Ontology matching tutorial

Jérôme Euzenat & LIG
Montbonnot Saint-Martin, France
Jerome.Euzenat@inria.fr

Pavel Shvaiko
Informatica Trentina SpA
Trento, Italy
pavel.shvaiko@infotn.it

October 2014

Goals of the tutorial

- Provide an introduction to ontology matching
- Discuss practical and methodological issues
- Demonstrate and use (advanced) matching technology
- Motivate future research

Outline

What is an ontology?

An ontology typically provides a vocabulary that describes a domain of interest and a specification of the meaning of terms used in the vocabulary.

Depending on the precision of this specification, the notion of ontology encompasses several data and conceptual models, including, sets of terms, classifications, thesauri, database schemas, or fully axiomatized theories.

Various forms of ontologies

'Ordinary' glossaries — Data dictionaries — Structured glossaries — Principled, informal hierarchies — XML schemas — Formal taxonomies — Description logics

→ *expressivity*

Terms — Ad hoc hierarchies — Thesauri — XML DTDs — Database schemas — Entity-relationship models — Frames — Logics

Glossaries and data dictionaries — **Thesauri and taxonomies** — **Metadata and data models** — **Formal ontologies**

adapted from [Uschold and Gruninger, 2004]

Being serious about the semantic web

- It is not one guy's ontology.
- It is not several guys' common ontology.
- It is many guys and girls' many ontologies.
- So it is a mess, but a meaningful mess.

Living with heterogeneity

The semantic web will be:

- huge,
- dynamic,
- heterogeneous.

These are not bugs, these are features.
We must learn to live with them and master them.

The heterogeneity problem

Often resources expressed in different ways must be reconciled before being used.
Mismatch between formalized knowledge can occur when:

- different languages are used,
- different terminologies are used,
- different modelling is used.

On reducing heterogeneity

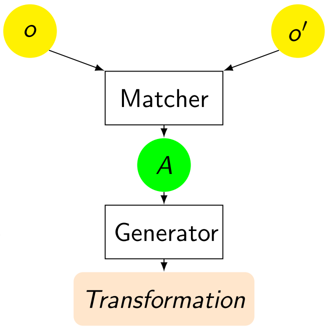
Reconciliation can be performed in 2 steps

Match,

thereby determine an alignment

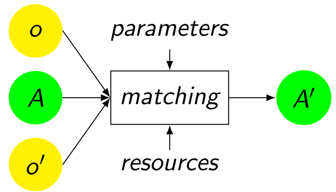
Generate

a processor (for merging, transforming, etc.)

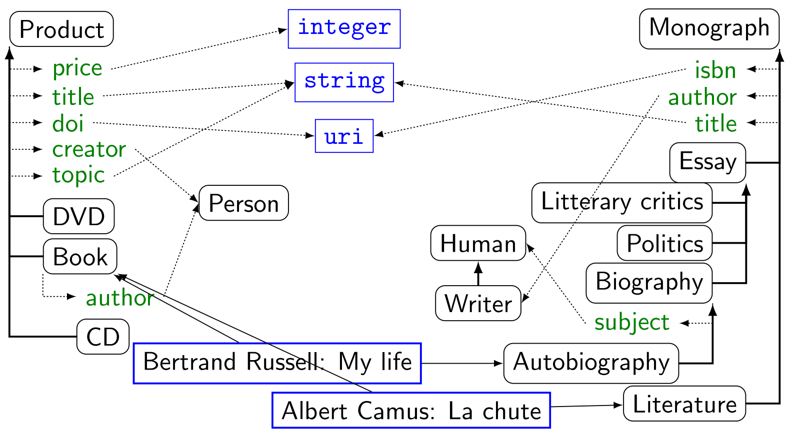


Matching can be achieved at run time or at design time.

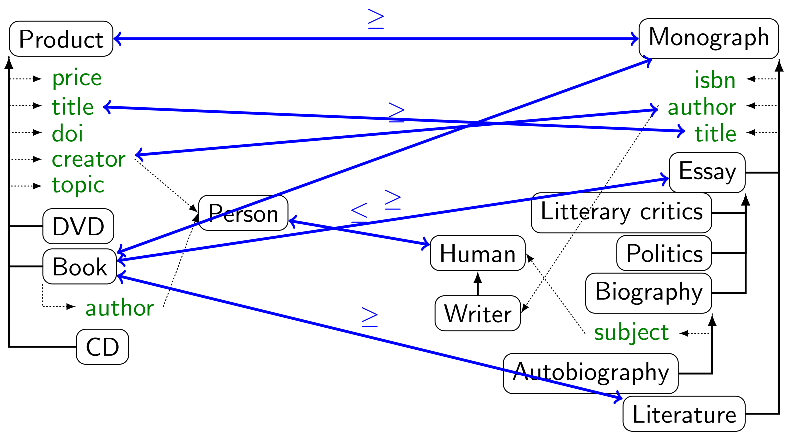
The matching process



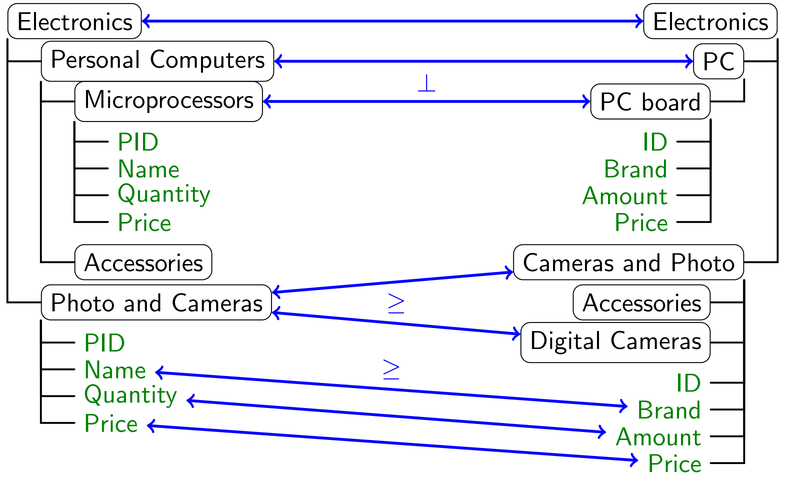
Motivation: two ontologies



Motivation: two ontologies



Motivation: two XML schemas



Correspondence

Definition (Correspondence)

Given two ontologies o and o', a **correspondence** between o and o' is a 3-uple: $\langle e, e', r \rangle$ such that:

- e and e' are entities of o and o', for instance, classes, XML elements;
- r is a relation, for instance, equivalence ($=$), more general ($\sqsupseteq$), disjointness ($\perp$).

Alignment

Definition (Alignment)

Given two ontologies o and o', an **alignment (A)** between o and o':

- is a set of correspondences on o and o'
- with some additional metadata (multiplicity: 1-1, 1-*, method, date, ...)

Terminology: a summary

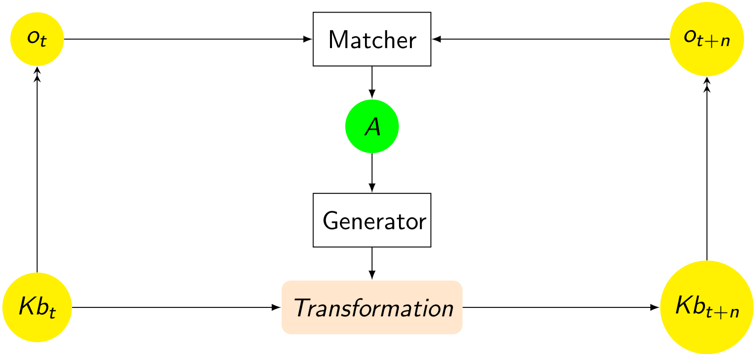
Matching is the process of finding relationships or correspondences between entities of different ontologies.

Alignment is a set of correspondences between two or more (in case of multiple matching) ontologies. The alignment is the output of the matching process.

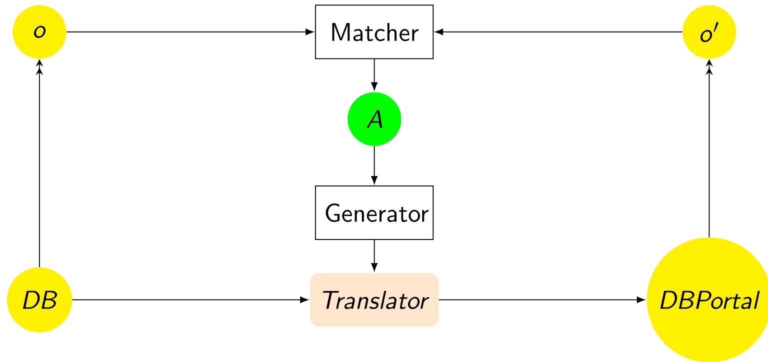
Correspondence is the relation supposed to hold according to a particular matching algorithm or individual, between entities of different ontologies.

Mapping is the oriented version of an alignment.

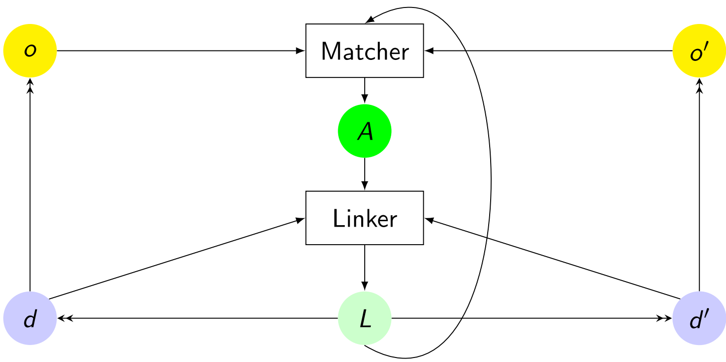
Applications: ontology evolution



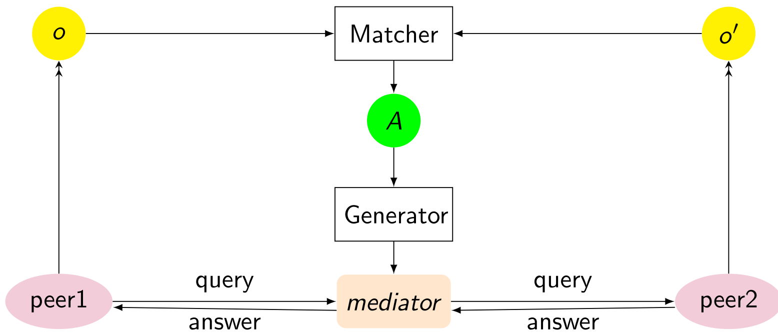
Applications: catalog integration



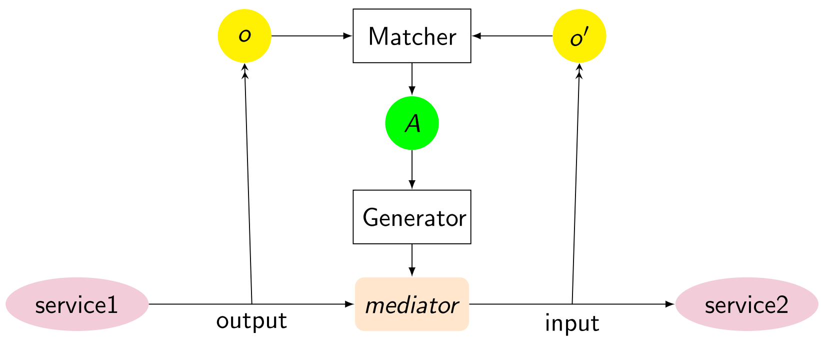
Applications: linked data interlinking



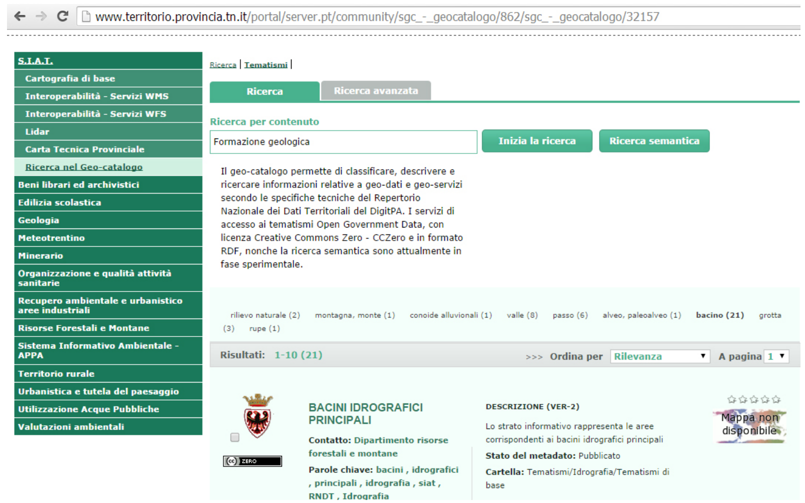
Applications: p2p information sharing



Applications: web service composition



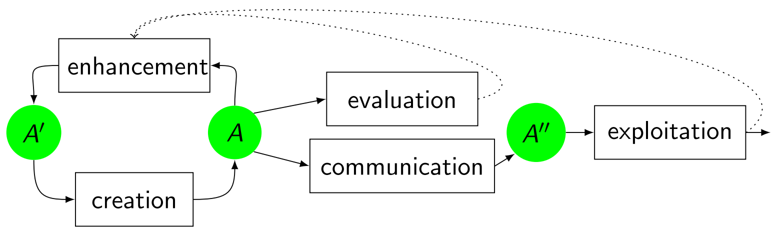
Applications: query answering



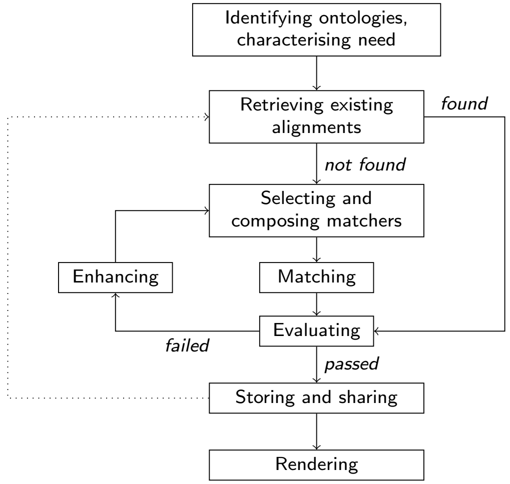
Applications requirements

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

The alignment life cycle



The matching methodology workflow



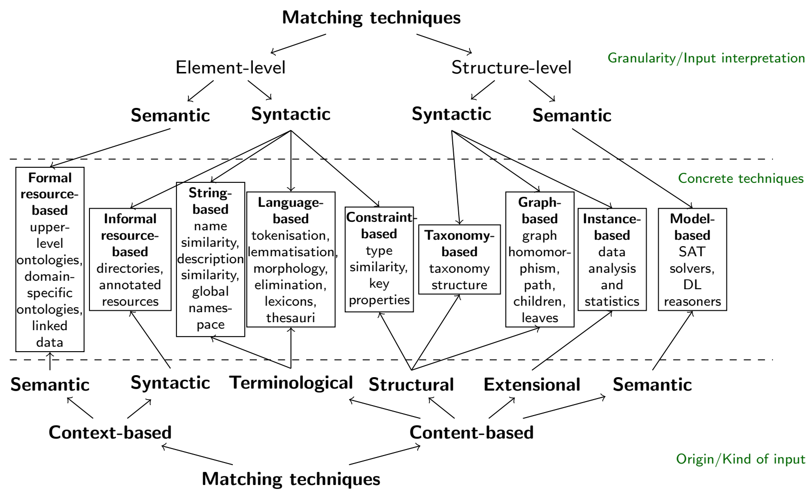
Matching dimensions

- Input dimensions
  - Underlying models (e.g., XML, OWL)
  - Schema-level vs. instance-level
  - Content vs. context
- Process dimensions
  - Approximate vs. exact
  - Interpretation of the input
- Output dimensions
  - Cardinality (e.g., 1-1, 1-*)
  - Equivalence vs. diverse relations (e.g., subsumption)
  - Graded vs. absolute confidence

Three layers

- The upper layer
  - Granularity of match
  - Interpretation of the input information
- The middle layer represents classes of matching techniques
- The lower layer
  - Origin
  - Kind of input information

Classification of matching techniques



Basic methods: string-based

- Prefix
  - takes as input two strings and checks whether the first string starts with the second one
  - net = network; but also hot = hotel
- Suffix
  - takes as input two strings and checks whether the first string ends with the second one
  - ID = PID; but also word = sword

(e.g., COMA, SF, S-Match, OLA)

Basic methods: string-based

Edit distance

- takes as input two strings and calculates the number of edition operations, (e.g., insertions, deletions, substitutions) of characters required to transform one string into another
- normalized by length of the maximum string
- EditDistance(NKN,Nikon) = N~~i~~K~~o~~N/5 = 2/5 = 0.4
- EditDistance(editeur,editor) = edit$\frac{e}{o}$~~ur~~/7= 3/7 = 0.43

(e.g., S-Match, OLA, Anchor-Prompt)

Basic methods: string-based

- N-gram
  - takes as input two strings and calculates the number of common n-grams (i.e., sequences of n characters) between them, normalized by $max(length(string1), length(string2))$
  - $trigram(3)$ for the string nikon are nik, iko, kon

(e.g., COMA, S-Match)

Basic methods: language-based

- Tokenization
  - parses names into tokens by recognizing punctuation, cases
  - Hands-Free_Kits → ⟨ hands, free, kits ⟩
- Lemmatization
  - analyses morphologically tokens in order to find all their possible basic forms
  - Kits → Kit

(e.g., COMA, Cupid, S-Match, OLA)

Basic methods: language-based

- Elimination
  - discards “empty” tokens that are articles, prepositions, conjunctions, etc.
  - a, the, by, type of, their, from

(e.g., Cupid, S-Match)

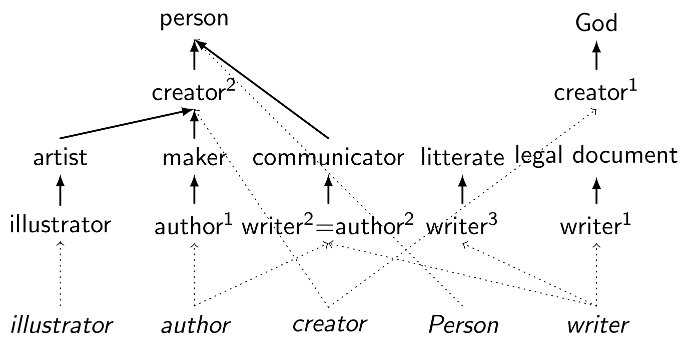
Basic methods: linguistic resources

- Sense-based: WordNet
  - $A \sqsubseteq B$ if A is a hyponym or meronym of B
    - Brand $\sqsubseteq$ Name
  - $A \sqsupseteq B$ if A is a hypernym or holonym of B
    - Europe $\sqsupseteq$ Greece
  - $A = B$ if they are synonyms
    - Quantity = Amount
  - $A \perp B$ if they are antonyms or the siblings in the part of hierarchy
    - Microprocessors $\perp$ PC Board

(e.g., Artemis, CtxMatch, S-Match)

Basic methods: linguistic resources

- Sense-based: WordNet hierarchy distance



Some other measures (e.g., Resnik measure) depend on the frequency of the terms in the corpus made of all the labels of the ontologies.
(e.g., S-Match)

Basic methods: multilingual matching

Ontologies can be multilingual if they use several different languages, e.g., EN, IT, FR. Matching can be done by comparing to a pivot language or through cross-translation. We distinguish between:

monolingual matching, which matches two ontologies based on their labels in a single language, such as English;

multilingual matching, which matches two ontologies based on labels in a variety of languages, e.g., English, French and Spanish. This can be achieved by parallel monolingual matching of terms or crosslingual matching of terms in different languages;

crosslingual matching, which matches two ontologies based on labels in two identified different languages, e.g., English vs. French.

Basic methods: constraint-based

- Datatype comparison
  - $integer < real$
  - $date \in [1/4/2005\ 30/6/2005] < date[year = 2005]$
  - $\{a, c, g, t\}[1-10] < \{a, c, g, u, t\}+$
- Multiplicity comparison
  - $[1\ 1] < [0\ 10]$

Can be turned into a distance by estimating the ratio of domain coverage of each datatype.
(e.g., OLA, COMA)

Basic methods: extensional

$$\epsilon : C \rightarrow E$$

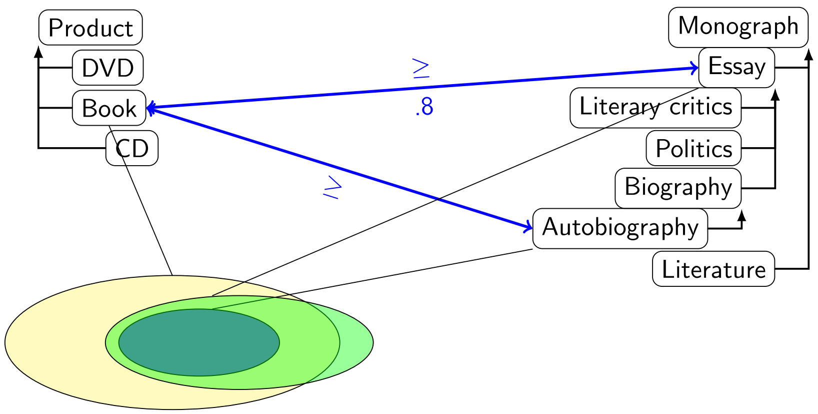
E can be a set of instances, a set of documents which are indexed by concepts, a set of items, e.g., people, which use these concepts.
Two cases:

- E is common to both ontologies;
- E depends on the ontology. This can be reduced to the former case by identification or record linkage techniques.

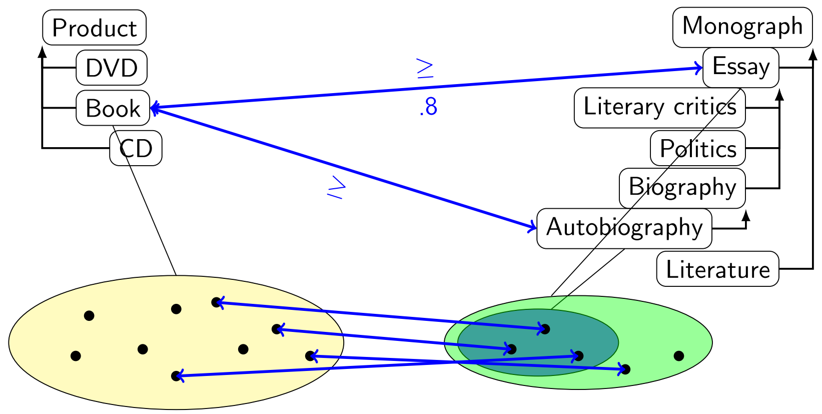
Techniques:

- statistical and machine learning techniques infer and compare the characteristics of populations;
- set-theoretic techniques compare the extensions;

Extensional techniques



Extensional techniques

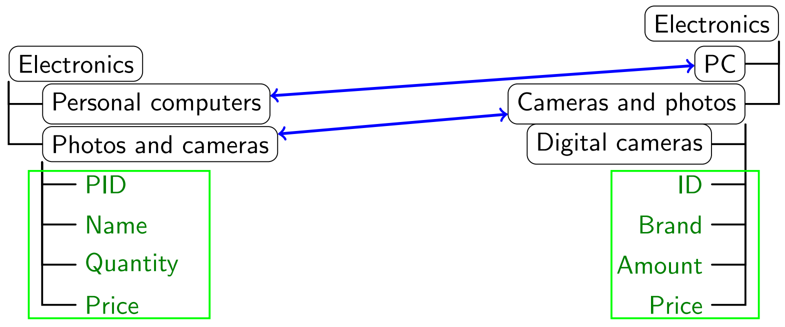


Global methods: tree-based

- Children
  - Two non-leaf schema elements are structurally similar if their immediate children sets are highly similar
- Leaves
  - Two non-leaf schema elements are structurally similar if their leaf sets are highly similar, even if their immediate children are not

(e.g., Cupid, COMA)

Global methods: tree-based



(e.g., Cupid, COMA)

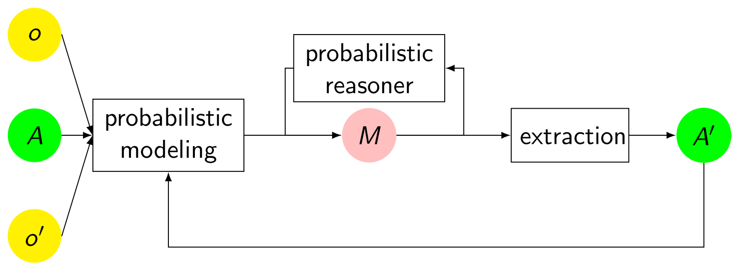
Global methods: graph-based

- Iterative fix point computation
  - If the neighbors of two nodes of the two ontologies are similar, they will be more similar.

(e.g., SF, OLA)

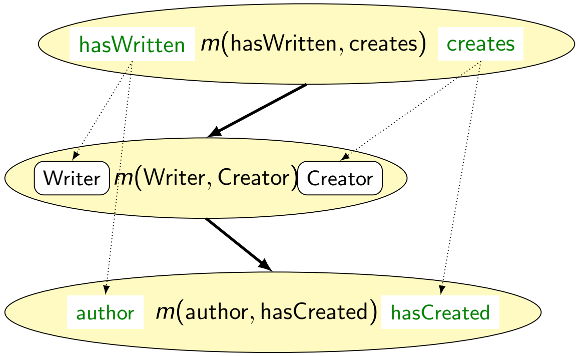
Global methods: probabilistic matching

Probabilistic methods, such as Bayesian networks or Markov networks, can be used universally in ontology matching, e.g., to enhance some available matching candidates.



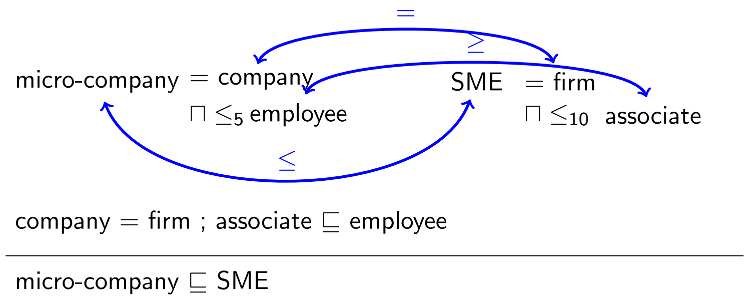
Probabilistic matching: bayesian networks example

Bayesian networks are made up of (*i*) a directed acyclic graph, containing nodes (also called variables) and arcs, and (*ii*) a set of conditional probability tables. Arcs between nodes stand for conditional dependencies and indicate the direction of influence.



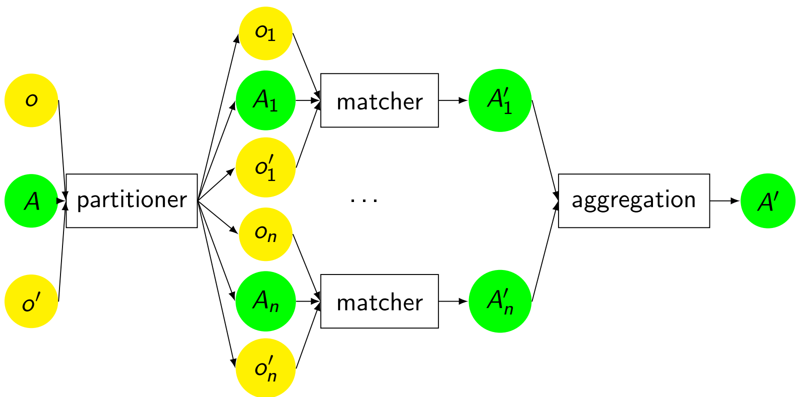
Global methods: model-based

Description logics (DL)-based

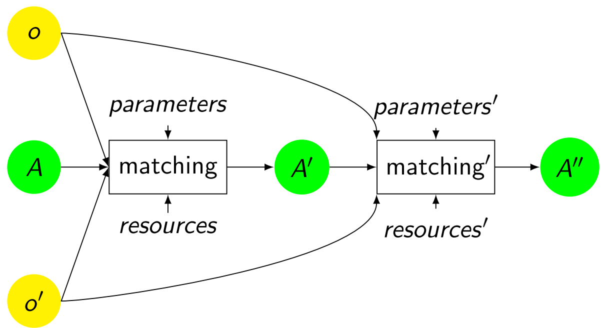


Ontology partitioning and search-space pruning

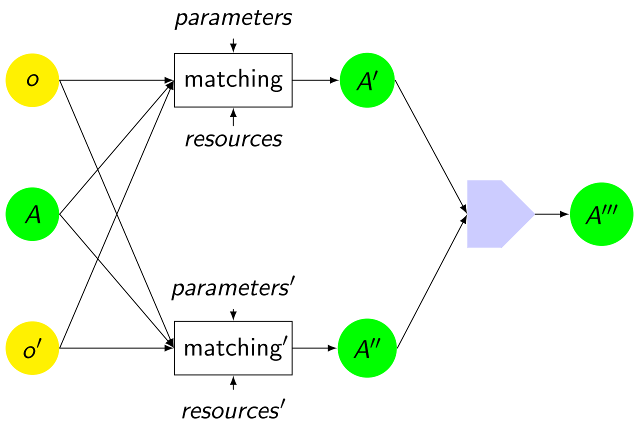
Partitioning: split large ontologies into smaller ontologies, and match these smaller ontologies, e.g., Falcon-AO, TaxoMap. **Pruning:** dynamically ignore parts of large ontologies when matching, e.g., AROMA, LogMap.



Sequential composition



Parallel composition



Context-based matching (CBM)

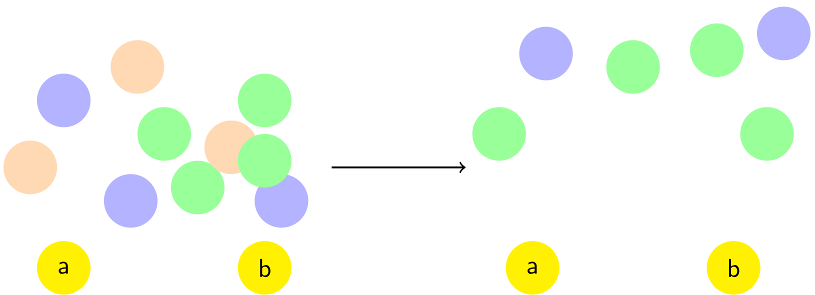
- Using the ontologies on the web as context
- Composing the relations obtained through these ontologies

The seven steps:

1. Ontology arrangement
2. Contextualization
3. Ontology selection
4. Local inference
5. Global inference
6. Composition
7. Aggregation

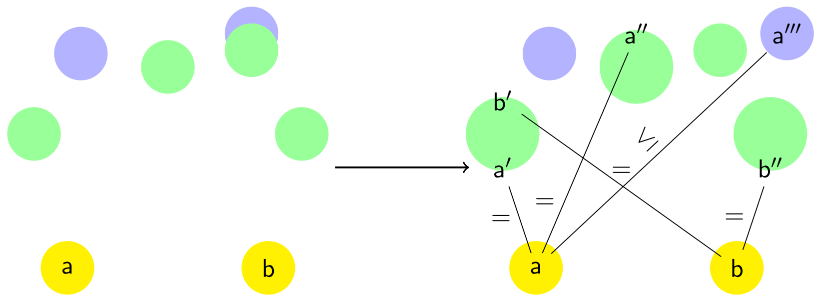
CBM step 1: ontology arrangement

Ontology arrangement preselects and ranks the ontologies to be explored as intermediate ontologies



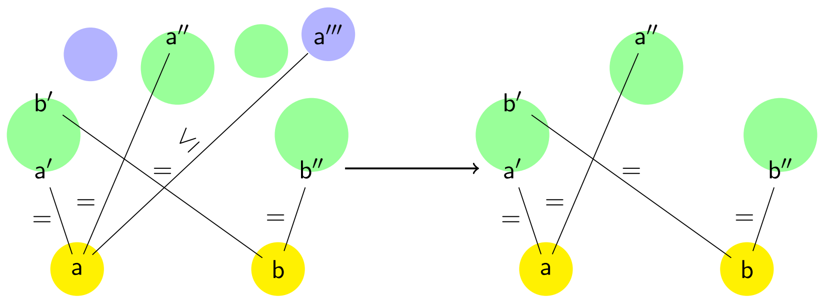
CBM step 2: contextualization

Contextualisation (or **anchoring**) finds anchors between the ontologies to be matched and the candidate intermediate ontologies



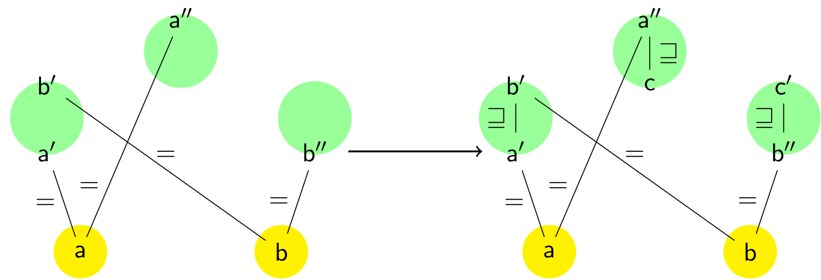
CBM step 3: selection

Ontology selection restricts the candidate ontologies that will actually be used



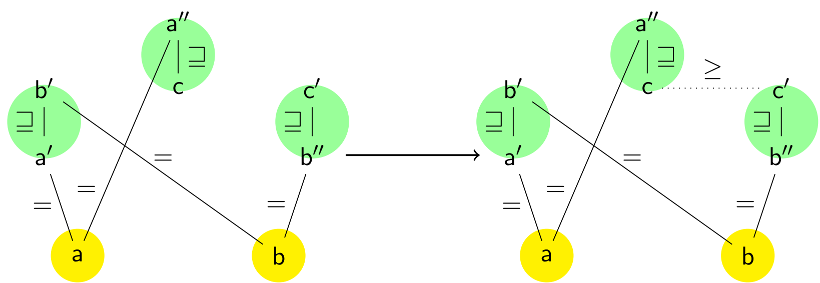
CBM step 4: local inference

Local inference obtains relations between entities of a single ontology. It may be reduced to logical entailment



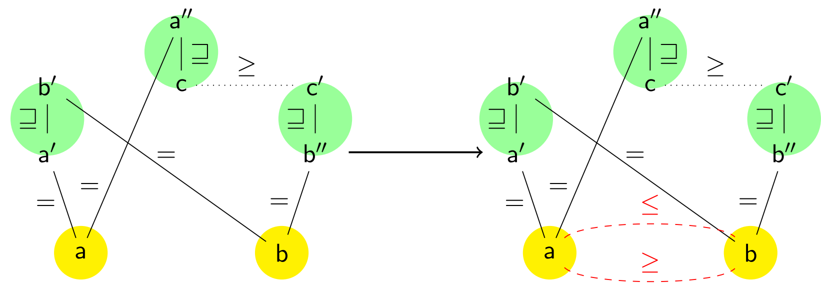
CBM step 5: global inference

Global inference finds relations between two concepts of the ontologies to be matched by concatenating relations obtained from local inference and correspondences across intermediate ontologies



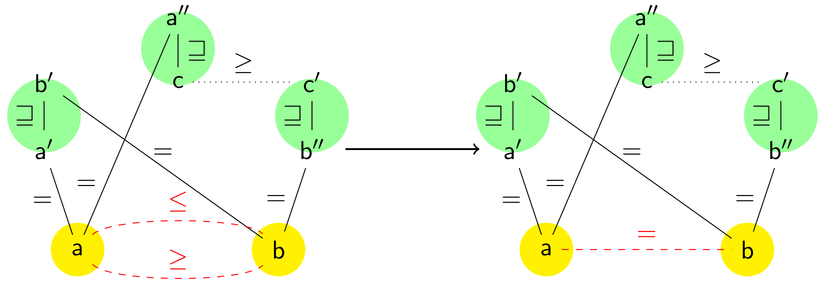
CBM step 6: composition

Composition determines the relations holding between the source and target entities by composing the relations in the path (sequence of relations) connecting them

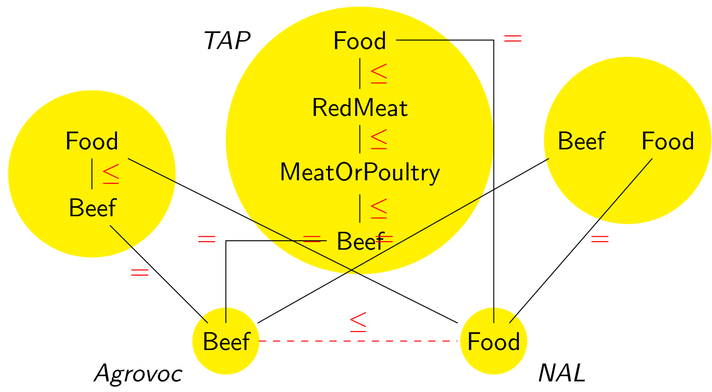


CBM step 7: aggregation

Aggregation combines relations obtained between the same pair of entities

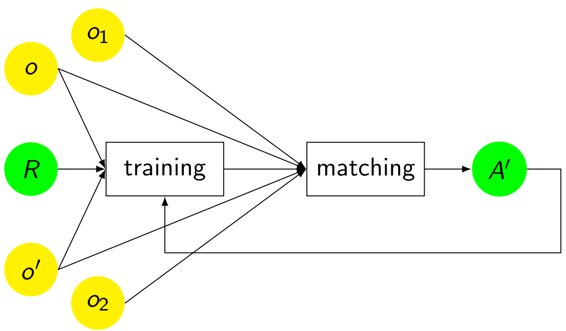


CBM example: Scarlet



Matching learning

These algorithms learn how to sort alignments through the presentation of many correct alignments (positive examples) and incorrect alignments (negative examples)



Matching learning: examples

A multistrategy learning approach is useful when several learners are used, each one handling a particular kind of pattern that it learns best, e.g., GLUE, CSR, YAM++.

Various well-known machine learning methods, which had been used for text categorisation, were also applied in ontology matching:

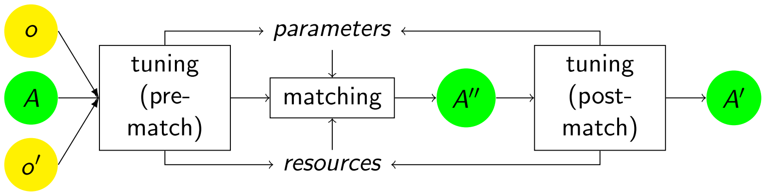
- Bayes learning,
- WHIRL learning,
- neural networks,
- support vector machines,
- decision trees.

And, in many cases the Weka data mining software was used.

Tuning

Tuning refers to the process of adjusting a matcher for a better functioning in terms of:

- better **quality** of matching results, measured, e.g., through precision or F-measure, and
- better **performance** of a matcher, measured through resource consumption, e.g., execution time, main memory.



Tuning: examples

From a methodological point of view, tuning may be applied at various levels of architectural granularity:

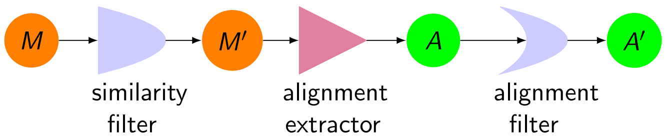
- for choosing a specific matcher, such as edit distance, from a library of matchers,
- for setting parameters of the matcher chosen, e.g., cost of edit distance operations,
- for aggregating the results of several matchers, e.g., through weighting,
- for enforcing constraints, such as 1:1 alignments,
- for selecting the final alignment, e.g., through thresholds.

Informed decisions, for instance, for choosing a specific threshold of 0.55 vs. 0.57 vs. 0.6, should be made.

A variety of systems have explored different possibilities, e.g., eTuner, MatchPlaner, ECOMatch, AMS.

Similarity filter, alignment extractor and alignment filter

Many algorithms are based on similarity or distance computation. A number of operations can be based on similarity/distance matrices.



Filtering similarities: thresholding

- Hard threshold retains all the correspondence above threshold n;
- Delta threshold consists of using as a threshold the highest similarity value out of which a particular constant value d is subtracted;
- Proportional threshold consists of using as a threshold the percentage of the highest similarity value;
- Percentage retains the n% correspondences above the others.

Extracting alignments

	Book	Translator	Publisher	Writer
Product	.84	0.	.90	.12
Provider	.12	0.	.84	.60
Creator	.60	.05	.12	.84

- Greedy algorithm: 1.86 (stable marriage)

Extracting alignments

	Book	Translator	Publisher	Writer
Product	.84	0.	.90	.12
Provider	.12	0.	.84	.60
Creator	.60	.05	.12	.84

- Greedy algorithm: 1.86 (stable marriage)
- Permutation: 2.1 (better)

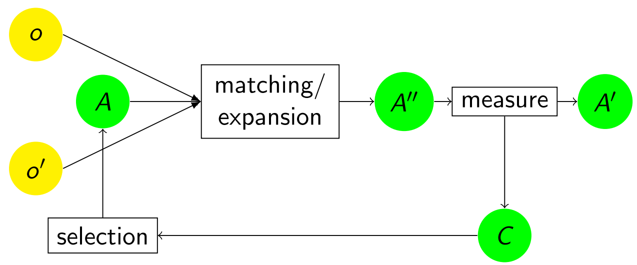
Extracting alignments

	Book	Translator	Publisher	Writer
Product	.84	0.	.90	.12
Provider	.12	0.	.84	.60
Creator	.60	.05	.12	.84

- Greedy algorithm: 1.86 (stable marriage)
- Permutation: 2.1 (better)
- Maximal weight match: 2.52 (optimal)

Alignment improvement

These algorithms measure some quality of a produced alignment, reduce the alignment, so that the quality may improve, and possibly iterate by expanding the resulting alignment, e.g., LogMap, ASMOV, ALCOMO.

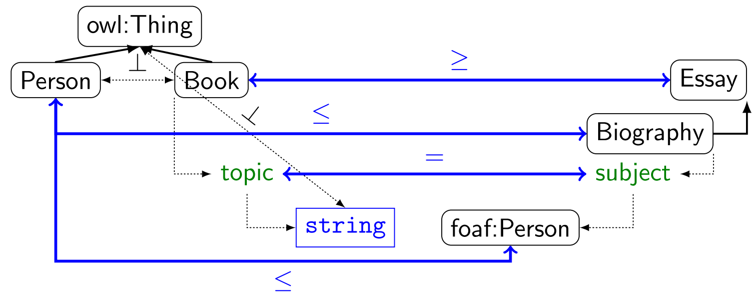


Alignment improvement: quality measures

Quality measures are the main ingredients for improvement. Contrary to the evaluation measures, such as precision and recall, these must be intrinsic measures of the alignment (they do not depend on any reference):

- threshold on confidence or average confidence,
- cohesion measures between matched entities, i.e., their neighbours are matched with each other,
- ambiguity degree, i.e., proportion of classes matched to several other classes,
- agreement or non-disagreement between the aligned ontologies,
- violation of some constraints, e.g., acyclicity in the correspondence paths,
- satisfaction of syntactic anti-patterns,
- consistency and coherence.

Alignment improvement: alignment debugging



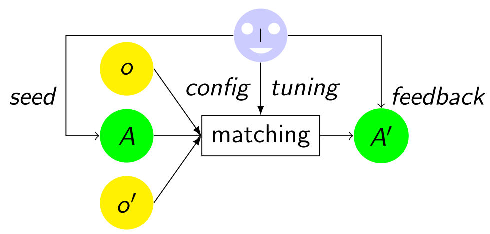
User involvement

- In traditional applications semi-automatic matching/design-time interaction is a promising way to improve quality of the results
- Burdenless to the user interaction schemes
  - Usability
  - Scalability of visualization
- Exploit the user feedback
  - to adjust matcher parameters
  - to take it as (partial) input alignment to a matcher
  - ...
- In dynamic settings, agents involved in the matching process can negotiate the mismatches in a fully automated way

Individual matching

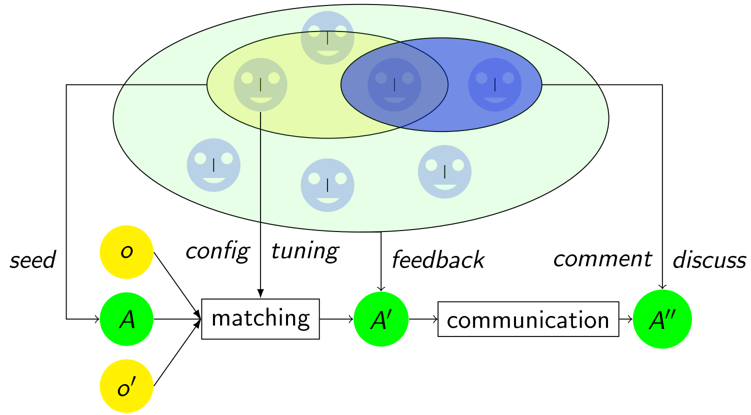
There are at least three areas in which users may be involved:

- by providing initial alignments to the system (before matching),
- by configuring and tuning the system, and
- by providing feedback to matchers in order for them to adapt their results.



Collective matching

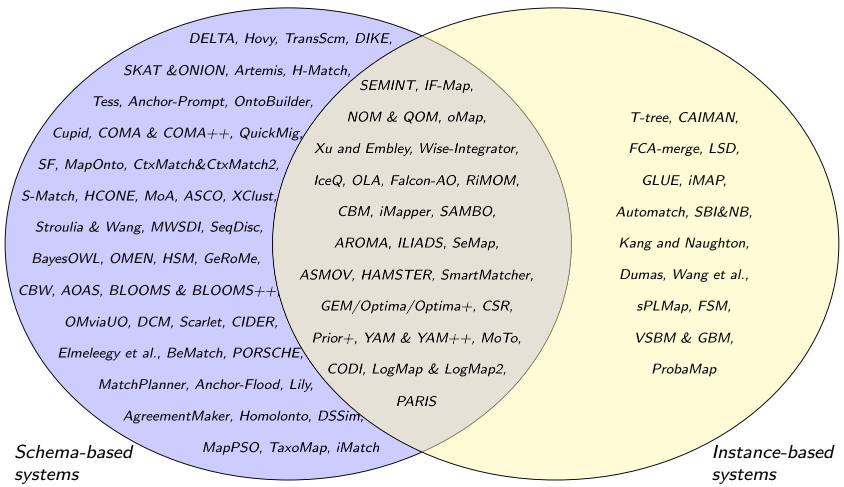
Besides involving a single user at a time, mostly in a synchronous fashion, matching may also be a collective effort in which several users are involved.



Social and collaborative ontology matching

- The network effect:
  - Each person has to do a small amount of work
  - Each person can improve on what has been done by others
  - Errors remain in minority
- A community of people can share alignments and argue about them by using annotations
- The key issues are to:
  - Provide adequate annotation support and description units
  - Handle adequately contradictory and incomplete alignments
  - Incentivise active user participation
  - Handle adequately the malicious users

An overview of the state of the art systems



State of the art systems

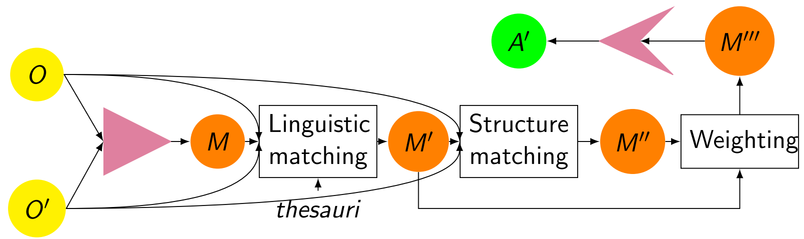
100+ matching systems exist, . . . we consider some of them

- Cupid (U. of Washington, Microsoft Corporation and U. of Leipzig)
- S-Match (U. of Trento)
- OLA (INRIA Rhône-Alpes and U. de Montréal)
- Falcon-AO (China Southwest U.)
- RiMOM (Tsinghua U.)
- ASMOV (INFOTECH Soft, Inc., U. of Miami)
- LogMap (U. of Oxford)
- eTuner (U. of Illinois and The MITRE Corporation)
- . . .

Cupid

- Schema-based
- Computes similarity coefficients in the [0 1] range
- Performs linguistic and structure matching
- Sequential system

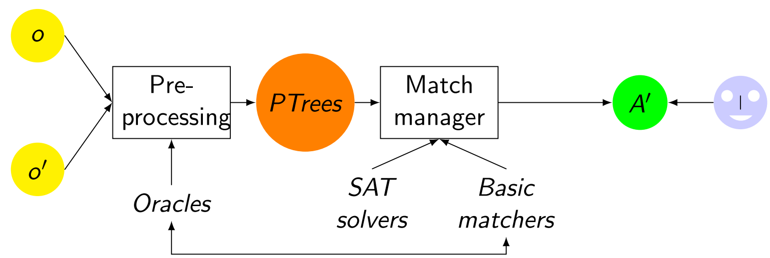
Cupid architecture



S-Match

- Schema-based
- Computes equivalence ($=$), more general ($\sqsupseteq$), less general ($\sqsubseteq$), disjointness ($\perp$)
- Transforms each ontology into a propositional theory based on external resources (WordNet definitions of terms) and ontology structure
- Sequential system with a composition at the element level

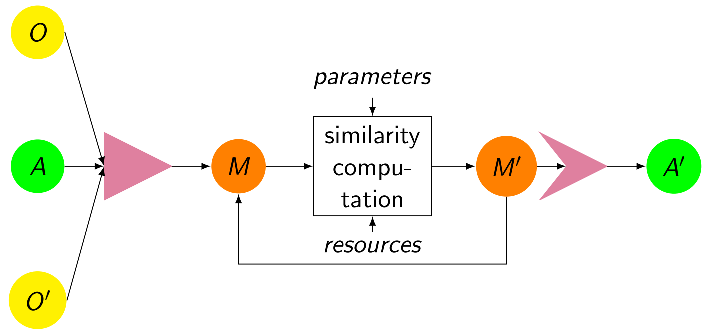
S-Match architecture



OLA

- Schema- and Instance-based
- Computes dissimilarities + extracts alignments (equivalences in the [0 1] range)
- Based on terminological (including linguistic) and structural (internal and relational) distances
- Neither sequential nor parallel

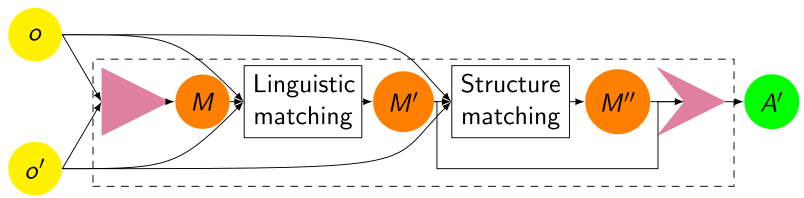
OLA architecture



Falcon-AO

- Schema- and instance-based
- Sequence of string-based and structural matcher
- Does not use the structural matcher if the terminological match is high enough
- String-based matcher based on so-called virtual documents
- Structural matcher close to OLA's
- Partition the ontologies so that they can be processed faster

Falcon-OA architecture

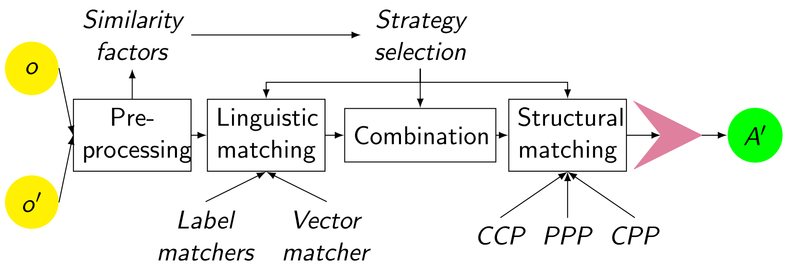


RiMOM

- Schema- and instance-based
- Dynamic strategy selection based on pre-processing
- Sequence of linguistic and structural matchers
- Linguistic matching is based on edit distance, WordNet and vector distance
- Structural matcher implements variations of similarity flooding

Problem Applications Methodology Classification Methods Process Systems Use Evaluation Conclusions

RiMOM architecture



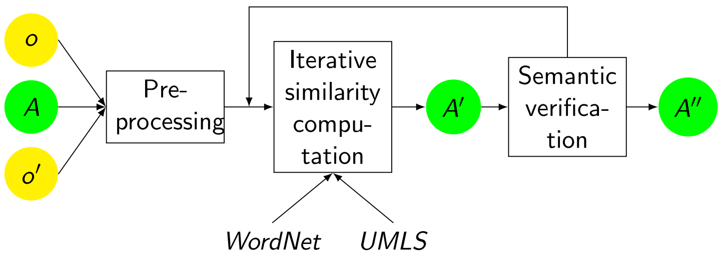
Problem Applications Methodology Classification Methods Process Systems Use Evaluation Conclusions

ASMOV

- Schema- and instance-based
- Iterative similarity computation with sematic verification
- Matchers: string-based, language based, WordNet UMLS, iterative fix point computation
- Verification through rule-based (anti-patterns) inference

Problem Applications Methodology Classification Methods Process Systems Use Evaluation Conclusions

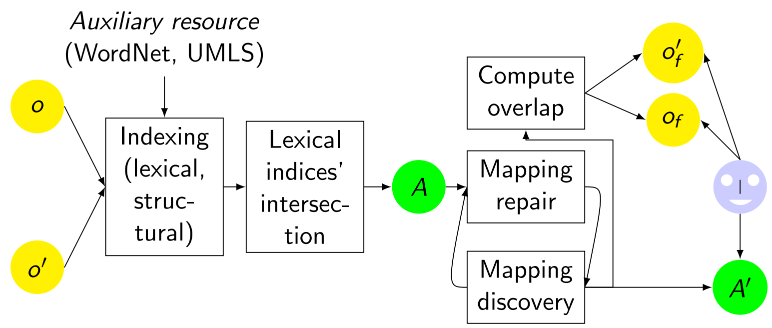
ASMOV architecture



LogMap

- Schema- and instance-based
- Applies partitioning and pruning of large ontologies
- Initial anchors through indices' intersection (exact strings)
- Mapping repair through propositional satisfiability
- String-based mapping discovery from anchors through class hierarchies

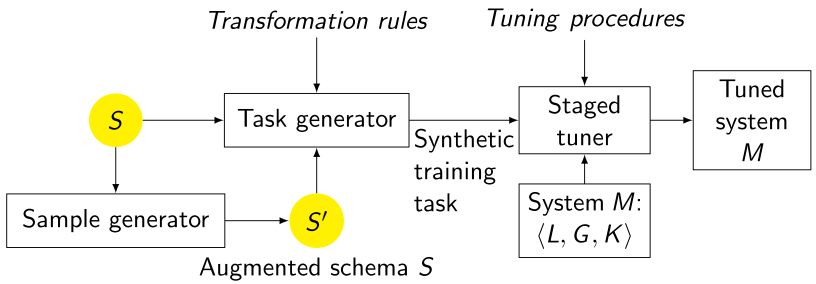
LogMap architecture



eTuner

- A metamatching system
- Automatically tunes a matching system for a particular task by choosing the most effective matchers and the best parameters to be used
- A matching system is modeled as a triple $\langle L, G, K \rangle$:
  - L is a library of matching components, e.g., edit distance; combiners, e.g., through averaging; constraint enforcers, e.g., pre-defined domain constraints; match selectors, e.g., thresholds.
  - G is a directed graph which encodes the execution flow among the components of the given matching system.
  - K is a set of knobs to be set.
- Two phases: training through synthetic workload and (greedy) search.

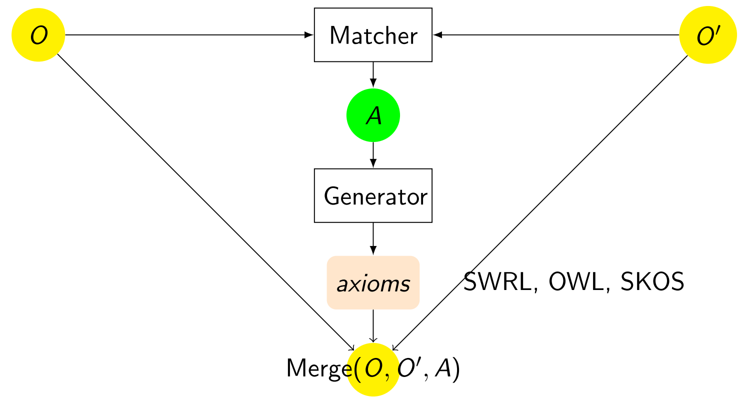
eTuner architecture



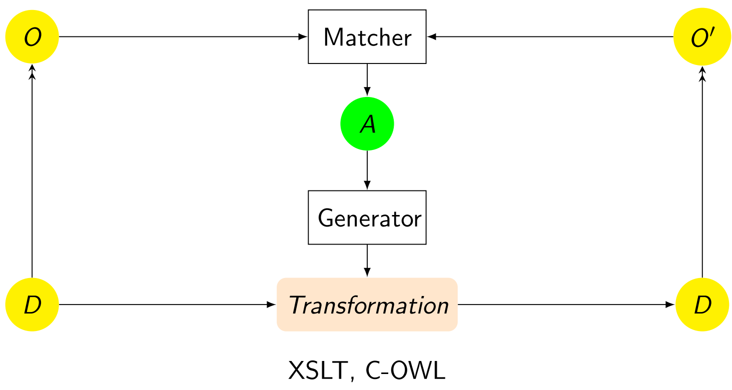
Main operations

- $Merge(o, o', A) = o''$
- $Translate(d, A) = d'$
- $Interlink(d, d', A) = L$
- $TransformQuery(q, A) = q'$ and $Translate(a', Invert(A)) = a$
- ...

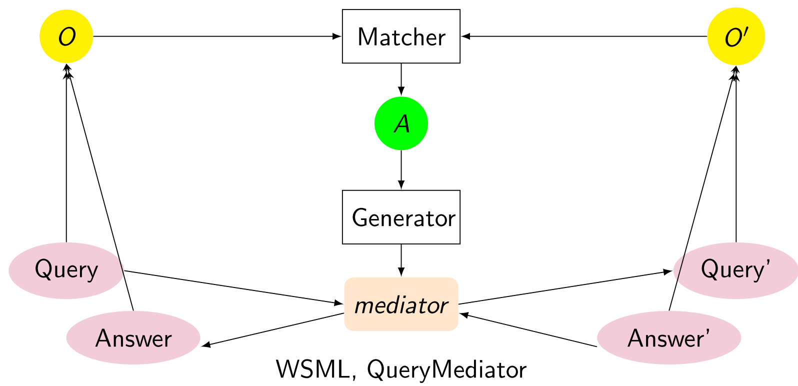
Merging



Data translation



Query mediator



Evaluation of matching algorithms

http://oaei.ontologymatching.org

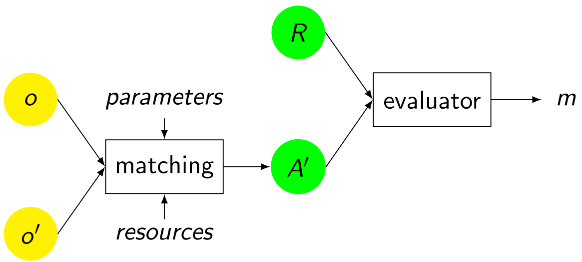
Goal: improvement of matching algorithms through comparison, measure of the evolution of the field.

- Yearly campaigns comparing algorithms on different test cases
- Participants submit their alignments in a standard format
- We use alignment API for comparing these formats with reference alignments
- Various degrees of blindness, expressiveness, realism
- Tests and results are published on the web site

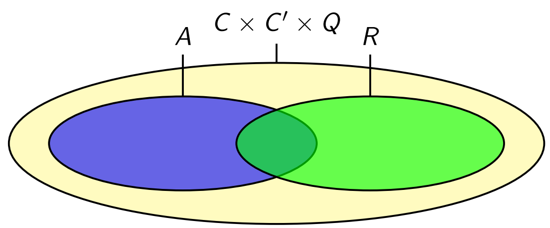
6 tracks, 8 test cases, 23 participants in 2013

test	formalism	relations	confidence	modalities	language	SEALS
benchmark	OWL	=	[0 1]	blind+open	EN	√
anatomy	OWL	=	[0 1]	open	EN	√
conference	OWL-DL	=, <=	[0 1]	blind+open	EN	√
large bio	OWL	=	[0 1]	open	EN	√
multifarm	OWL	=	[0 1]	open	CZ, CN, EN, ...	√
library	OWL	=	[0 1]	open	EN, DE	√
interactive	OWL-DL	=, <=	[0 1]	open	EN	√
rdft	RDF	=	[0 1]	blind	EN	

Evaluation process



Precision and recall

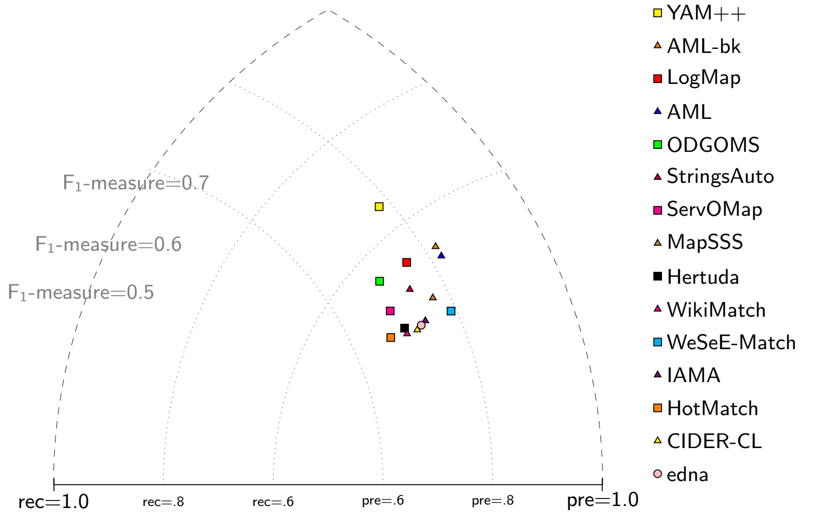


Definition (Precision, Recall)

Given a reference alignment R, the precision of some alignment A is given by

$$P(A,R) = \frac{|R \cap A|}{|A|} \text{ and recall is given by } R(A,R) = \frac{|R \cap A|}{|R|}.$$

2013: Precision-recall graph for the conference test case



Summary

- Heterogeneity of ontologies is in the nature of the semantic web;
- Ontology matching is part of the solution;
- It can be based on many different techniques;
- There are already numerous systems around;
- A relatively solid research field has emerged (tools, formats, evaluation, etc.) and it keeps making progress;
- But there remain serious challenges ahead.

Challenges

- Large-scale and efficient matching,
- Matching with background knowledge,
- Matcher selection, combination and tuning,
- User involvement,
- Social and collaborative matching,
- Uncertainty in matching,
- Reasoning with alignments,
- Alignment management.

and, of course, many others...

Acknowledgments

We thank all the participants of the Heterogeneity workpackage of the Knowledge Web network of excellence

knowledgeweb

In particular, we are grateful to Than-Le Bach, Jesus Barrasa, Paolo Bouquet, Jan De Bo, Jos De Bruijn, Rose Dieng-Kuntz, Enrico Franconi, Raúl García Castro, Manfred Hauswirth, Pascal Hitzler, Mustafa Jarrar, Markus Krötzsch, Ruben Lara, Malgorzata Mochol, Amedeo Napoli, Luciano Serafini, François Sharffe, Giorgos Stamou, Heiner Stuckenschmidt, York Sure, Vojtěch Svátek, Valentina Tamma, Sergio Tessaris, Paolo Traverso, Raphaël Troncy, Sven van Acker, Frank van Harmelen, and Ilya Zaihrayeu.

And more specifically to Marc Ehrig, Fausto Giunchiglia, Loredana Laera, Diana Maynard, Deborah McGuinness, Petko Valchev, Mikalai Yatskevich, and Antoine Zimmermann for their support and insightful comments

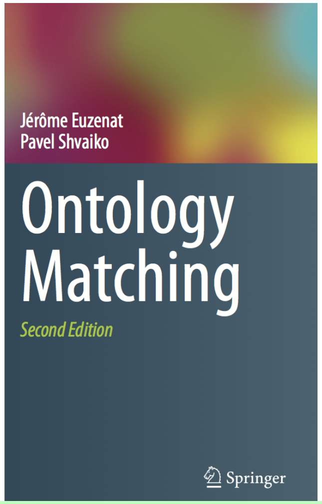
Part of this work was carried out while Pavel Shvaiko was with the University of Trento.

Ontology matching the book, 2nd edition

Jérôme Euzenat, Pavel Shvaiko

Ontology matching

1. Applications
2. The matching problem
3. Methodology
4. Classification
5. Basic similarity measures
6. Global matching methods
7. Strategies
8. Systems
9. Evaluation
10. Representation
11. User involvement
12. Processing



http://book.ontologymatching.org

Thank you

for your attention and interest!

Questions?

Jerome.Euzenat@inria.fr

Pavel.Shvaiko@infotn.it

http://www.ontologymatching.org