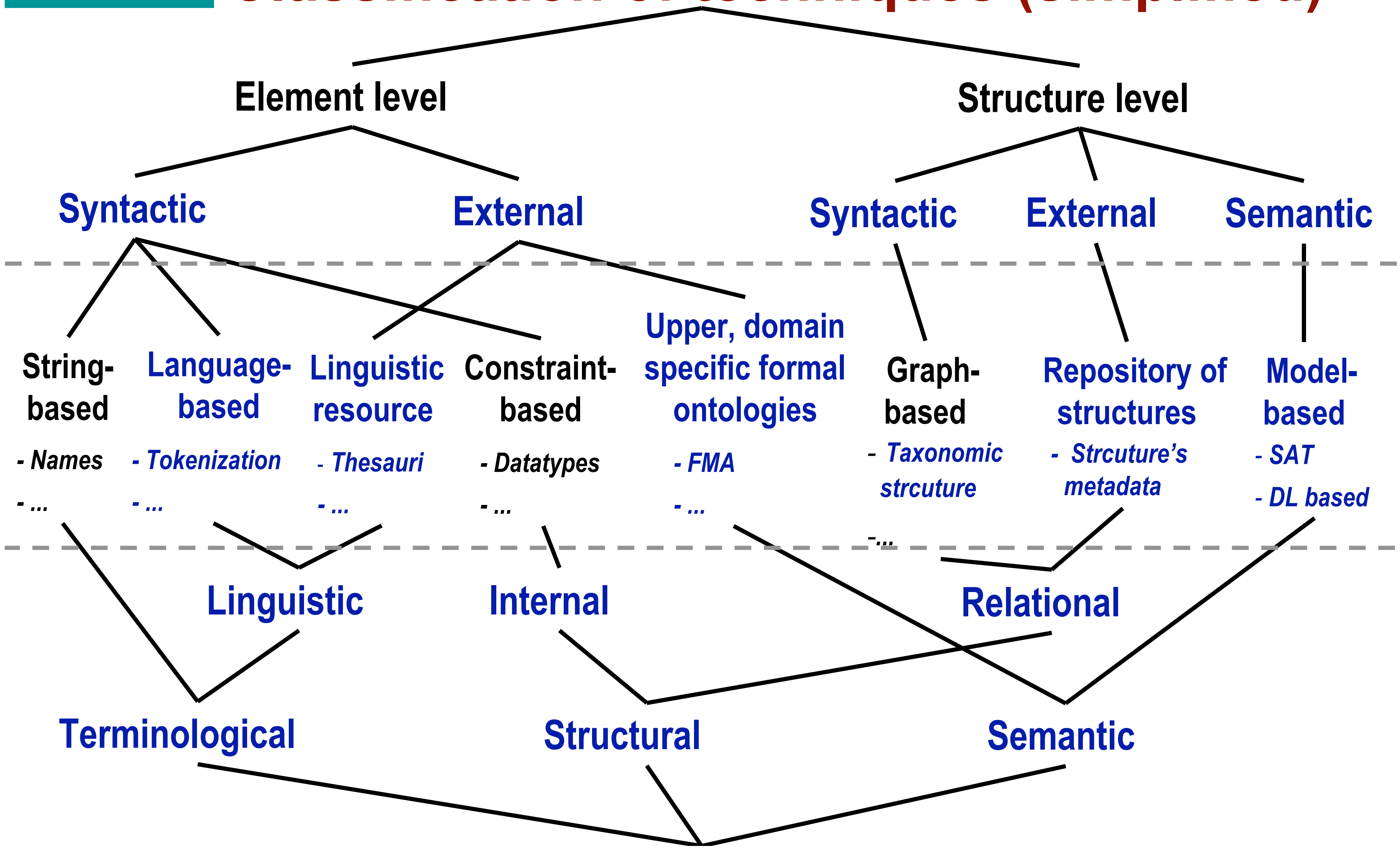
- Part I: The matching problem
- Part II: State of the art in ontology matching
  - Classification of matching techniques
  - Overview of matching systems
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Classification of basic techniques

Three layers

- **The upper layer**
  - Granularity of matching
  - Interpretation of input information
- **The middle layer** represents classes of elementary (basic) matching techniques
- **The lower layer** is based on the kind of input which is used by elementary matching techniques

Classification of techniques (simplified)



Basic techniques

String-based

- Edit distance
  - It takes as input two strings and calculates the number of *insertions*, *deletions*, and *substitutions* of characters required to transform one string into another, normalized by *max(length(string1), length(string2))*
  - EditDistance(NKN,Nikon) = 0.4

Basic techniques (cont'd)

Linguistic resources: WordNet

It computes relations between ontology entities by using (lexical) relationships of WordNet

- $A \subseteq B$ if A is a **hyponym** or **meronym** of B
  - Brand $\subseteq$ Name
- $A \supseteq B$ if A is a **hypernym** or **holonym** of B
  - Europe $\supseteq$ Greece
- A = B if they are **synonyms**
  - Quantity = Amount
- $A \perp B$ if they are **antonyms** or **siblings in part of hierarchy**
  - Microprocessors $\perp$ PC Board

Systems: analytical comparison

~50 matching systems exist, …we consider some of them

		SF	Artemis	Cupid	COMA	Prompt	OLA	S-Match
Element-level	Syntactic	string-based, data types, key properties	domain compatibility, language-based	string-based, language-based, data types, key properties	string-based language-based, data types	string-based, domains and ranges	string-based, data types, language-based	string-based, language-based
Element-level	External	-	common thesaurus (CT)	auxiliary dictionary	auxiliary dictionary	-	WordNet	WordNet
Structure-level	Syntactic	iterative fix-point computation	matching of neighbors via CT	tree matching weighted by leaves	DAG (tree) matching with a bias towards leaf or children structures	bounded path matching (arbitrary links, *is-a* links)	iterative fix-point computation, matching of neighbors	-
Structure-level	Semantic	-	-	-	-	-	-	SAT

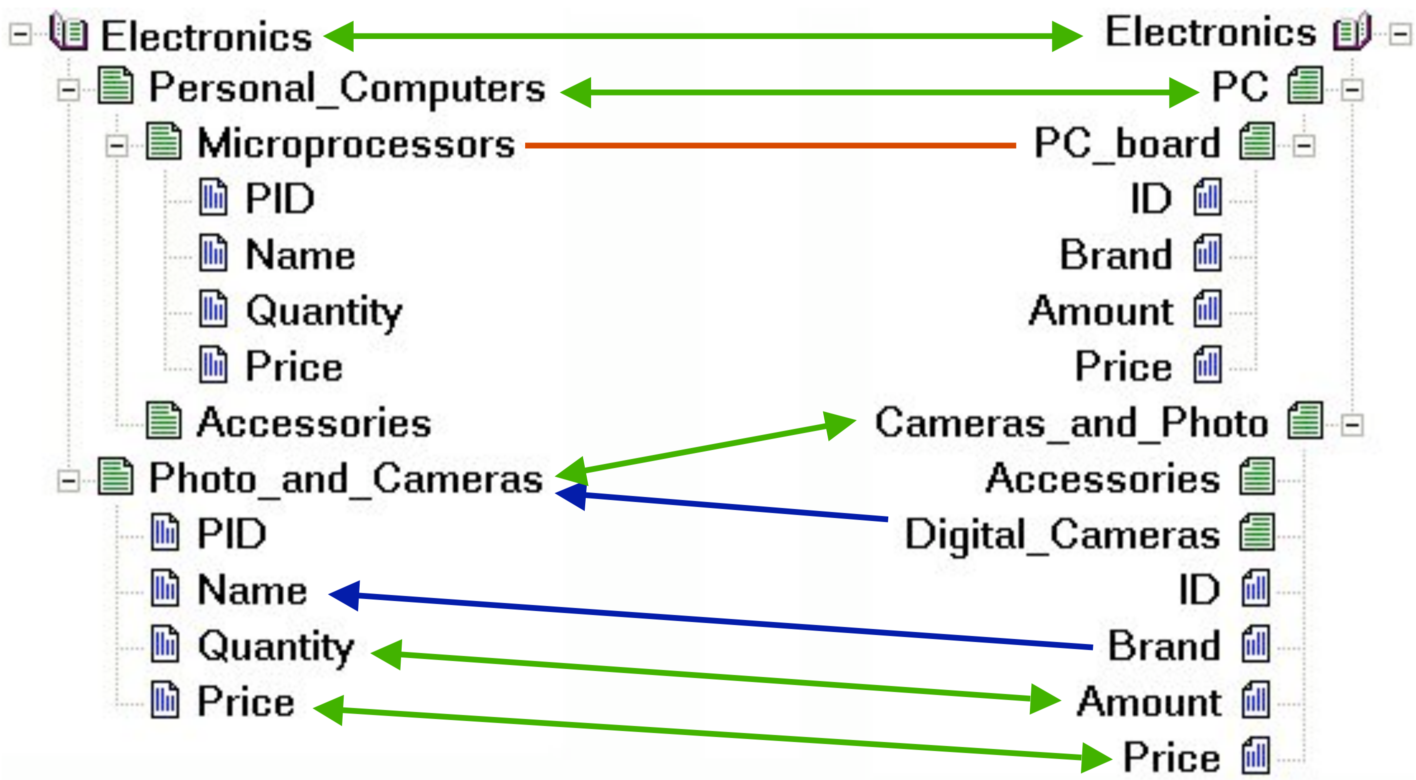
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
  - Semantic matching
  - Iterative semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Generic matching

Information sources (classifications, XML schemas, …) can be viewed as **graph-like** structures containing terms and their inter-relationships

Matching takes two graph-like structures and produces **correspondences** between the nodes of the graphs that are supposed to correspond to each other



Semantic matching in a nutshell

Semantic matching: Given two graphs *G1* and *G2*, for any node $n1_i \in G1$, find the strongest semantic relation *R'* holding with node $n2_j \in G2$

Computed *R's*, listed in the decreasing binding strength order:

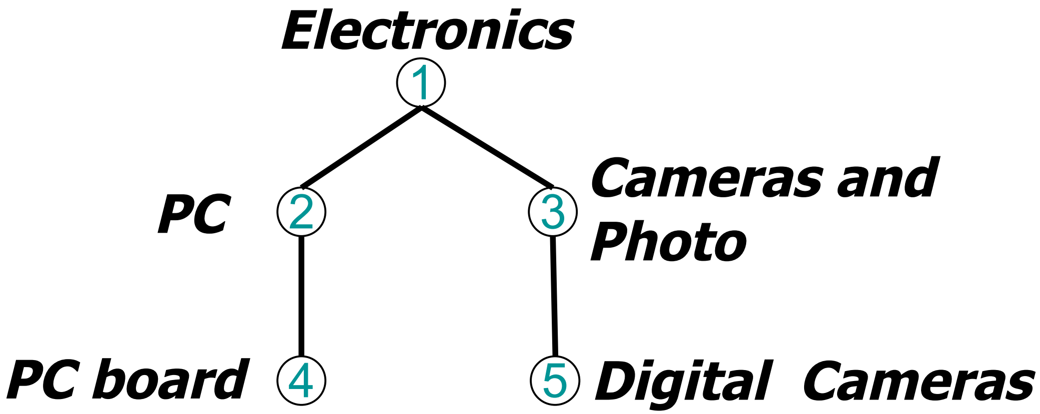
- equivalence { = }
- more general/specific { $\sqsupseteq$, $\sqsubseteq$ }
- disjointness { $\perp$ }

In case no relation is found, 'I don't know' {idk} is returned

We compute semantic relations by analyzing the ***meaning*** *(concepts, not labels)* which is codified in the elements and the structures of ontologies

Technically, labels at nodes written in natural language are translated into propositional DL formulas which codify labels' intended meaning. This allows us to codify the matching problem into a propositional validity problem

Concept of a label & concept at a node



Concept of a label is a propositional DL formula which encodes the set of documents, one would classify under this **label**

Concept at a node is a propositional DL formula which encodes the set of documents, one would classify under this node, given its **label** and its **position** in the tree

Four macro steps

Given two labeled trees T1 and T2, do:

1. **For all labels in T1 and T2 compute *concepts at labels***
2. **For all nodes in T1 and T2 compute *concepts at nodes***
3. **For all pairs of labels in T1 and T2 compute relations between concepts at labels (background knowledge)**
4. **For all pairs of nodes in T1 and T2 compute relations between concepts at nodes**

Steps 1 and 2 constitute the preprocessing phase, and are executed once and each time after the ontology is changed (OFF- LINE part)

Steps 3 and 4 constitute the matching phase, and are executed every time two ontologies need to be matched (ON - LINE part)

Step 1: compute concepts at labels

The idea

- Translate labels at nodes written in natural language into propositional DL formulas which codify labels’ intended meaning

Preprocessing

- **Tokenization.** Labels (according to punctuation, spaces, etc.) are parsed into tokens. E.g., Photo and Cameras → <Photo, and, Cameras>
- **Lemmatization.** Tokens are morphologically analyzed in order to find all their possible basic forms. E.g., Cameras → Camera
- **More NLP.** Named entity locating, word sense disambiguation, and syntactic parsing are required for a more accurate translation
- **Building atomic concepts.** An oracle (WordNet) is used to extract senses of lemmas. E.g., Camera has 2 senses
- **Building complex concepts.** Prepositions, conjunctions are translated into logical connectives and used to build complex concepts out of the atomic concepts

E.g., $C_{Cameras\_and\_Photo} = \langle Cameras, \{WN_{Camera}\} \rangle \sqcup \langle Photo, \{WN_{Photo}\} \rangle$

Step 2: compute concepts at nodes

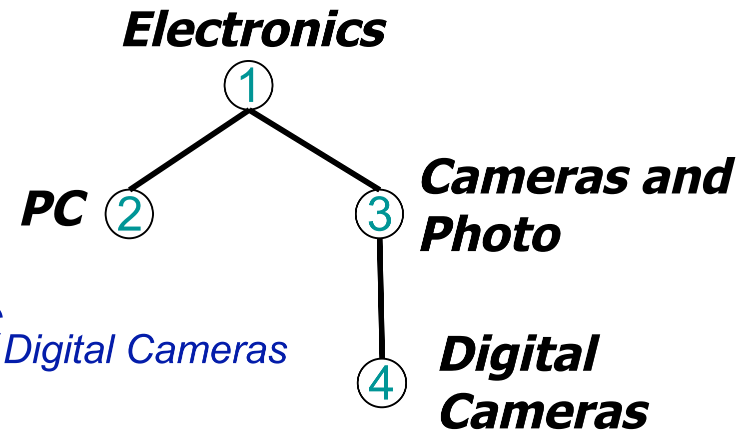
The idea

Extend **concepts at labels** by capturing the knowledge encoded in the structure of the ontology tree in order to define the context in which the given **concept at a label** occurs

Computation

Concept at a node for some node *n* is computed as the conjunction of **concepts at labels** located above the given node, including the node itself

Example:

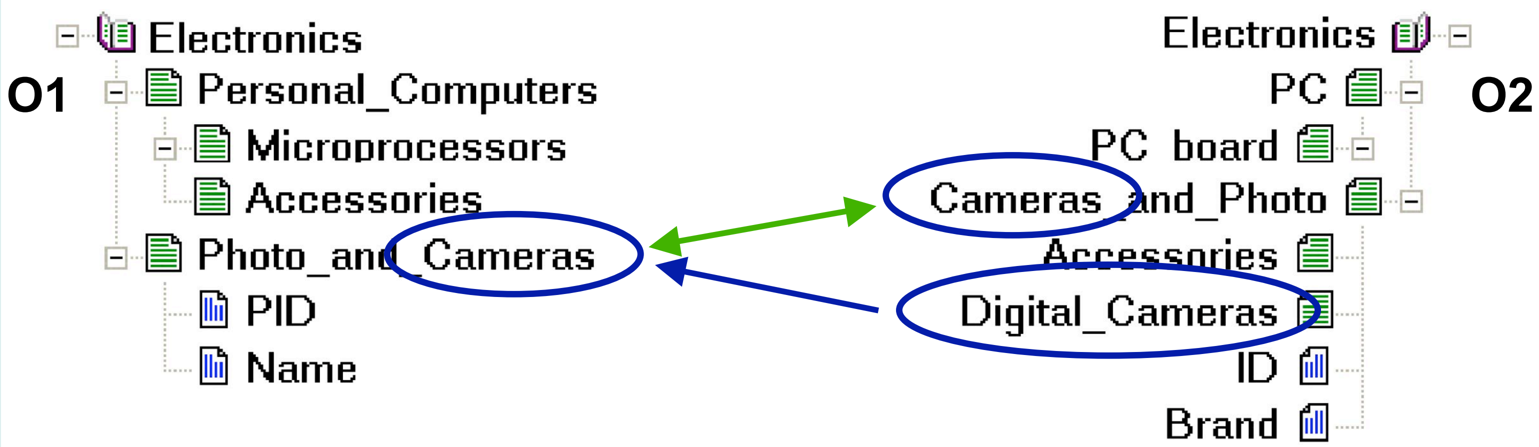


$$C_4 = C_{Electronics} \sqcap (C_{Cameras} \sqcup C_{Photo}) \sqcap C_{Digital\ Cameras}$$

Step 3: compute relations between (atomic) concepts at labels

The idea

- Exploit a priori knowledge, e.g., lexical, domain knowledge, with help of element level semantic matchers



cLabsMatrix (result of Step 3)

	Cameras_2	Photo_2	Digital_Cameras_2
Photo_1	*idk*	=	*idk*
Cameras_1	=	*idk*	$\sqsupseteq$

Step 3:
Element level semantic matchers

Sense-based matchers have two WordNet senses in input and produce semantic relations exploiting (direct) lexical relations of WordNet

String-based matchers have two labels in input and produce semantic relations exploiting string comparison techniques

Matcher name	Execution order	Approximation level	Matcher type	Schema info
WordNet	1	1	Sense-based	WordNet senses
Prefix	2	2	String-based	Labels
Suffix	3	2	String-based	Labels
Edit distance	4	2	String-based	Labels
Ngram	5	2	String-based	Labels

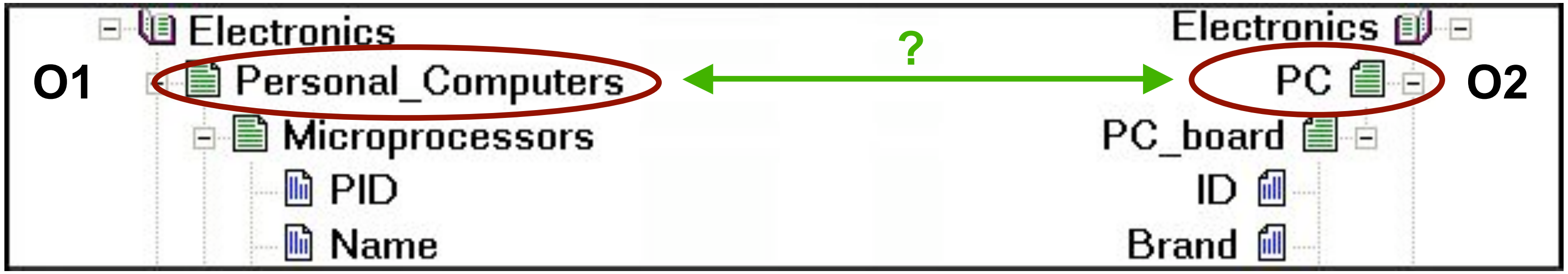
Step 4: compute relations between concepts at nodes

The idea

- Decompose the tree matching problem into the set of **node matching** problems
- Translate each node matching problem, namely pairs of nodes with possible relations between them, into a propositional formula
- Check the propositional formula for validity

Step 4:
Example of a node matching task

$$\textit{Axioms} \rightarrow rel(context_1, context_2)$$



Axioms

$$(Electronics_1 \leftrightarrow Electronics_2) \wedge (Personal\_Computers_1 \leftrightarrow PC_2) \rightarrow$$

$$(Electronics_1 \wedge Personal\_Computers_1) \leftrightarrow (Electronics_2 \wedge PC_2)$$

$context_1$ *$context_2$*

Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
  - Semantic matching
  - Iterative semantic matching
- Part IV: Evaluation (technology showcase)
- Part V: Conclusions

Motivation:
Problem of low recall (incompleteness) - I

Facts

- Matching (usually) has two components: element level matching and structure level matching
- Contrarily to many other systems, the semantic matching structure level algorithm is correct and complete
- Still, the quality of results is not very good

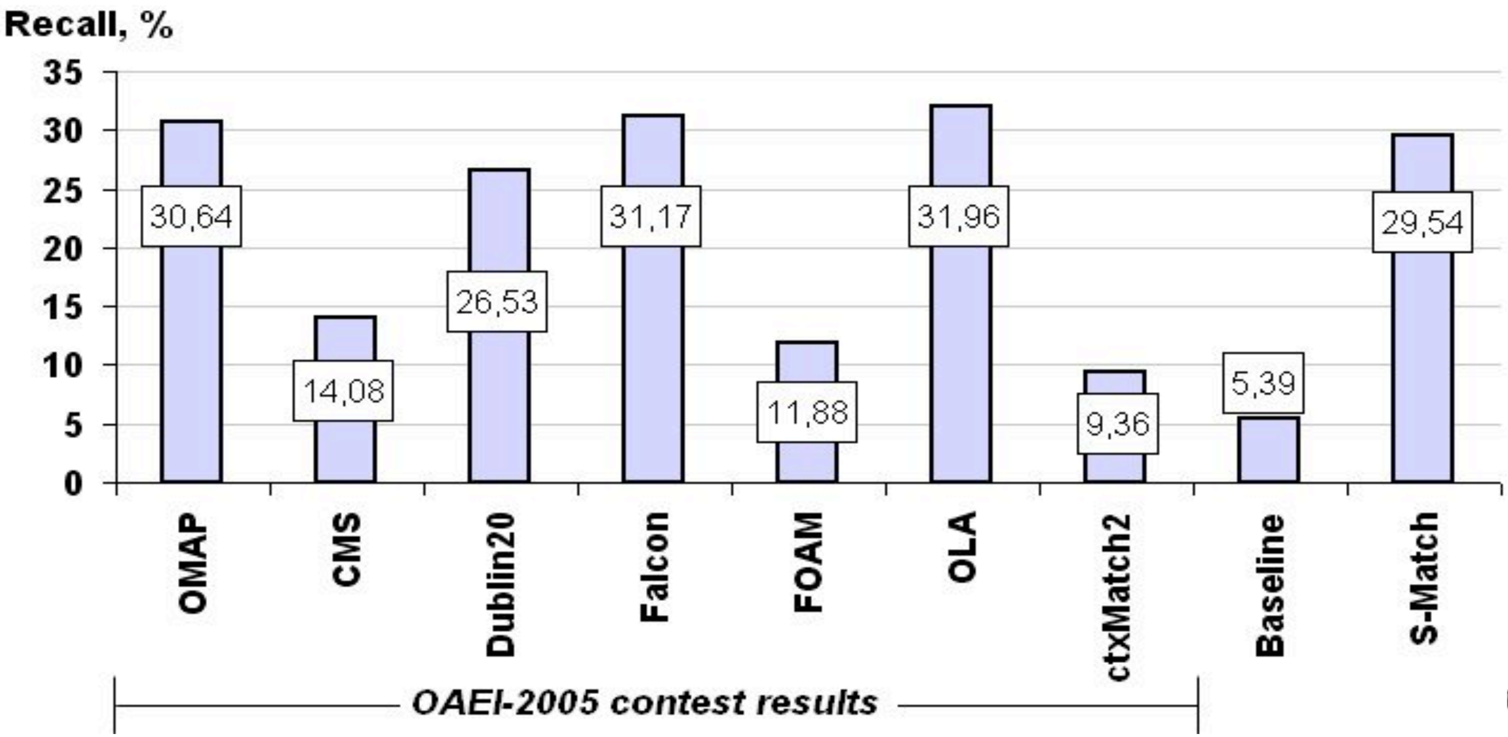
Why? ... the problem of lack of knowledge

Motivation:
Problem of low recall (incompleteness) - II

Preliminary (analytical) evaluation

Matching tasks	#nodes	max depth	#labels per tree
Google vs Looksmart	706/1081	11/16	1048/1715
Google vs Yahoo	561/665	11/11	722/945
Yahoo vs Looksmart	74/140	8/10	101/222

Dataset
[P. Avesani et al., ISWC'05]



On increasing the recall: an overview

Multiple strategies

- Add new element level matchers
- Reuse of previous match results from the same domain of interest
  - PO = Purchase Order
- Use general knowledge sources (unlikely to help)
  - WWW
- Use, if available (!), domain specific sources of knowledge
  - UMLS, FMA

Iterative semantic matching (ISM)

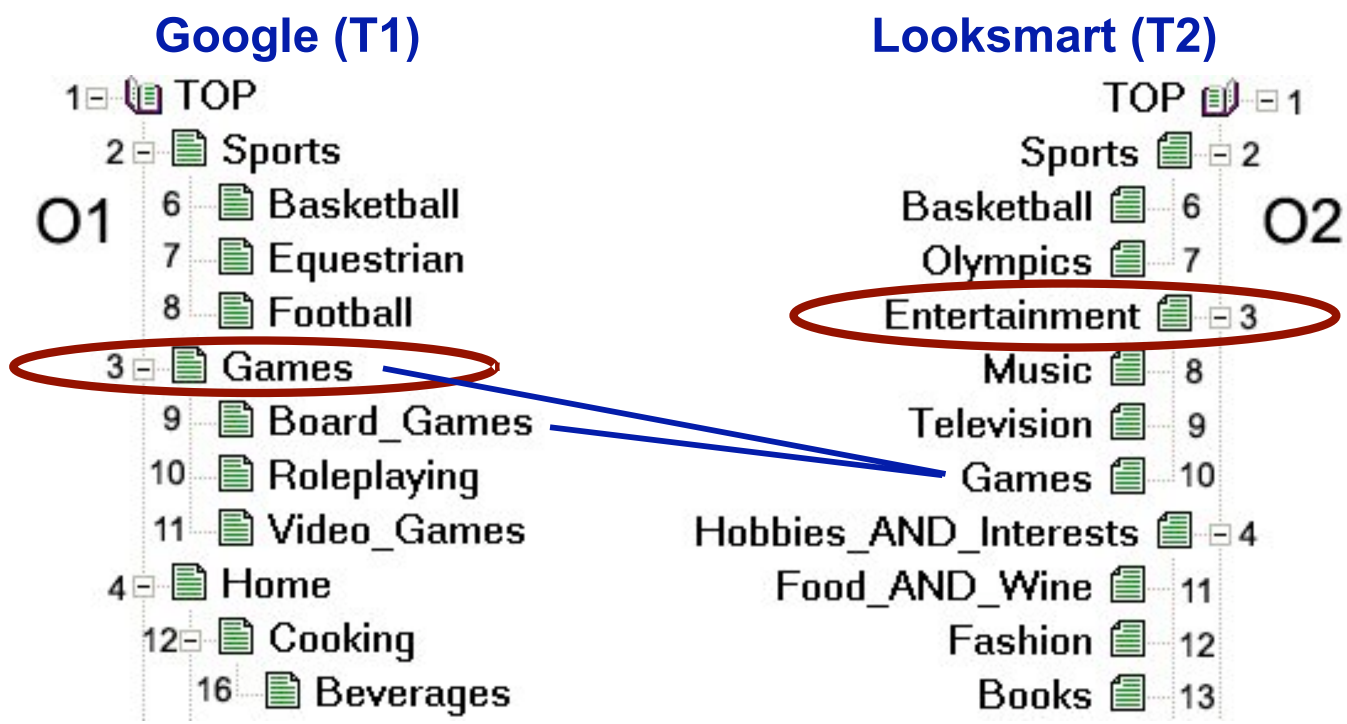
The idea

Repeat ***Step 3*** and ***Step 4*** of the matching algorithm for some **critical** (hard) matching tasks

ISM macro steps

- Discover ***critical points*** in the matching process
- Generate candidate ***missing axiom(s)***
- Re-run SAT solver on a critical task taking into account the new axiom(s)
- If SAT returns **false**, save the newly discovered axiom(s) for future reuse

ISM:
Discovering critical points - example



cLabsMatrix (result of Step 3)

	TOP_1	$Games_1$	$Board\_Games_1$
TOP_2	=	*idk*	*idk*
$Entertainment_2$	*idk*	*idk*	*idk*
$Games_2$	*idk*	=	$\sqsupseteq$

cNodesMatrix (result of Step 4)

	$C1_1$	$C1_2$	$C1_3$	$C1_4$	$C1_9$	$C1_{10}$	$C1_{11}$
$C2_1$	=	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$	$\sqsupseteq$
$C2_3$	$\sqsubseteq$	*idk*	*idk*	*idk*	*idk*	*idk*	*idk*

ISM:
Generating candidate axioms

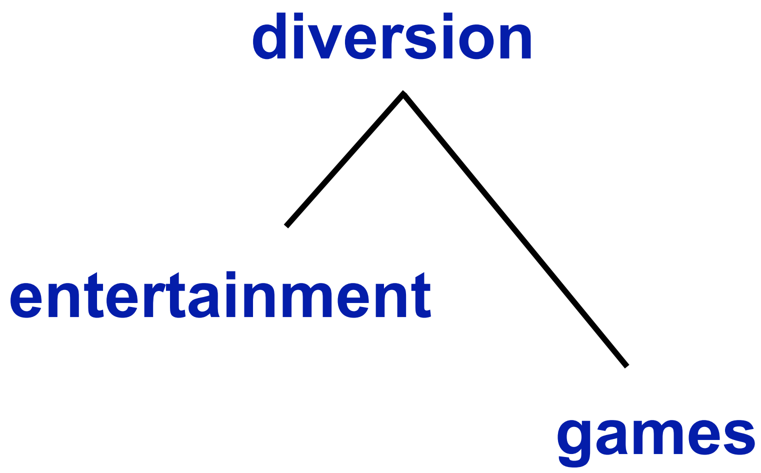
- ***Sense-based* matchers** have two WordNet senses in input and produce semantic relations exploiting structural properties of WordNet hierarchies
  - Hierarchy Distance (HD)
- ***Gloss-based* matchers** have two WordNet senses as input and produce relations exploiting gloss comparison techniques
  - WordNet Gloss (WNG)
  - Extended WordNet Gloss (EWNG)
  - Gloss Comparison (GC)

ISM: generating candidate axioms
Hierarchy Distance

Hierarchy distance returns the equivalence relation if the distance between two input senses in WordNet hierarchy is less than a given threshold value (e.g., 3) and *idk* otherwise

There is no direct relation between *games* and *entertainment* in WordNet

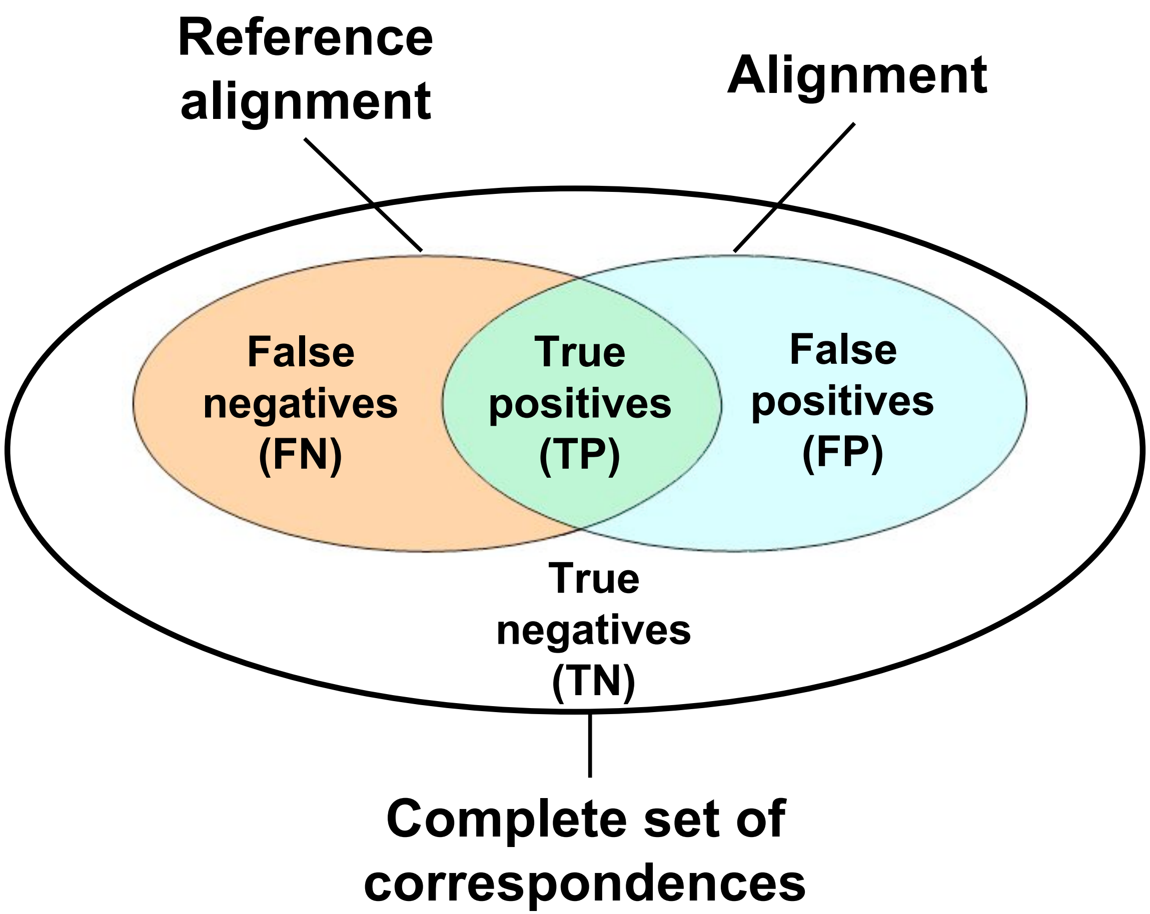
Distance between these concepts is 2 (1 more general link and 1 less general). Thus, we can conclude that *games* and *entertainment* are close in their meaning and return the equivalence relation



Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
  - Evaluation setup
  - Evaluation results
- Part V: Conclusions

Evaluation (quality) measures



Test cases

#	Matching task	#nodes	max depth	#labels per tree
1	Images vs Europe	4/5	2/2	6/5
2	Product schemas	13/14	4/4	14/15
3	Yahoo Finance vs Standard	10/16	2/2	22/45
4	Cornell vs Washington	34/39	3/3	62/64
5	CIDX vs Excel	34/39	3/3	56/58
6	Google vs Looksmart	706/1081	11/16	1048/1715
7	Google vs Yahoo	561/665	11/11	722/945
8	Yahoo vs Looksmart	74/140	8/10	101/222
9	Iconclass vs Aria	999/553	9/3	2688/835

Matching systems

Schema-based systems

- S-Match
- Cupid
- COMA
- Similarity Flooding as implemented in Rondo
- OAEI-2005 and OAEI-2006 participants

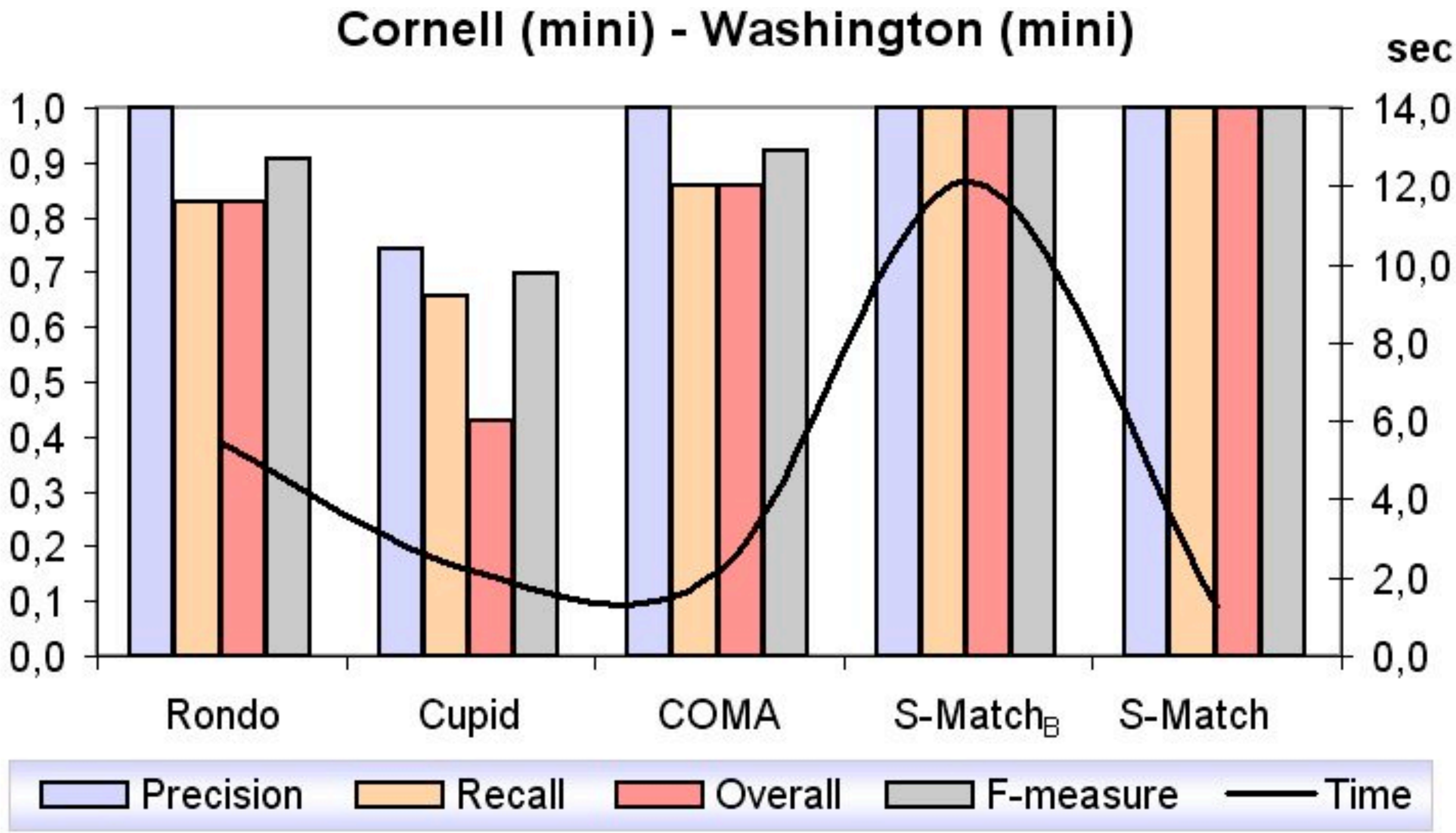
Systems were used in default configurations

PC: PIV 1,7Ghz; 512Mb. RAM; Win XP

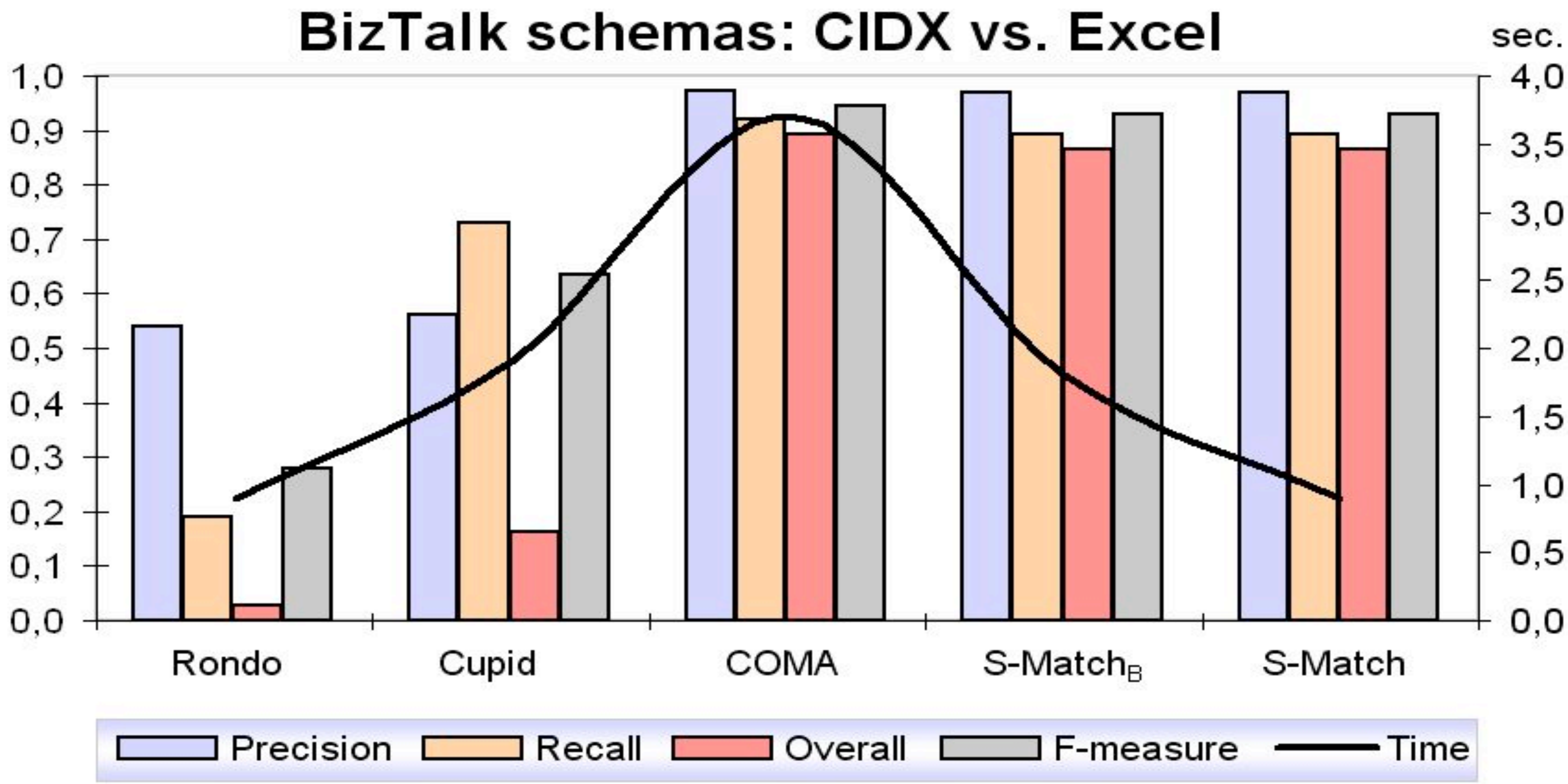
Outline

- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- **Part IV: Evaluation (technology showcase)**
  - Evaluation setup
  - **Evaluation results**
- Part V: Conclusions

Experimental results, test case #4

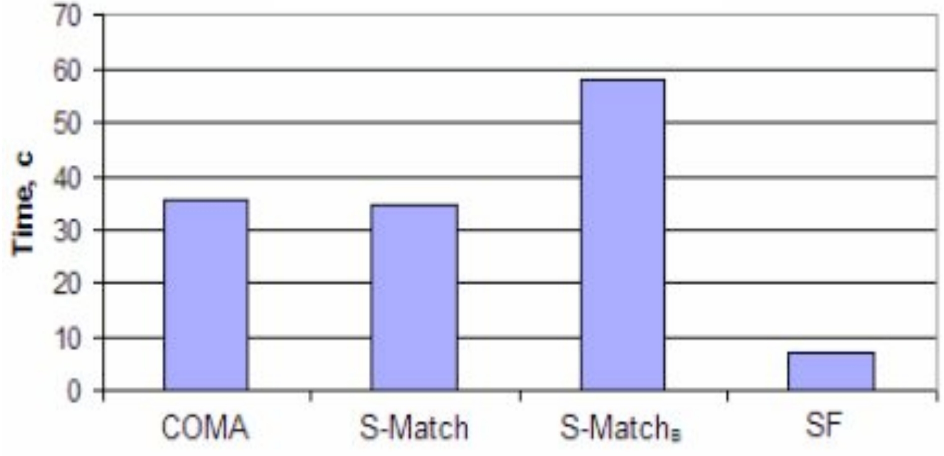


Experimental results, test case #5

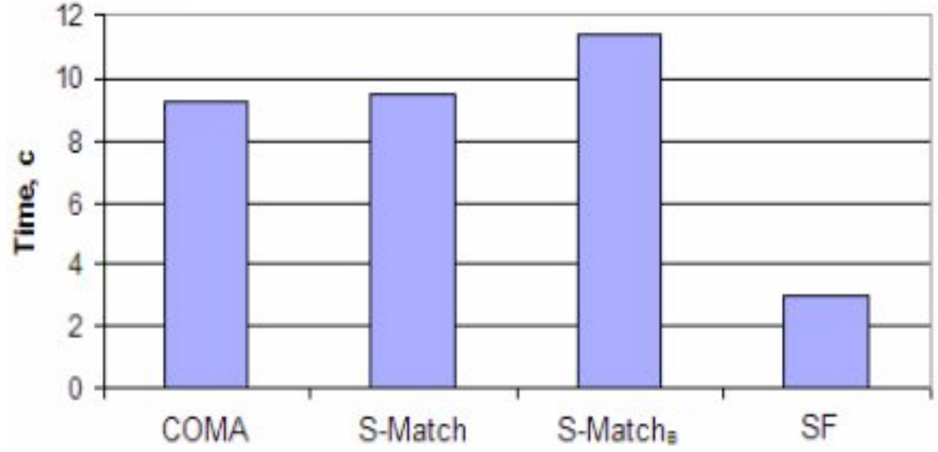


Experimental results, #3,6,7,8: efficiency

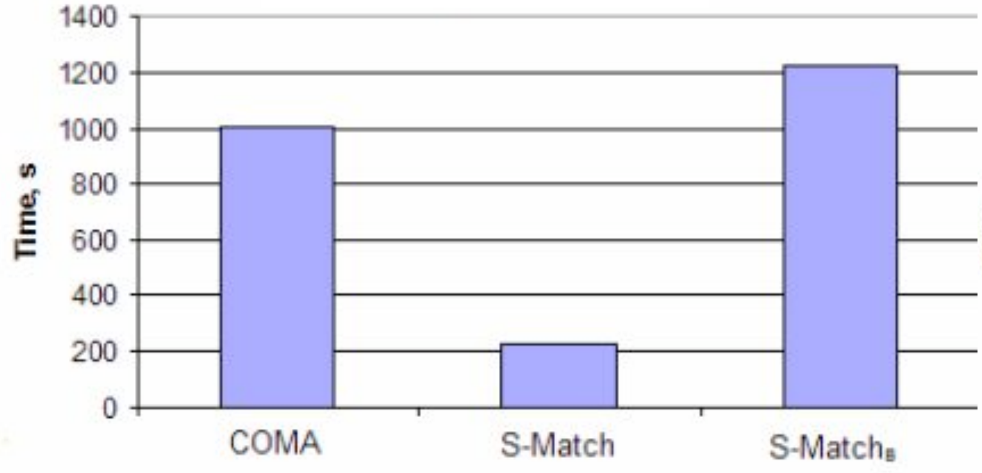
Yahoo-Standard



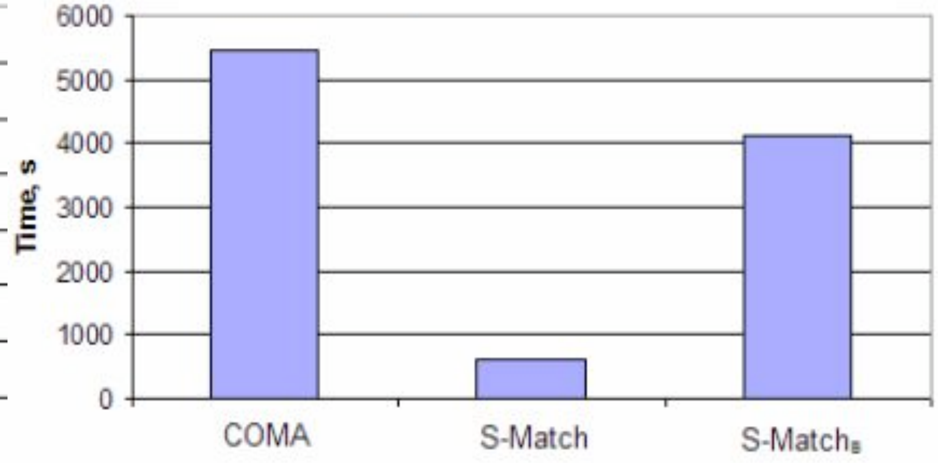
Looksmart -Yahoo



Google-Yahoo

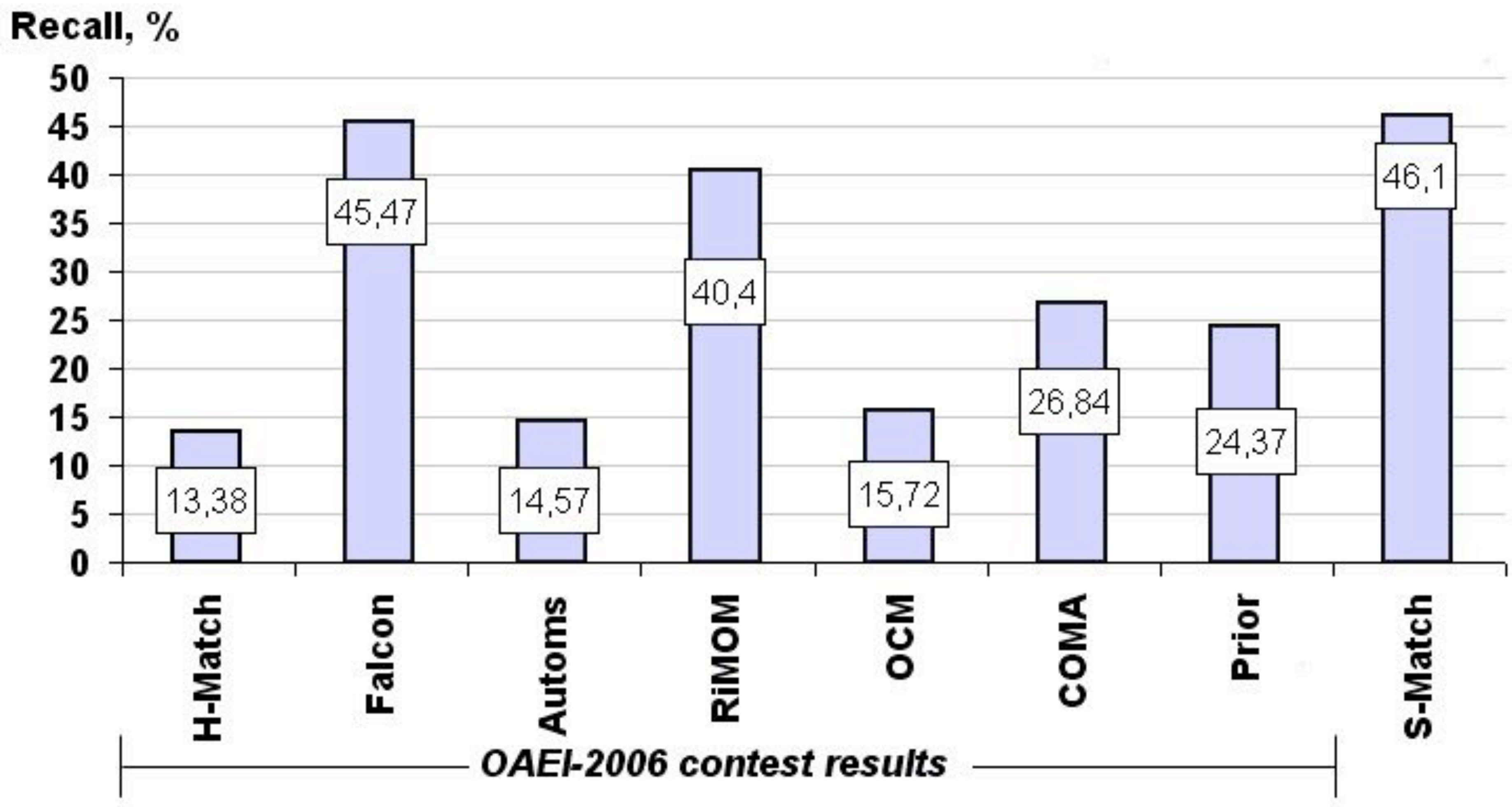


Google-Looksmart



Experimental results, #6,7,8: incompleteness

Experimental results, #6,7,8: incompleteness (OAEI-2006 comparison)



Outline

- Thesis contributions
- Part I: The matching problem
- Part II: State of the art in ontology matching
- Part III: Schema-based semantic matching
- Part IV: Evaluation (technology showcase)
- **Part V: Conclusions**

Summary

- Ontology matching applications and their requirements
- Overview of the state of the art, including classification of matching techniques and systems
- Semantic matching approach, including algorithms for basic and iterative semantic matching
- Evaluation of the approach on various data sets with encouraging results

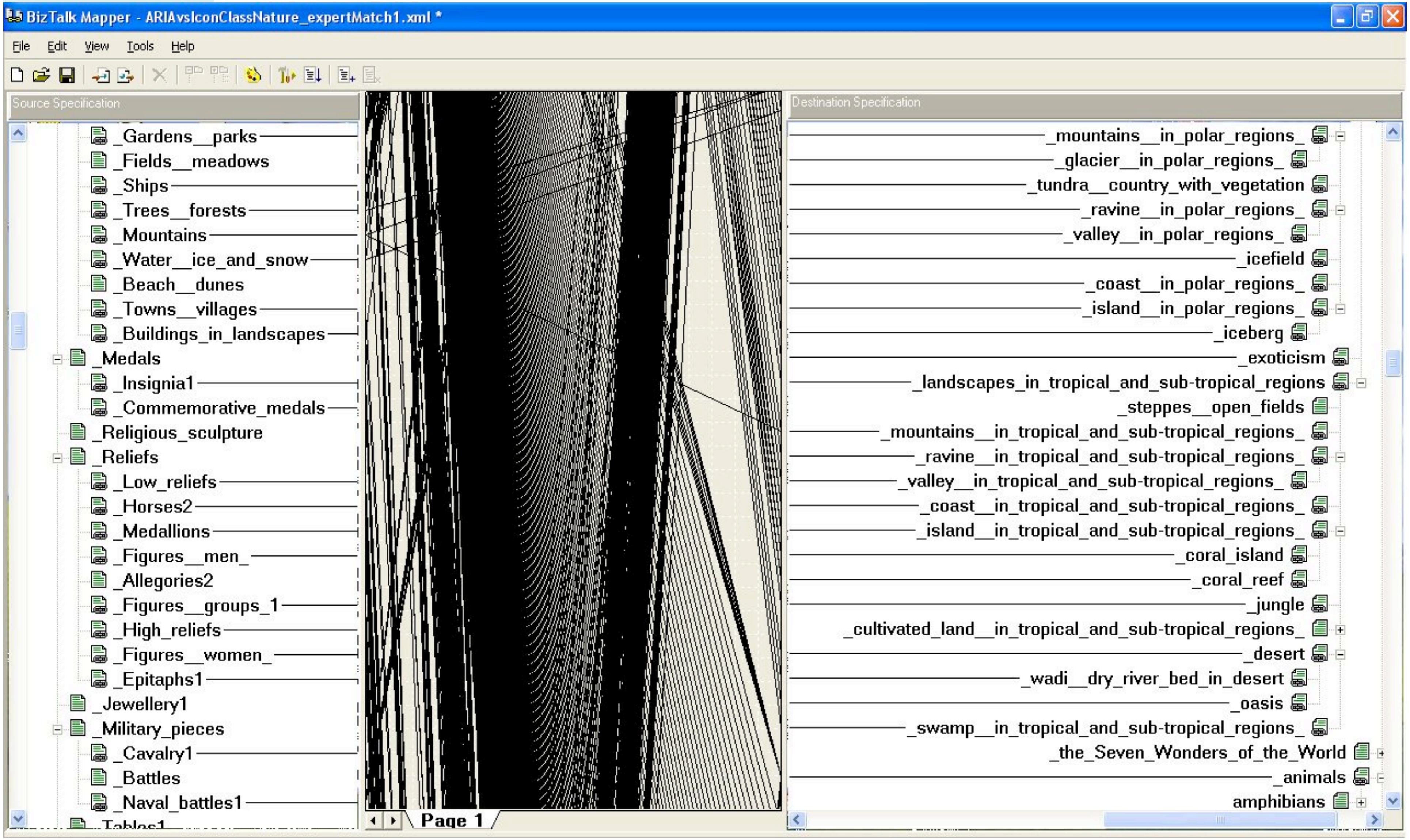
Summary (cont'd)

- Automated reasoning techniques (e.g., SAT) provide good performance for industrial-strength matching tasks
- The issue is not efficiency but rather missing **domain knowledge**
  - This problem on the industrial size matching tasks is very hard
  - We have investigated it by examples of lightweight ontologies, such as Google and Yahoo
  - A partial solution is applying semantic matching iteratively

Future challenges

- Missing background knowledge
- Natural language processing
- Interactive approaches
- Explanations of matching results
- Social and collaborative ontology matching
- Large-scale evaluation
- Infrastructures
- ...

Future challenges: scalability of visualization



(Some) references

- Project website - KNOWDIVE: http://www.dit.unitn.it/~knowdive/
- Ontology Matching website: http://www.OntologyMatching.org
- F. Giunchiglia, M. Yatskevich, P. Shvaiko: Semantic matching: algorithms and implementation. Journal on Data Semantics, IX, 2007.
- F. Giunchiglia, P. Shvaiko, M. Yatskevich: Discovering missing background knowledge in ontology matching. In Proceedings of *ECAI'06.*
- I. Zaihrayeu, L. Sun, F. Giunchiglia, W. Pan, Q. Ju, M. Chi, and X. Huang: From Web Directories To Ontologies: Natural Language Processing Challenges. In Proceedings of ISWC'07
- P. Shvaiko and J. Euzenat: A survey of schema-based matching approaches. Journal on Data Semantics, IV, 2005.
- P. Shvaiko, J. Euzenat, N. Noy, H. Stuckenschmidt, R. Benjamins, M. Uschold. Proceedings of the ISWC International Workshop on Ontology Matching, 2006.
- P. Avesani, F. Giunchiglia, M. Yatskevich: A large scale taxonomy mapping evaluation. In Proceedings of *ISWC'05*.
- B. Magnini, M. Speranza, C. Girardi. A semantic-based approach to interoperability of classification hierarchies: Evaluation of linguistic techniques. In Proceedings of *COLING'04*.
- P. Bouquet, L. Serafini, S. Zanobini: Semantic coordination: a new approach and an application. In Proceedings of *ISWC'03*.
- C. Ghidini, F. Giunchiglia: Local models semantics, or contextual reasoning = locality + compatibility. Artificial Intelligence Journal, 127(3), 2001.

You are welcome to attend (11 Nov):

Ontology Matching @ ISWC'07+ASWC'07

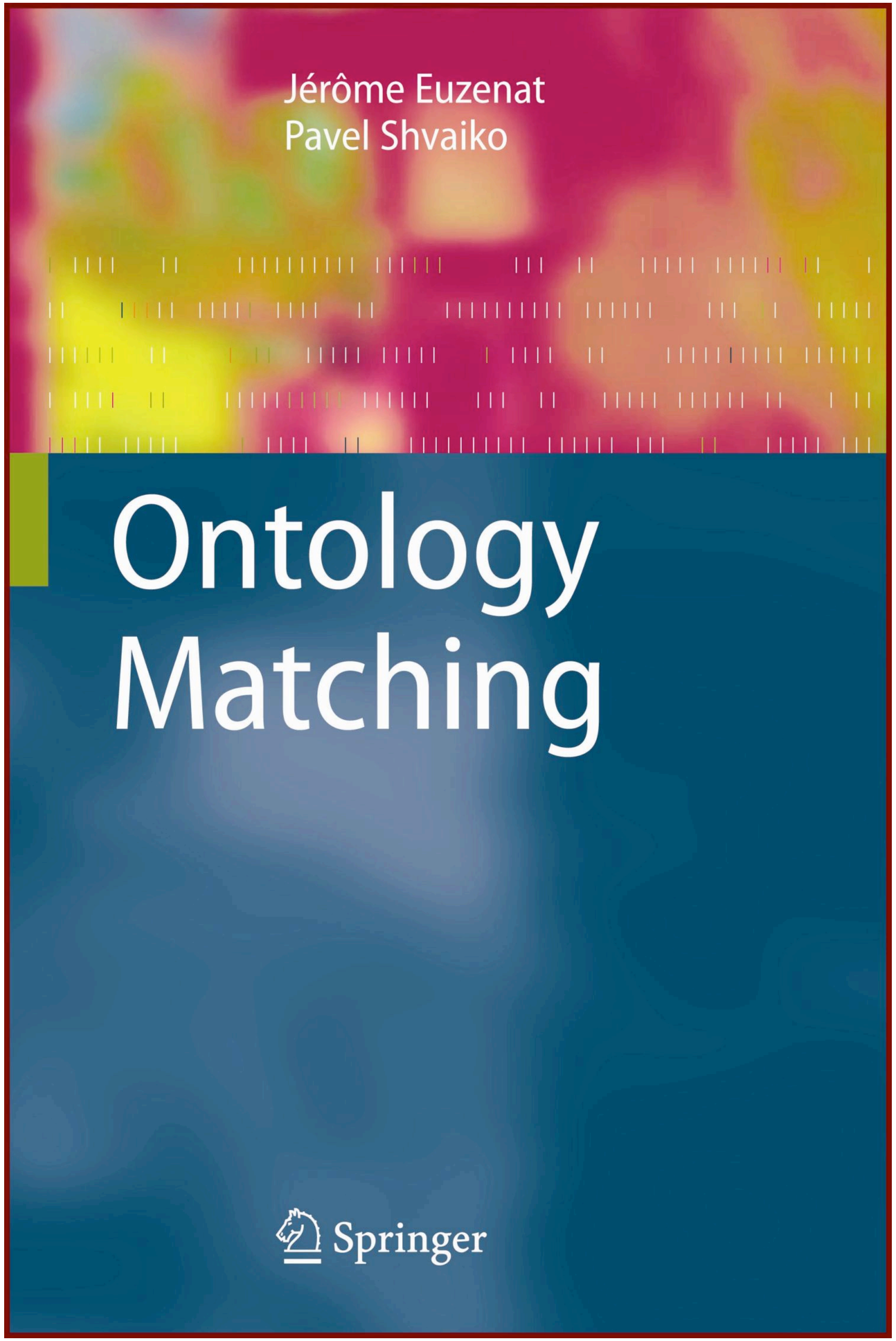
http://om2007.OntologyMatching.org

8/26 technical papers to be presented

Ontology Alignment Evaluation Initiative OAEI–2007 campaign

http://oaei.OntologyMatching.org/2007

Evalution of 17 systems to be presented


Jérôme Euzenat
Pavel Shvaiko
Ontology Matching
Springer

Thank you

for your attention and interest!