Results of the Ontology Alignment Evaluation Initiative 2013*

Bernardo Cuenca Grau[1], Zlatan Dragisic[2], Kai Eckert[3], Jérôme Euzenat[4], Alfio Ferrara[5], Roger Granada[6,7], Valentina Ivanova[2], Ernesto Jiménez-Ruiz[1], Andreas Oskar Kempf[8], Patrick Lambrix[2], Andriy Nikolov[9], Heiko Paulheim[3], Dominique Ritze[3], François Scharffe[10], Pavel Shvaiko[11], Cássia Trojahn[7], and Ondřej Zamazal[12]

[1] University of Oxford, UK
{berg,ernesto}@cs.ox.ac.uk
[2] Linköping University & Swedish e-Science Research Center, Linköping, Sweden
{zlatan.dragisic,valentina.ivanova,patrick.lambrix}@liu.se
[3] University of Mannheim, Mannheim, Germany
{kai,heiko,dominique}@informatik.uni-mannheim.de
[4] INRIA & LIG, Montbonnot, France
Jerome.Euzenat@inria.fr
[5] Università degli studi di Milano, Italy
alfio.ferrara@unimi.it
[6] Pontifícia Universidade Católica do Rio Grande do Sul, Porto Alegre, Brazil
roger.granada@acad.pucrs.br
[7] IRIT & Université Toulouse II, Toulouse, France
cassia.trojahn@irit.fr
[8] GESIS – Leibniz Institute for the Social Sciences, Cologne, Germany
andreas.kempf@gesis.org
[9] Fluid operations, Walldorf, Germany
andriy.nikolov@fluidops.com
[10] LIRMM, Montpellier, France
francois.scharffe@lirmm.fr
[11] TasLab, Informatica Trentina, Trento, Italy
pavel.shvaiko@infotn.it
[12] University of Economics, Prague, Czech Republic
ondrej.zamazal@vse.cz

Abstract. Ontology matching consists of finding correspondences between semantically related entities of two ontologies. OAEI campaigns aim at comparing ontology matching systems on precisely defined test cases. These test cases can use ontologies of different nature (from simple thesauri to expressive OWL ontologies) and use different modalities, e.g., blind evaluation, open evaluation and consensus. OAEI 2013 offered 6 tracks with 8 test cases followed by 23 participants. Since 2010, the campaign has been using a new evaluation modality which provides more automation to the evaluation. This paper is an overall presentation of the OAEI 2013 campaign.

* This paper improves on the "Preliminary results" initially published in the on-site proceedings of the ISWC workshop on Ontology Matching (OM-2013). The only official results of the campaign, however, are on the OAEI web site.

1 Introduction

The Ontology Alignment Evaluation Initiative[1] (OAEI) is a coordinated international initiative, which organizes the evaluation of the increasing number of ontology matching systems [11, 14]. The main goal of OAEI is to compare systems and algorithms on the same basis and to allow anyone for drawing conclusions about the best matching strategies. Our ambition is that, from such evaluations, tool developers can improve their systems.

Two first events were organized in 2004: (i) the Information Interpretation and Integration Conference (I3CON) held at the NIST Performance Metrics for Intelligent Systems (PerMIS) workshop and (ii) the Ontology Alignment Contest held at the Evaluation of Ontology-based Tools (EON) workshop of the annual International Semantic Web Conference (ISWC) [28]. Then, a unique OAEI campaign occurred in 2005 at the workshop on Integrating Ontologies held in conjunction with the International Conference on Knowledge Capture (K-Cap) [3]. Starting from 2006 through 2012 the OAEI campaigns were held at the Ontology Matching workshops collocated with ISWC [12, 10, 5, 7–9, 1]. In 2013, the OAEI results were presented again at the Ontology Matching workshop[2] collocated with ISWC, in Sydney, Australia.

Since 2011, we have been promoting an environment for automatically processing evaluations (§2.2), which has been developed within the SEALS (Semantic Evaluation At Large Scale) project[3]. SEALS provided a software infrastructure, for automatically executing evaluations, and evaluation campaigns for typical semantic web tools, including ontology matching. For OAEI 2013, almost all of the OAEI data sets were evaluated under the SEALS modality, providing a more uniform evaluation setting.

This paper synthetizes the 2013 evaluation campaign and introduces the results provided in the papers of the participants. The remainder of the paper is organised as follows. In Section 2, we present the overall evaluation methodology that has been used. Sections 3-10 discuss the settings and the results of each of the test cases. Section 11 overviews lessons learned from the campaign. Finally, Section 12 concludes the paper.

2 General methodology

We first present the test cases proposed this year to the OAEI participants (§2.1). Then, we discuss the resources used by participants to test their systems and the execution environment used for running the tools (§2.2). Next, we describe the steps of the OAEI campaign (§2.3-2.5) and report on the general execution of the campaign (§2.6).

2.1 Tracks and test cases

This year's campaign consisted of 6 tracks gathering 8 test cases and different evaluation modalities:

[1] http://oaei.ontologymatching.org
[2] http://om2013.ontologymatching.org
[3] http://www.seals-project.eu

The benchmark track (§3): Like in previous campaigns, a systematic benchmark series has been proposed. The goal of this benchmark series is to identify the areas in which each matching algorithm is strong or weak by systematically altering an ontology. This year, we generated a new benchmark based on the original bibliographic ontology.

The expressive ontology track offers real world ontologies using OWL modelling capabilities:

Anatomy (§4): The anatomy real world test case is about matching the Adult Mouse Anatomy (2744 classes) and a small fragment of the NCI Thesaurus (3304 classes) describing the human anatomy.

Conference (§5): The goal of the conference test case is to find all correct correspondences within a collection of ontologies describing the domain of organizing conferences. Results were evaluated automatically against reference alignments and by using logical reasoning techniques.

Large biomedical ontologies (§6): The Largebio test case aims at finding alignments between large and semantically rich biomedical ontologies such as FMA, SNOMED-CT, and NCI. The UMLS Metathesaurus has been used as the basis for reference alignments.

Multilingual

Multifarm(§7): This test case is based on a subset of the Conference data set, translated into eight different languages (Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish) and the corresponding alignments between these ontologies. Results are evaluated against these alignments.

Directories and thesauri

Library(§8): The library test case is a real-word task to match two thesauri. The goal of this test case is to find whether the matchers can handle such lightweight ontologies including a huge amount of concepts and additional descriptions. Results are evaluated both against a reference alignment and through manual scrutiny.

Interactive matching

Interactive(§9): This test case offers the possibility to compare different interactive matching tools which require user interaction. Its goal is to show if user interaction can improve matching results, which methods are most promising and how many interactions are necessary. All participating systems are evaluated on the conference data set using an oracle based on the reference alignment.

Instance matching (§10): The goal of the instance matching track is to evaluate the performance of different tools on the task of matching RDF individuals which originate from different sources but describe the same real-world entity. Both the training data set and the evaluation data set were generated by exploiting the same configuration of the RDFT transformation tool. It performs controlled alterations of an initial data source generating data sets and reference links (i.e. alignments). Reference links were provided for the training set but not for the evaluation set, so the evaluation is blind.

Table 1 summarizes the variation in the proposed test cases.

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, DE, EN, ES, FR, NL, RU, PT	√
library	OWL	=	[0 1]	open	EN, DE	√
interactive	OWL-DL	=, <=	[0 1]	open	EN	√
im-rdft	RDF	=	[0 1]	blind	EN	

Table 1. Characteristics of the test cases (open evaluation is made with already published reference alignments and blind evaluation is made by organizers from reference alignments unknown to the participants).

2.2 The SEALS platform

In 2010, participants of the Benchmark, Anatomy and Conference test cases were asked for the first time to use the SEALS evaluation services: they had to wrap their tools as web services and the tools were executed on the machines of the developers [29]. Since 2011, tool developers had to implement a simple interface and to wrap their tools in a predefined way including all required libraries and resources. A tutorial for tool wrapping was provided to the participants. It describes how to wrap a tool and how to use a simple client to run a full evaluation locally. After local tests are passed successfully, the wrapped tool had to be uploaded on the SEALS portal[4]. Consequently, the evaluation was executed by the organizers with the help of the SEALS infrastructure. This approach allowed to measure runtime and ensured the reproducibility of the results. As a side effect, this approach also ensures that a tool is executed with the same settings for all of the test cases that were executed in the SEALS mode.

2.3 Preparatory phase

Ontologies to be matched and (where applicable) reference alignments have been provided in advance during the period between June 15^{th} and July 3^{rd}, 2013. This gave potential participants the occasion to send observations, bug corrections, remarks and other test cases to the organizers. The goal of this preparatory period is to ensure that the delivered tests make sense to the participants. The final test base was released on July 3^{rd}, 2013. The data sets did not evolve after that.

2.4 Execution phase

During the execution phase, participants used their systems to automatically match the test case ontologies. In most cases, ontologies are described in OWL-DL and serialized in the RDF/XML format [6]. Participants can self-evaluate their results either by comparing their output with reference alignments or by using the SEALS client to compute

[4] `http://www.seals-project.eu/join-the-community/`

precision and recall. They can tune their systems with respect to the non blind evaluation as long as the rules published on the OAEI web site are satisfied. This phase has been conducted between July 3^{rd} and September 1^{st}, 2013.

2.5 Evaluation phase

Participants have been encouraged to provide (preliminary) results or to upload their wrapped tools on the SEALS portal by September 1^{st}, 2013. For the SEALS modality, a full-fledged test including all submitted tools has been conducted by the organizers and minor problems were reported to some tool developers, who had the occasion to fix their tools and resubmit them.

First results were available by September 23^{rd}, 2013. The organizers provided these results individually to the participants. The results were published on the respective web pages by the organizers by October 1^{st}. The standard evaluation measures are usually precision and recall computed against the reference alignments. More details on evaluation measures are given in each test case section.

2.6 Comments on the execution

The number of participating systems has regularly increased over the years: 4 participants in 2004, 7 in 2005, 10 in 2006, 17 in 2007, 13 in 2008, 16 in 2009, 15 in 2010, 18 in 2011, 21 in 2012, 23 in 2013. However, participating systems are now constantly changing. In 2013, 11 (7 in 2012) systems have not participated in any of the previous campaigns. The list of participants is summarized in Table 2. Note that some systems were also evaluated with different versions and configurations as requested by developers (see test case sections for details).

System	AML	CIDER-CL	CroMatcher	HerTUDA	HotMatch	IAMA	LilyIOM	LogMap	LogMapLite	MaasMtch	MapSSS	ODGOMS	OntoK	RiMOM2013	ServOMap	SLINT++	SPHeRe	StringsAuto	Synthesis	WeSeE	WikiMatch	XMap	YAM++	Total=23
Confidence	√	√	√				√	√		√		√	√	√	√	√				√		√	√	14
benchmarks	√	√	√	√	√	√		√	√	√	√	√	√	√	√			√	√	√	√	√	√	20
anatomy	√	√		√	√	√		√	√	√	√	√			√			√		√	√	√	√	17
conference	√	√	√	√	√	√		√	√	√	√	√	√	√	√			√	√	√	√	√	√	20
multifarm	√	√		√	√	√		√	√	√	√	√	√	√				√	√	√	√		√	17
library	√			√	√	√		√	√			√			√			√				√	√	11
interactive	√			√				√												√				4
large bio	√			√	√	√		√	√	√		√			√		√	√				√	√	13
im-rdft							√	√						√		√								4
total	7	4	2	7	6	6	1	8	6	5	4	6	3	4	5	1	1	6	3	5	4	5	6	106

Table 2. Participants and the state of their submissions. Confidence stands for the type of results returned by a system: it is ticked when the confidence is a non boolean value.

Only four systems participated in the instance matching track, where two of them (LogMap and RiMOM2013) also participated in the SEALS tracks. The interactive track also had the same participation, since there are not yet many tools supporting user intervention within the matching process. Finally, some systems were not able to pass some test cases as indicated in Table 2. SPHeRe is an exception since it only participated in the largebio test case. It is a special system based on cloud computing which did not use the SEALS interface this year.

The result summary per test case is presented in the following sections.

3 Benchmark

The goal of the benchmark data set is to provide a stable and detailed picture of each algorithm. For that purpose, algorithms are run on systematically generated test cases.

3.1 Test data

The systematic benchmark test set is built around a seed ontology and many variations of it. Variations are artificially generated, and focus on the characterization of the behavior of the tools rather than having them compete on real-life problems.

Since OAEI 2011.5, they are obtained by discarding and modifying features from a seed ontology. Considered features are names of entities, comments, the specialization hierarchy, instances, properties and classes. Full description of the systematic benchmark test set can be found on the OAEI web site.

This year, we used a version of the benchmark test suite generated by the test generator described in [13] from the usual bibliography ontology. The biblio seed ontology concerns bibliographic references and is inspired freely from BibTeX. It contains 33 named classes, 24 object properties, 40 data properties, 56 named individuals and 20 anonymous individuals. The test case was not available to participants: participants could test their systems with respect to last year data sets, but they have been evaluated against a newly generated test. The tests were also blind for the organizers since we did not looked into them before running the systems.

We also generated and run another test suite from a different seed ontology, but we decided to cancel the evaluation because, due to the particular nature of the seed ontology, the generator was not able to properly discard important information. We did not run scalability tests this year.

The reference alignments are still restricted to named classes and properties and use the "=" relation with confidence of 1.

3.2 Results

We run the experiments on a Debian Linux virtual machine configured with four processors and 8GB of RAM running under a Dell PowerEdge T610 with 2*Intel Xeon Quad Core 2.26GHz E5607 processors and 32GB of RAM, under Linux ProxMox 2 (Debian). All matchers where run under the SEALS client using Java 1.7 and a maximum heap size of 6GB. No timeout was explicitly set.

Reported figures are the average of 5 runs. As has already been shown in [13], there is not much variance in compliance measures across runs. This is not necessarily the case for time measurements so we report standard deviations with time measurements.

From the 23 systems listed in Table 2, 20 systems participated in this test case. Three systems were only participating in the instance matching or largebio test cases. XMap had two different system versions.

A few of these systems encountered problems (marked * in the results table): LogMap and OntoK had quite random problems and did not return results for some tests sometimes; ServOMap did not return results for tests past #261-4; MaasMatch did not return results for tests past #254; MapSSS and StringsAuto did not return results for tests past #202 or #247. Besides the two last systems, the problems where persistent across all 5 runs. MapSSS and StringsAuto alternated between the two failure patterns. We suspect that some of the random problems are due to internal or network timeouts.

Compliance Concerning F-measure results, YAM++ (.89) and CroMatcher (.88) are far ahead before Cider-CL (.75), IAMA (.73) and ODGOMS (.71). Without surprise, such systems have all the same profile: their precision is higher than their recall.

With respect to 2012, some systems maintained their performances or slightly improved them (YAM++, MaasMatch, Hertuda, HotMatch, WikiMatch) while other showed severe degradations. Some of these are explained by failures (MapSSS, ServOMap, LogMap) some others are not explained (LogMapLite, WeSeE). Matchers with lower performance than the baseline are those mentioned before as encountering problems when running tests. This is a problem that such matchers are not robust to these classical tests. It is noteworthy, and surprising, that most of the systems which did not complete all the tests were systems which completed them in 2012!

Confidence accuracy Confidence-weighted measures reward systems able to provide accurate confidence values. Using confidence-weighted F-measures does not increase the evaluation of systems (beside edna which does not perform any filtering). In principle, the weighted recall cannot be higher, but the weighted precision can. In fact, only edna, OntoK and XMapSig have an increased precision. The order given above does not change much with the weighted measures: IAMA and ODGOMS pass CroMatcher and Cider-CL. The only system to suffer a dramatic decrease is RiMOM, owing to the very low confidence measures that it provides.

For those systems which have provided their results with confidence measures different from 1 or 0, it is possible to draw precision/recall graphs in order to compare them; these graphs are given in Figure 1. The graphs show the real precision at n% recall and they stop when no more correspondences are available; then the end point corresponds to the precision and recall reported in the Table 3.

The precision-recall curves confirm the good performances of YAM++ and CroMatcher. CroMatcher achieves the same level of recall as YAM++ but with consistently lower precision. The curves show the large variability across systems. This year, systems seem to be less focussed on precision and make progress at the expense of precision. However, this may be an artifact due to systems facing problems.

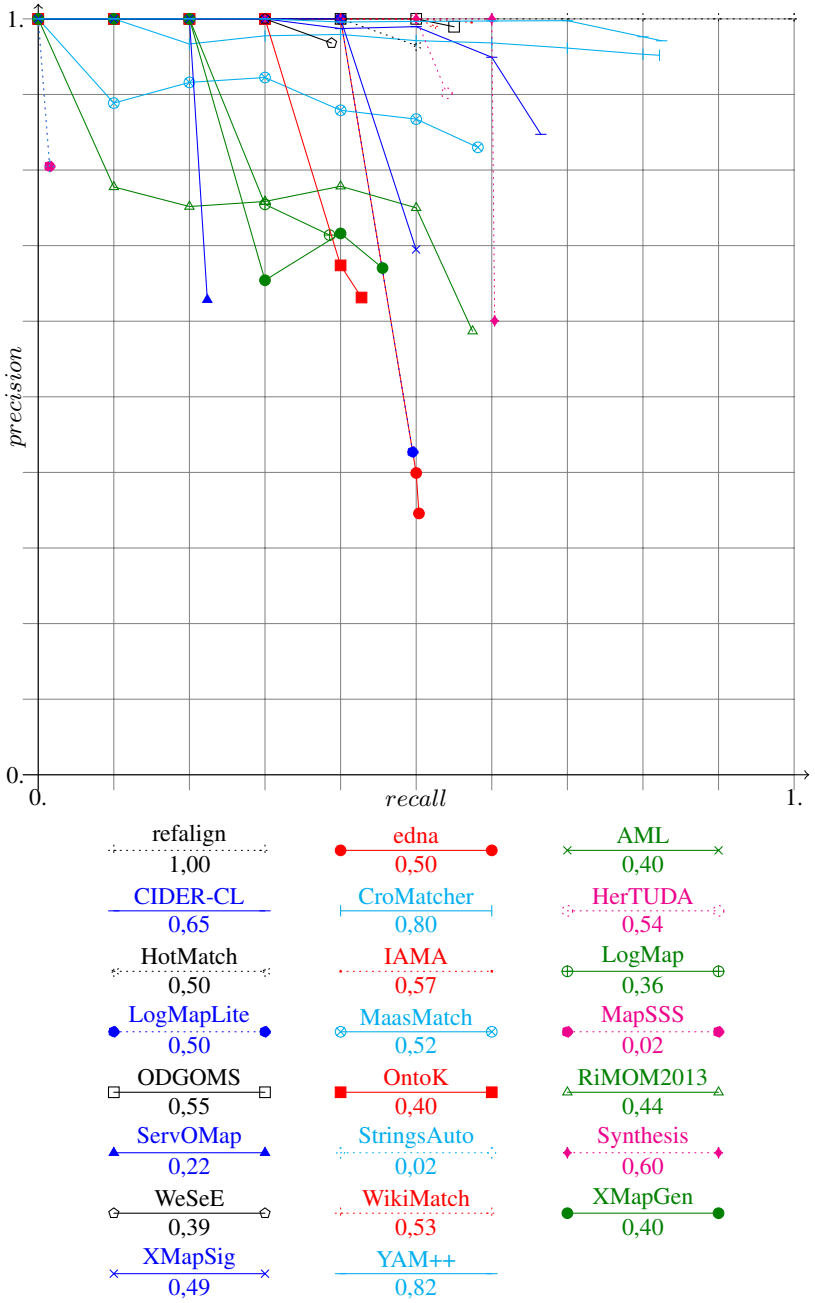

Fig. 1. Precision/recall graphs for benchmarks. The alignments generated by matchers are cut under a threshold necessary for achieving $n\%$ recall and the corresponding precision is computed. Systems for which these graphs are not meaningful (because they did not provide graded confidence values) are drawn in dashed lines.

Runtime There is a large discrepancy between matchers concerning the time spent to match one test run, i.e., 94 matching tests. It ranges from less than a minute for LogMapLite and AML (we do not count StringsAuto which failed to perform many tests) to nearly three hours for OntoK. In fact, OntoK takes as much time as all the other matchers together. Beside these large differences, we also observed large deviations across runs.

We provide (Table 3) the average F-measure point provided per second by matchers. This makes a different ordering of matchers: AML (1.04) comes first before Hertuda (0.94) and LogMapLite (0.81). None of the matchers with the best performances come first. This means that, for achieving good results, considerable time should be spent (however, YAM++ still performs the 94 matching operations in less than 12 minutes).

3.3 Conclusions

Regarding compliance, we observed that, with very few exceptions, the systems performed always better than the baseline. Most of the systems are focussing on precision. This year there was a significant number of systems unable to pass the tests correctly. On the one hand, this is good news: this means that systems are focussing on other test cases than benchmarks. On the other hand, it exhibits system brittleness.

Except for a very few exception, system run time performance is acceptable on tests of that size, but we did not perform scalability tests like last year.

	biblio2 (2012)			biblioc			time		
Matching system	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.	Time(s)	St. Dev.	pt F-m./s
edna	0.46	0.48	0.50	0.35	0.41	0.50			
	(0.61)	(0.55)		(0.58)	(0.54)				
AML				1.00	0.57	0.40	**55**	±6	**1.04**
CIDER-CL				0.85	0.75	0.67	844	±19	0.09
				(0.84)	(0.66)	(0.55)			
CroMatcher				0.95	**0.88**	0.82	1114	±21	0.08
				(0.75)	(0.68)	(0.63)			
Hertuda	0.93	0.67	0.53	0.90	0.68	0.54	72	±6	0.94
Hotmatch	0.99	0.68	0.52	0.96	0.68	0.50	103	±6	0.66
IAMA				0.99	0.73	0.57	102	±10	0.72
LogMap	1.00	0.64	0.47	0.72	0.53	0.42	123	±7	0.43
		(0.59)	(0.42)		(0.51)	(0.39)			
LogMapLt	0.95	0.66	0.50	0.43	0.46	0.50	57	±7	0.81
MaasMtch (*)	0.6	0.6	0.6	0.84	0.69	0.59	173	±6	0.40
	(0.93)	(0.65)	(0.5)	(0.66)	(0.50)	(0.41)			
MapSSS (*)	1.00	**0.86**	0.75	0.84	0.14	0.08	81	±44	0.17
ODGOMS				0.99	0.71	0.55	100	±6	0.71
				(0.98)	(0.70)	(0.54)			
OntoK				0.63	0.51	0.43	10241	±347	0.00
				(0.69)		(0.40)			
RiMOM2013				0.59	0.58	0.58	105	±34	0.55
				(0.49)	(0.19)	(0.12)			
ServOMap (*)	1.00	0.67	0.5	0.53	0.33	0.22	409	±33	0.08
StringsAuto (*)				0.84	0.14	0.08	56	±38	0.25
Synthesis				0.60	0.60	0.60	659	±11	0.09
WeSeE	1.00	0.69(0.68)	0.52	0.96	0.55	0.39	4933	±40	0.01
Wikimatch	0.97	0.67(0.68)	0.52	0.99	0.69	0.53	1845	±39	0.04
XMapGen				0.66	0.54	0.46	594	±5	0.09
					(0.52)	(0.44)			
XMapSig				0.70	0.58	0.50	612	±11	0.09
				(0.71)	(0.59)				
YAM++	0.96	**0.89**	0.82	0.97	**0.89**	0.82	702	±46	0.13
	(1.00)	(0.72)	(0.56)	(0.84)	(0.77)	(0.70)			

Table 3. Results obtained by participants on the biblio benchmark test suite aggregated with harmonic means (values within parentheses are *weighted* version of the measure reported only when different).

4 Anatomy

The anatomy test case confronts matchers with a specific type of ontologies from the biomedical domain. We focus on two fragments of biomedical ontologies which describe the human anatomy[5] and the anatomy of the mouse[6]. This data set has been used since 2007 with some improvements over the years.

4.1 Experimental setting

We conducted experiments by executing each system in its standard setting and we compare precision, recall, F-measure and recall+. The measure recall+ indicates the amount of detected non-trivial correspondences. The matched entities in a non-trivial correspondence do not have the same normalized label. The approach that generates only trivial correspondences is depicted as baseline *StringEquiv* in the following section.

This year we run the systems on a server with 3.46 GHz (6 cores) and 8GB RAM allocated to the matching systems. This is a different setting compared to previous years, so, runtime results are not fully comparable across years. The evaluation was performed with the SEALS client. However, we slightly changed the way how precision and recall are computed, i.e., the results generated by the SEALS client vary in some cases by 0.5% compared to the results presented below. In particular, we removed trivial correspondences in the `oboInOwl` namespace like

```
http://...oboInOwl#Synonym = http://...oboInOwl#Synonym
```

as well as correspondences expressing relations different from equivalence. Using the Pellet reasoner we also checked whether the generated alignment is coherent, i.e., there are no unsatisfiable concepts when the ontologies are merged with the alignment.

4.2 Results

In Table 4, we analyze all participating systems that could generate an alignment in less than ten hours. The listing comprises of 20 entries sorted by F-measure. Four systems participated each with two different versions. These are AML and GOMMA with versions which use background knowledge (indicated with suffix "-bk"), LogMap with a lightweight version LogMapLite that uses only some core components and XMap with versions XMapSig and XMapGen which use two different parameters. For comparison purposes, we run again last year version of GOMMA. GOMMA and HerTUDA participated with the same system as last year (indicated by * in the table). In addition to these two tools we have eight more systems which participated in 2012 and now participated with new versions (HotMatch, LogMap, MaasMatch, MapSSS, ServOMap, WeSeE, WikiMatch and YAM++). Due to some software and hardware incompatibilities,

[5] `http://www.cancer.gov/cancertopics/cancerlibrary/terminologyresources/`

[6] `http://www.informatics.jax.org/searches/AMA_form.shtml`

YAM++ had to be run on a different machine and therefore its runtime (indicated by **) is not fully comparable to that of other systems. Thus, 20 different systems generated an alignment within the given time frame. Four participants (CroMatcher, RiMOM2013, OntoK and Synthesis) did not finish in time or threw an exception.

Matcher	Runtime	Size	Precision	F-measure	Recall	Recall+	Coherent
AML-bk	43	1477	0.95	0.94	0.93	0.82	√
GOMMA-bk*	11	1534	0.92	0.92	0.93	0.81	-
YAM++	62**	1395	0.94	0.90	0.87	0.66	-
AML	15	1315	0.95	0.89	0.83	0.54	√
LogMap	13	1398	0.92	0.88	0.85	0.59	√
GOMMA*	9	1264	0.96	0.87	0.80	0.47	-
StringsAuto	1444	1314	0.90	0.83	0.78	0.43	-
LogMapLite	7	1148	0.96	0.83	0.73	0.29	-
MapSSS	2040	1296	0.90	0.83	0.77	0.44	-
ODGOMS	1212	1102	0.98	0.82	0.71	0.24	-
WikiMatch	19366	1027	0.99	0.80	0.67	0.15	-
HotMatch	300	989	0.98	0.77	0.64	0.14	-
StringEquiv	-	946	*1.00*	0.77	0.62	0.00	-
XMapSig	393	1192	0.86	0.75	0.67	0.13	-
ServOMap	43	975	0.96	0.75	0.62	0.10	-
XMapGen	403	1304	0.81	0.75	0.69	0.19	-
IAMA	10	845	*1.00*	0.71	0.55	0.01	-
CIDER-CL	12308	1711	0.65	0.69	0.73	0.31	-
HerTUDA*	117	1479	0.69	0.68	0.67	0.15	-
WeSeE	34343	935	0.62	0.47	0.38	0.09	-
MaasMatch	8532	2011	0.36	0.41	0.48	0.23	-

Table 4. Comparison, ordered by F-measure, against the reference alignment, runtime is measured in seconds, the "size" column refers to the number of correspondences in the generated alignment.

Nine systems finished in less than 100 seconds, compared to 8 systems in OAEI 2012 and 2 systems in OAEI 2011. This year, 20 out of 24 systems generated results compared to last year when 14 out of 18 systems generated results within the given time frame. The top systems in terms of runtimes are LogMap, GOMMA, IAMA and AML. Depending on the specific version of the systems, they require between 7 and 15 seconds to match the ontologies. The table shows that there is no correlation between quality of the generated alignment in terms of precision and recall and required runtime. This result has also been observed in previous OAEI campaigns.

Table 4 also shows the results for precision, recall and F-measure. In terms of F-measure, the two top ranked systems are AML-bk and GOMMA-bk. These systems use specialised background knowledge, i.e., they are based on mapping composition techniques and the reuse of mappings between UMLS, Uberon and FMA. AML-bk and GOMMA-bk are followed by a group of matching systems (YAM++, AML, LogMap, GOMMA) generating alignments that are very similar with respect to precision, recall and F-measure (between 0.87 and 0.91 F-measure). LogMap uses the general (biomedical) purpose UMLS Lexicon, while the other systems either use Wordnet or no back-

ground knowledge. The results of these systems are at least as good as the results of the best system in OAEI 2007-2010. Only AgreementMaker using additional background knowledge could generate better results than these systems in 2011.

This year, 8 out of 20 systems achieved an F-measure that is lower than the baseline which is based on (normalized) string equivalence (StringEquiv in the table).

Moreover, nearly all systems find many non-trivial correspondences. An exception are IAMA and WeSeE which generated an alignment that is quite similar to the alignment generated by the baseline approach.

From the systems which participated last year WikiMatch showed a considerable improvement. It increased precision from 0.86 to 0.99 and F-measure from 0.76 to 0.80. The other systems produced very similar results compared to the previous year. One exception is WeSeE which achieved a much lower F-measure than in 2012.

Three systems have produced an alignment which is coherent. Last year two systems produced such alignments.

4.3 Conclusions

This year 24 systems (or system variants) participated in the anatomy test case out of which 20 produced results within 10 hours. This is so far the highest number of participating systems as well as the highest number of systems which produce results given time constraints for the anatomy test case.

As last year, we have witnessed a positive trend in runtimes as the majority of systems finish execution in less than one hour (16 out of 20). The AML-bk system improves the best result in terms of F-measure set by a previous version of the system in 2010 and makes it also the top result for the anatomy test case.

5 Conference

The conference test case introduces matching several moderately expressive ontologies. Within this test case, participant results were evaluated against reference alignments (containing merely equivalence correspondences) and by using logical reasoning. The evaluation has been performed with the SEALS infrastructure.

5.1 Test data

The data set consists of 16 ontologies in the domain of organizing conferences. These ontologies have been developed within the OntoFarm project[7].

The main features of this test case are:

– *Generally understandable domain.* Most ontology engineers are familiar with organizing conferences. Therefore, they can create their own ontologies as well as evaluate the alignments among their concepts with enough erudition.

[7] http://nb.vse.cz/~svatek/ontofarm.html

– *Independence of ontologies.* Ontologies were developed independently and based on different resources, they thus capture the issues in organizing conferences from different points of view and with different terminologies.
– *Relative richness in axioms.* Most ontologies were equipped with OWL DL axioms of various kinds; this opens a way to use semantic matchers.

Ontologies differ in their numbers of classes, of properties, in expressivity, but also in underlying resources.

5.2 Results

We provide results in terms of $F_{0.5}$-measure, F_1-measure and F_2-measure, comparison with baseline matchers, precision/recall triangular graph and coherency evaluation.

Evaluation based on reference alignments We evaluated the results of participants against blind reference alignments (labelled as *ra2* on the conference web-page). This includes all pairwise combinations between 7 different ontologies, i.e. 21 alignments.

These reference alignments have been generated as a transitive closure computed on the original reference alignments. In order to obtain a coherent result, conflicting correspondences, i.e., those causing unsatisfiability, have been manually inspected and removed by evaluators. As a result, the degree of correctness and completeness of the new reference alignment is probably slightly better than for the old one. However, the differences are relatively limited. Whereas the new reference alignments are not open, the old reference alignments (labeled as *ra1* on the conference web-page) are available. These represent close approximations of the new ones.

Table 5 shows the results of all participants with regard to the new reference alignment. $F_{0.5}$-measure, F_1-measure and F_2-measure are computed for the threshold that provides the highest average F_1-measure. F_1 is the harmonic mean of precision and recall where both are equally weighted; F_2 weights recall higher than precision and $F_{0.5}$ weights precision higher than recall. The matchers shown in the table are ordered according to their highest average F_1-measure. This year we employed two baselines matcher. *edna* (string edit distance matcher) is used within the benchmark test case and with regard to performance it is very similar as previously used *baseline2*; *StringEquiv* is used within the anatomy test case. These baselines divide matchers into three groups. Group 1 consists of matchers (YAM++, AML-bk –AML standing for AgreementMakerLight–, LogMap, AML, ODGOMS, StringsAuto, ServOMap, MapSSS, HerTUDA, WikiMatch, WeSeE-Match, IAMA, HotMatch, CIDER-CL) having better (or the same) results than both baselines in terms of highest average F_1-measure. Group 2 consists of matchers (OntoK, LogMapLite, XMapSigG, XMapGen and SYNTHESIS) performing better than baseline *StringEquiv* but worse than *edna*. Other matchers (RIMOM2013, CroMatcher and MaasMatch) performed worse than both baselines. CroMatcher was unable to process any ontology pair where conference.owl ontology was included. Therefore, the evaluation was run only on 15 test cases. Thus, its results are just an approximation.

Performance of matchers from Group 1 regarding F_1-measure is visualized in Figure 2.

Matcher	Prec.	$F_{0.5}$-m.	F_1-m.	F_2-m.	Rec.	Size	Inc. Al.	Inc-dg
YAM++	0.78	0.75	0.71	0.67	0.65	12.524	0	0.0%
AML-bk	0.82	0.74	0.64	0.57	0.53	9.714	0	0.0%
LogMap	0.76	0.70	0.63	0.57	0.54	10.714	0	0.0%
AML	0.82	0.73	0.63	0.55	0.51	9.333	0	0.0%
ODGOMS1_2	0.70	0.66	0.62	0.57	0.55	11.762	13	6.5%
StringsAuto	0.74	0.68	0.60	0.53	0.50	11.048	0	0.0%
ServOMap_v104	0.69	0.64	0.58	0.53	0.50	11.048	4	2.0%
MapSSS	0.77	0.68	0.58	0.50	0.46	9.857	0	0.0%
ODGOMS1_1	0.72	0.65	0.57	0.51	0.47	9.667	9	4.4%
HerTUDA	0.70	0.63	0.56	0.49	0.46	9.857	9	5.3%
WikiMatch	0.70	0.63	0.55	0.48	0.45	9.762	10	6.1%
WeSeE-Match	0.79	0.67	0.55	0.46	0.42	8	1	0.4%
IAMA	0.74	0.65	0.55	0.48	0.44	8.857	7	4%
HotMatch	0.67	0.62	0.55	0.50	0.47	10.524	9	5.0%
CIDER-CL	0.72	0.64	0.55	0.48	0.44	22.857	21	19.5%
edna	*0.73*	*0.64*	*0.55*	*0.48*	*0.44*			
OntoK	0.72	0.63	0.54	0.47	0.43	8.952	7	3.9%
LogMapLite	0.68	0.62	0.54	0.48	0.45	9.952	7	5.4%
XMapSiG1_3	0.68	0.61	0.53	0.47	0.44	10.714	0	0.0%
XMapGen1_4	0.64	0.59	0.53	0.48	0.45	18.571	0	0.0%
SYNTHESIS	0.73	0.63	0.53	0.45	0.41	8.429	9	4.8%
StringEquiv	*0.76*	*0.64*	*0.52*	*0.43*	*0.39*			
RIMOM2013*	0.55	0.53	0.51	0.48	0.47	56.81	20	27.1%
XMapSiG1_4	0.75	0.62	0.50	0.41	0.37	8.048	1	0.8%
CroMatcher	0.56	0.53	0.49	0.45	0.43			
XMapGen	0.70	0.59	0.48	0.40	0.36	12	1	0.4%
MaasMatch	0.27	0.30	0.36	0.44	0.53			

Table 5. The highest average $F_{[0.5|1|2]}$-measure and their corresponding precision and recall for each matcher with its F_1-optimal threshold (ordered by F_1-measure). Average size of alignments, number of incoherent alignments and average degree of incoherence. The mark * is added when we only provide lower bound of the degree of incoherence due to the combinatorial complexity of the problem.

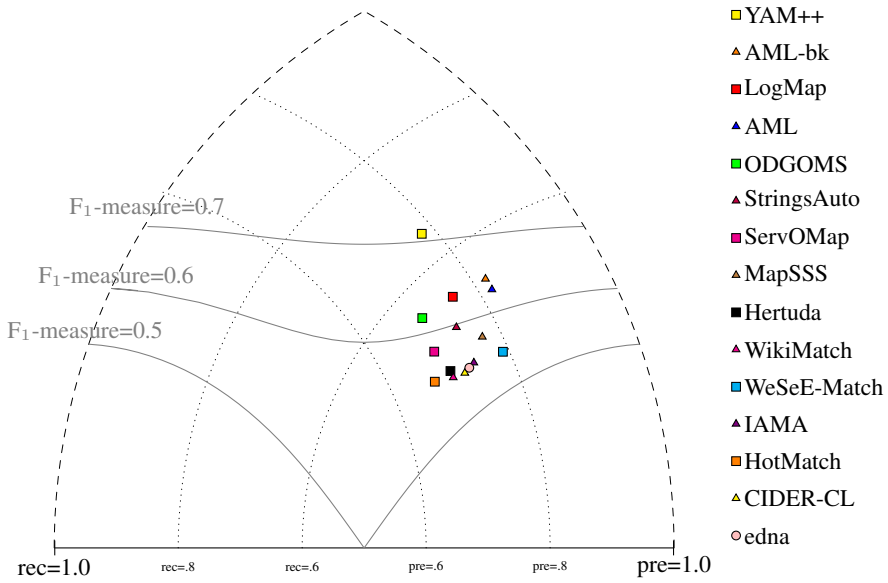

Fig. 2. Precision/recall triangular graph for the conference test case. Dotted lines depict level of precision/recall while values of F_1-measure are depicted by areas bordered by corresponding lines F_1-measure=0.[5|6|7].

Comparison with previous years Ten matchers also participated in this test case in OAEI 2012. The largest improvement was achieved by MapSSS (precision from .47 to .77, whereas recall remains the same, .46) and ServOMap (precision from .68 to .69 and recall from .41 to .50).

Runtimes We measured the total time of generating 21 alignments. It was executed on a laptop under Ubuntu running on Intel Core i5, 2.67GHz and 8GB RAM. In all, there are eleven matchers which finished all 21 tests within 1 minute or around 1 minute (AML-bk: 16s, ODGOMS: 19s, LogMapLite: 21s, AML, HerTUDA, StringsAuto, HotMatch, LogMap, IAMA, RIMOM2013: 53s and MaasMatch: 76s). Next, four systems needed less than 10 minutes (ServOMap, MapSSS, SYNTHESIS, CIDER-CL). 10 minutes are enough for the next three matchers (YAM++, XMapGen, XMapSiG). Finally, three matchers needed up to 40 minutes to finish all 21 test cases (WeSeE-Match: 19 min, WikiMatch: 26 min, OntoK: 40 min).

In conclusion, regarding performance we can see (clearly from Figure 2) that YAM++ is on the top again. The next four matchers (AML-bk, LogMap, AML, ODGOMS) are relatively close to each other. This year there is a larger group of matchers (15) which are above the edna *baseline* than previous years. This is partly because a couple of previous system matchers improved and a couple of high quality new system matchers entered the OAEI campaign.

Evaluation based on alignment coherence As in the previous years, we apply the Maximum Cardinality measure to evaluate the degree of alignment incoherence, see

Table 5. Details on this measure and its implementation can be found in [21]. We computed the average for all 21 test cases of the conference test case for which there exists a reference alignment. In one case, RIMOM2013, marked with an asterisk, we could not compute the exact degree of incoherence due to the combinatorial complexity of the problem, however we were still able to compute a lower bound for which we know that the actual degree is higher. We do not provide numbers for CroMatcher since it did not generate all 21 alignments. For MaasMatch, we could not compute the degree of incoherence since its alignments are highly incoherent (and thus the reasoner encountered exceptions).

This year eight systems managed to generate coherent alignments: AML, AML-bk, LogMap, MapSSS, StringsAuto, XMapGen, XMapSiG and YAM++. Coherent results need not only be related to a specific approach ensuring the coherency, but it can be indirectly caused by generating small and highly precise alignments. However, looking at Table 5 it seems that there is no matcher which on average generate too small alignments. In all, this is a large important improvement compared to previous years, where we observed that only four (two) systems managed to generate (nearly) coherent alignments in 2011-2012.

6 Large biomedical ontologies (largebio)

The Largebio test case aims at finding alignments between the large and semantically rich biomedical ontologies FMA, SNOMED-CT, and NCI, which contains 78,989, 306,591 and 66,724 classes, respectively.

6.1 Test data

The test case has been split into three matching problems: FMA-NCI, FMA-SNOMED and SNOMED-NCI; and each matching problem in 2 tasks involving different fragments of the input ontologies.

The UMLS Metathesaurus [4] has been selected as the basis for reference alignments. UMLS is currently the most comprehensive effort for integrating independently-developed medical thesauri and ontologies, including FMA, SNOMED-CT, and NCI. Although the standard UMLS distribution does not directly provide "alignments" (in the OAEI sense) between the integrated ontologies, it is relatively straightforward to extract them from the information provided in the distribution files (see [18] for details).

It has been noticed, however, that although the creation of UMLS alignments combines expert assessment and auditing protocols they lead to a significant number of logical inconsistencies when integrated with the corresponding source ontologies [18].

To address this problem, in OAEI 2013, unlike previous editions, we have created a unique refinement of the UMLS mappings combining both Alcomo (mapping) debugging system [21] and LogMap's (mapping) repair facility [17], and manual curation when necessary. This refinement of the UMLS mappings, which does not lead to unsatisfiable classes[8], has been used as the Large BioMed reference alignment. Objections

[8] For the SNOMED-NCI case we used the OWL 2 EL reasoner ELK, see Section 6.4 for details.

have been raised on the validity (and fairness) of the application of mapping repair techniques to make reference alignments coherent [24]. For next year campaign, we intend to take into consideration their suggestions to mitigate the effect of using repair techniques. This year reference alignment already aimed at mitigating the fairness effect by combining two mapping repair techniques, however further improvement should be done in this line.

6.2 Evaluation setting, participation and success

We have run the evaluation in a high performance server with 16 CPUs and allocating 15 Gb RAM. Precision, Recall and F-measure have been computed with respect to the UMLS-based reference alignment. Systems have been ordered in terms of F-measure.

In the largebio test case, 13 out of 21 participating systems have been able to cope with at least one of the tasks of the largebio test case. Synthesis, WeSeEMatch and WikiMatch failed to complete the smallest task with a time out of 18 hours, while MapSSS, RiMOM, CIDER-CL, CroMatcher and OntoK threw an exception during the matching process. The latter two threw an out-of-memory exception. In total we have evaluated 20 system configurations.

6.3 Tool variants and background knowledge

There were, in this test case, different variants of tools using background knowledge to certain degree. These are:

- XMap participates with two variants. XMapSig, which uses a sigmoid function, and XMapGen, which implements a genetic algorithm. ODGOMS also participates with two versions (v1.1 and v1.2). ODGOMS-v1.1 is the original submitted version while ODGOMS-v1.2 includes some bug fixes and extensions.
- LogMap has also been evaluated with two variants: LogMap and LogMap-BK. LogMap-BK uses normalisations and spelling variants from the general (biomedical) purpose UMLS Lexicon[9] while LogMap has this feature deactivated.
- AML has been evaluated with 6 different variants depending on the use of repair techniques (R), general background knowledge (BK) and specialised background knowledge based on the UMLS Metathesaurus (SBK).
- YAM++ and MaasMatch also use the general purpose background knowledge provided by WordNet[10].

Since the reference alignment of this test case is based on the UMLS Metathesaurus, we did not included within the results the alignments provided by AML-SBK and AML-SBK-R. Nevertheless we consider their results very interesting: AML-SBK and AML-SBK-R averaged F-measures higher than 0.90 in all 6 tasks.

We have also re-run the OAEI 2012 version of GOMMA. The results of GOMMA may slightly vary w.r.t. those in 2012 since we have used a different reference alignment.

[9] http://www.nlm.nih.gov/pubs/factsheets/umlslex.html
[10] http://wordnet.princeton.edu/

System	FMA-NCI		FMA-SNOMED		SNOMED-NCI		Average	#
	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6		
LogMapLt	7	59	14	101	54	132	61	6
IAMA	13	139	27	217	98	206	117	6
AML	16	201	60	542	291	569	280	6
AML-BK	38	201	93	530	380	571	302	6
AML-R	18	194	86	554	328	639	303	6
GOMMA$_{2012}$	39	243	54	634	220	727	320	6
AML-BK-R	42	204	121	583	397	635	330	6
YAM++	93	365	100	401	391	712	344	6
LogMap-BK	44	172	85	556	444	1,087	398	6
LogMap	41	161	78	536	433	1,232	414	6
ServOMap	140	2,690	391	4,059	1,698	6,320	2,550	6
SPHeRe (*)	*16*	*8,136*	*154*	*20,664*	*2,486*	*10,584*	7,007	6
XMapSiG	1,476	-	11,720	-	-	-	6,598	2
XMapGen	1,504	-	12,127	-	-	-	6,816	2
Hertuda	3,403	-	17,610	-	-	-	10,507	2
ODGOMS-v1.1	6,366	-	27,450	-	-	-	16,908	2
HotMatch	4,372	-	32,243	-	-	-	18,308	2
ODGOMS-v1.2	10,204	-	42,908	-	-	-	26,556	2
StringsAuto	6,358	-	-	-	-	-	6,358	1
MaasMatch	12,409	-	-	-	-	-	12,409	1
# Systems	**20**	**12**	**18**	**12**	**12**	**12**	**5,906**	**86**

Table 6. System runtimes (s) and task completion. GOMMA is a system provided in 2012. (*) SPHeRe times were reported by the authors. SPHeRe is a special tool which relies on the utilization of cloud computing resources.

6.4 Alignment coherence

Together with Precision, Recall, F-measure and Runtimes we have also evaluated the coherence of alignments. We report (1) the number of unsatisfiabilities when reasoning with the input ontologies together with the computed mappings, and (2) the ratio of unsatisfiable classes with respect to the size of the union of the input ontologies.

We have used the OWL 2 reasoner MORe [2] to compute the number of unsatisfiable classes. For the cases in which MORe could not cope with the input ontologies and the mappings (in less than 2 hours) we have provided a lower bound on the number of unsatisfiable classes (indicated by $\geq$) using the OWL 2 EL reasoner ELK [20].

In this OAEI edition, only three systems have shown mapping repair facilities, namely: YAM++, AML with (R)epair configuration and LogMap. Tables 7-10 show that even the most precise alignment sets may lead to a huge amount of unsatisfiable classes. This proves the importance of using techniques to assess the coherence of the generated alignments.

6.5 Runtimes and task completion

Table 6 shows which systems (including variants) were able to complete each of the matching tasks in less than 18 hours and the required computation times. Systems have

been ordered with respect to the number of completed tasks and the average time required to complete them. Times are reported in seconds.

The last column reports the number of tasks that a system could complete. For example, 12 system configurations were able to complete all six tasks. The last row shows the number of systems that could finish each of the tasks. The tasks involving SNOMED were also harder with respect to both computation times and the number of systems that completed the tasks.

6.6 Results for the FMA-NCI matching problem

Table 7 summarizes the results for the tasks in the FMA-NCI matching problem. LogMap-BK and YAM++ provided the best results in terms of both Recall and F-measure in Task 1 and Task 2, respectively. IAMA provided the best results in terms of precision, although its recall was below average. Hertuda provided competitive results in terms of recall, but the low precision damaged the final F-measure. On the other hand, StringsAuto, XMapGen and XMapSiG provided a set of alignments with high precision, however, the F-measure was damaged due to the low recall of their alignments. Overall, the results were very positive and many systems obtained an F-measure higher than 0.80 in the two tasks.

Efficiency in Task 2 has decreased with respect to Task 1. This is mostly due to the fact that larger ontologies also involve more possible candidate alignments and it is harder to keep high precision values without damaging recall, and vice versa.

6.7 Results for the FMA-SNOMED matching problem

Table 8 summarizes the results for the tasks in the FMA-SNOMED matching problem. YAM++ provided the best results in terms of F-measure on both Task 3 and Task 4. YAM++ also provided the best Precision and Recall in Task 3 and Task 4, respectively; while AML-BK provided the best Recall in Task 3 and AML-R the best Precision in Task 4.

Overall, the results were less positive than in the FMA-NCI matching problem and only YAM++ obtained an F-measure greater than 0.80 in the two tasks. Furthermore, 9 systems failed to provide a recall higher than 0.4. Thus, matching FMA against SNOMED represents a significant leap in complexity with respect to the FMA-NCI matching problem.

As in the FMA-NCI matching problem, efficiency also decreases as the ontology size increases. The most important variations were suffered by SPHeRe, IAMA and GOMMA in terms of precision.

6.8 Results for the SNOMED-NCI matching problem

Table 9 summarizes the results for the tasks in the SNOMED-NCI matching problem. LogMap-BK and ServOMap provided the best results in terms of both Recall and F-measure in Task 5 and Task 6, respectively. YAM++ provided the best results in terms of precision in Task 5 while AML-R in Task 6.

Task 1: small FMA and NCI fragments							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
LogMap-BK	45	2,727	0.95	0.91	0.88	2	0.02%
YAM++	94	2,561	0.98	0.91	0.85	2	0.02%
$GOMMA_{2012}$	40	2,626	0.96	0.91	0.86	2,130	20.9%
AML-BK-R	43	2,619	0.96	0.90	0.86	2	0.02%
AML-BK	39	2,695	0.94	0.90	0.87	2,932	28.8%
LogMap	41	2,619	0.95	0.90	0.85	2	0.02%
AML-R	19	2,506	0.96	0.89	0.82	2	0.02%
ODGOMS-v1.2	10,205	2,558	0.95	0.89	0.83	2,440	24.0%
AML	16	2,581	0.95	0.89	0.83	2,598	25.5%
LogMapLt	8	2,483	0.96	0.88	0.81	2,104	20.7%
ODGOMS-v1.1	6,366	2,456	0.96	0.88	0.81	1,613	15.8%
ServOMap	141	2,512	0.95	0.88	0.81	540	5.3%
SPHeRe	16	2,359	0.96	0.86	0.77	367	3.6%
HotMatch	4,372	2,280	0.96	0.84	0.75	285	2.8%
Average	2,330	2,527	0.90	0.81	0.75	1,582	15.5%
IAMA	14	1,751	0.98	0.73	0.58	166	1.6%
Hertuda	3,404	4,309	0.59	0.70	0.87	2,675	26.3%
StringsAuto	6,359	1,940	0.84	0.67	0.55	1,893	18.6%
XMapGen	1,504	1,687	0.83	0.61	0.48	1,092	10.7%
XMapSiG	1,477	1,564	0.86	0.60	0.46	818	8.0%
MaasMatch	12,410	3,720	0.41	0.46	0.52	9,988	98.1%

Task 2: whole FMA and NCI ontologies							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
YAM++	366	2,759	0.90	0.87	0.85	9	0.01%
$GOMMA_{2012}$	243	2,843	0.86	0.85	0.83	5,574	3.8%
LogMap	162	2,667	0.87	0.83	0.79	10	0.01%
LogMap-BK	173	2,668	0.87	0.83	0.79	9	0.01%
AML-BK	201	2,828	0.82	0.80	0.79	16,120	11.1%
AML-BK-R	205	2,761	0.83	0.80	0.78	10	0.01%
Average	1,064	2,711	0.84	0.80	0.77	9,223	6.3%
AML-R	194	2,368	0.89	0.80	0.72	9	0.01%
AML	202	2,432	0.88	0.80	0.73	1,044	0.7%
SPHeRe	8,136	2,610	0.85	0.80	0.75	1,054	0.7%
ServOMap	2,690	3,235	0.73	0.76	0.80	60,218	41.3%
LogMapLt	60	3,472	0.69	0.74	0.81	26,442	18.2%
IAMA	139	1,894	0.90	0.71	0.58	180	0.1%

Table 7. Results for the FMA-NCI matching problem.

As in the previous matching problems, efficiency decreases as the ontology size increases. For example, in Task 6, only ServOMap and YAM++ could reach an F-measure higher than 0.7. The results were also less positive than in the FMA-SNOMED matching problem, and thus, the SNOMED-NCI case represented another leap in complexity.

Task 3: small FMA and SNOMED fragments							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
YAM++	100	6,635	0.98	0.84	0.73	13,040	55.3%
AML-BK	93	6,937	0.94	0.82	0.73	12,379	52.5%
AML	60	6,822	0.94	0.82	0.72	15,244	64.7%
AML-BK-R	122	6,554	0.95	0.80	0.70	15	0.06%
AML-R	86	6,459	0.95	0.80	0.69	14	0.06%
LogMap-BK	85	6,242	0.96	0.79	0.67	0	0.0%
LogMap	79	6,071	0.97	0.78	0.66	0	0.0%
ServOMap	391	5,828	0.95	0.75	0.62	6,018	25.5%
ODGOMS-v1.2	42,909	5,918	0.86	0.69	0.57	9,176	38.9%
Average	8,073	4,248	0.89	0.55	0.44	7,308	31.0%
GOMMA_{2012}	54	3,666	0.92	0.54	0.38	2,058	8.7%
ODGOMS-v1.1	27,451	2,267	0.88	0.35	0.22	938	4.0%
HotMatch	32,244	2,139	0.87	0.34	0.21	907	3.9%
LogMapLt	15	1,645	0.97	0.30	0.18	773	3.3%
Hertuda	17,610	3,051	0.57	0.29	0.20	1,020	4.3%
SPHeRe	154	1,577	0.92	0.27	0.16	805	3.4%
IAMA	27	1,250	0.96	0.24	0.13	22,925	97.3%
XMapGen	12,127	1,827	0.69	0.24	0.14	23,217	98.5%
XMapSiG	11,720	1,581	0.76	0.23	0.13	23,025	97.7%

Task 4: whole FMA ontology with SNOMED large fragment							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
YAM++	402	6,842	0.95	0.82	0.72	$\geq$57,074	$\geq$28.3%
AML-BK	530	6,186	0.94	0.77	0.65	$\geq$40,162	$\geq$19.9%
AML	542	5,797	0.96	0.76	0.62	$\geq$39,472	$\geq$19.6%
AML-BK-R	584	5,858	0.94	0.74	0.62	29	0.01%
AML-R	554	5,499	0.97	0.74	0.59	7	0.004%
ServOMap	4,059	6,440	0.86	0.72	0.62	$\geq$164,116	$\geq$81.5%
LogMap-BK	556	6,134	0.87	0.71	0.60	0	0.0%
LogMap	537	5,923	0.89	0.71	0.59	0	0.0%
Average	2,448	5,007	0.83	0.59	0.48	40,143	19.9%
GOMMA_{2012}	634	5,648	0.41	0.31	0.26	9,918	4.9%
LogMapLt	101	1,823	0.88	0.30	0.18	$\geq$4,393	$\geq$2.2%
SPHeRe	20,664	2,338	0.61	0.25	0.16	6,523	3.2%
IAMA	218	1,600	0.75	0.23	0.13	$\geq$160,022	$\geq$79.4%

Table 8. Results for the FMA-SNOMED matching problem.

6.9 Summary results for the top systems

Table 10 summarizes the results for the systems that completed all 6 tasks of the Large BioMed Track. The table shows the total time in seconds to complete all tasks and averages for Precision, Recall, F-measure and Incoherence degree. The systems have been ordered according to the average F-measure.

Task 5: small SNOMED and NCI fragments							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
LogMap-BK	444	13,985	0.89	0.77	0.68	≥40	≥0.05%
LogMap	433	13,870	0.90	0.77	0.67	≥47	≥0.06%
ServOMap	1,699	12,716	0.93	0.76	0.64	≥59,944	≥79.8%
AML-BK-R	397	13,006	0.92	0.76	0.65	≥32	≥0.04%
AML-BK	380	13,610	0.89	0.76	0.66	≥66,389	≥88.4%
AML-R	328	12,622	0.92	0.75	0.63	≥36	≥0.05%
YAM++	391	11,672	0.97	0.75	0.61	≥0	≥0.0%
AML	291	13,248	0.89	0.75	0.64	≥63,305	≥84.3%
Average	602	12,003	0.92	0.72	0.60	32,222	42.9%
LogMapLt	55	10,962	0.94	0.70	0.56	≥60,427	≥80.5%
GOMMA_{2012}	221	10,555	0.94	0.68	0.54	≥50,189	≥66.8%
SPHeRe	2,486	9,389	0.92	0.62	0.47	≥46,256	≥61.6%
IAMA	99	8,406	0.96	0.60	0.44	≥40,002	≥53.3%

Task 6: whole NCI ontology with SNOMED large fragment							
System	Time (s)	# Mappings	Scores			Incoherence	
			Prec.	F-m.	Rec.	Unsat.	Degree
ServOMap	6,320	14,312	0.82	0.72	0.64	≥153,259	≥81.0%
YAM++	713	12,600	0.88	0.71	0.60	≥116	≥0.06%
AML-BK	571	11,354	0.92	0.70	0.56	≥121,525	≥64.2%
AML-BK-R	636	11,033	0.93	0.69	0.55	≥41	≥0.02%
LogMap-BK	1,088	12,217	0.87	0.69	0.58	≥1	≥0.001%
LogMap	1,233	11,938	0.88	0.69	0.57	≥1	≥0.001%
AML	570	10,940	0.93	0.69	0.55	≥121,171	≥64.1%
AML-R	640	10,622	0.94	0.68	0.54	≥51	≥0.03%
Average	1,951	11,581	0.88	0.67	0.55	72,365	38.3%
LogMapLt	132	12,907	0.80	0.66	0.56	≥150,773	≥79.7%
GOMMA_{2012}	728	12,440	0.79	0.63	0.53	≥127,846	≥67.6%
SPHeRe	10,584	9,776	0.88	0.61	0.47	≥105,418	≥55.7%
IAMA	207	8,843	0.92	0.59	0.44	≥88,185	≥46.6%

Table 9. Results for the SNOMED-NCI matching problem.

YAM++ was a step ahead and obtained the best average Precision and Recall. AML-R obtained the second best Precision while AML-BK obtained the second best Recall.

Regarding mapping incoherence, LogMap-BK computed, on average, the mapping sets leading to the smallest number of unsatisfiable classes. The configurations of AML using (R)epair also obtained very good results in terms mapping coherence.

Finally, LogMapLt was the fastest system. The rest of the tools, apart from ServoMap and SPHeRe, were also very fast and only needed between 11 and 53 minutes to complete all 6 tasks. ServOMap required around 4 hours to complete them while SPHeRe required almost 12 hours.

System	Total Time (s)	Average			
		Prec.	F-m.	Rec.	Inc. Degree
YAM++	2,066	0.94	0.82	0.73	14%
AML-BK	1,814	0.91	0.79	0.71	44%
LogMap-BK	2,391	0.90	0.78	0.70	*0%*
AML-BK-R	1,987	0.92	0.78	0.69	*0%*
AML	1,681	0.93	0.78	0.68	43%
LogMap	2,485	0.91	0.78	0.69	*0%*
AML-R	1,821	0.94	0.78	0.67	*0%*
ServOMap	15,300	0.87	0.77	0.69	52%
GOMMA$_{2012}$	1,920	0.81	0.65	0.57	29%
LogMapLt	371	0.87	0.60	0.52	34%
SPHeRe	42,040	0.86	0.57	0.46	21%
IAMA	704	0.91	0.52	0.39	46%

Table 10. Summary results for the top systems (values in italics are not absolute 0 but are caused by rounding).

6.10 Conclusions

Although the proposed matching tasks represent a significant leap in complexity with respect to the other OAEI test cases, the results have been very promising and 12 systems (including all system configurations) completed all matching tasks with very competitive results.

There is, however, plenty of room for improvement: (1) most of the participating systems disregard the coherence of the generated alignments; (2) the size of the input ontologies should not significantly affect efficiency, and (3) recall in the tasks involving SNOMED should be improved while keeping the current precision values.

The alignment coherence measure was the weakest point of the systems participating in this test case. As shown in Tables 7-10, even highly precise alignment sets may lead to a huge number of unsatisfiable classes. The use of techniques to assess mapping coherence is critical if the input ontologies together with the computed mappings are to be used in practice. Unfortunately, only a few systems in OAEI 2013 have shown to successfully use such techniques. We encourage ontology matching system developers to develop their own repair techniques or to use state-of-the-art techniques such as Alcomo [21], the repair module of LogMap (LogMap-Repair) [17] or the repair module of AML [26], which have shown to work well in practice [19].

7 MultiFarm

For evaluating the ability of matching systems to deal with ontologies in different natural languages, the MultiFarm data set has been proposed [22]. This data set results from the translation of 7 Conference test case ontologies (cmt, conference, confOf, iasted, sigkdd, ekaw and edas), into 8 languages (Chinese, Czech, Dutch, French, German, Portuguese, Russian, and Spanish, in addition to English). The 9 language versions result in 36 pairs of languages. For each pair of language, we take into account the alignment direction (cmt$_{en}$→confOf$_{de}$ and cmt$_{de}$→confOf$_{en}$, for instance, as two matching

tasks), what results in 49 alignments. Hence, MultiFarm contains 36×49 matching tasks.

7.1 Experimental setting

For the 2013 evaluation campaign, we have used a subset of the whole MultiFarm data set, omitting all the matching tasks involving the edas and ekaw ontologies (resulting in $36 \times 25 = 900$ matching tasks). In this sub set, we can distinguish two types of matching tasks: (i) those test cases where two different ontologies have been translated in different languages, e.g., cmt→confOf, and (ii) those test cases where the same ontology has been translated in different languages, e.g., cmt→cmt. For the test cases of type (ii), good results are not necessarily related to the use of specific techniques for dealing with ontologies in different natural languages, but on the ability to exploit the fact that both ontologies have an identical structure (and that the reference alignment covers all entities described in the ontologies).

This year, 7 systems (out of 23 participants, see Table 2) use specific cross-lingual[11] methods : CIDER-CL, MapSSS, RiMOM2013, StringsAuto, WeSeE, WikiMatch, and YAM++. This maintains the number of participants implementing specific modules as in 2012 (ASE, AUTOMSv2, GOMMA, MEDLEY, WeSeE, WikiMatch, and YAM++), counting on 4 new participants (some of them, extensions of systems participating in previous campaigns). The other systems are not specifically designed to match ontologies in different languages nor do they use any component for that purpose. CIDER-CL uses textual definitions of concepts (from Wikipedia articles) and computes co-occurrence information between multilingual definitions. MapSSS, StringsAuto and RiMOM2013[12] apply translation, using Google Translator API, before the matching step. In particular, RiMOM2013 uses a two-step translation: a first step for translating labels from the target language into the source language and a second step for translating all labels into English (for using WordNet). WeSeE uses the Web Translator API and YAM++ uses Microsoft Bing Translation, where both of them consider English as pivot language. Finally, WikiMatch exploits Wikipedia for extracting cross-language links for helping in the task of finding correspondences between the ontologies.

7.2 Execution setting and runtime

All systems have been executed on a Debian Linux virtual machine configured with four processors and 20GB of RAM running under a Dell PowerEdge T610 with 2*Intel

[11] We have revised the definitions of multilingual and cross-lingual matching. Initially, as reported in [22], MultiFarm was announced as a benchmark for multilingual ontology matching, i.e., *multilingual* in the sense that we have a set of ontologies in 8 languages. However, it is more appropriate to use the term *cross-lingual* ontology matching. Cross-lingual ontology matching refers to the matching cases where each ontology uses a different natural language (or a different set of natural languages) for entity naming, i.e., the intersection of sets is empty. It is the case of the matching tasks in MultiFarm.

[12] These 3 systems have encountered problems for accessing Google servers. New versions of these tools were received after the deadline, improving, for some test cases, the results reported here.

Xeon Quad Core 2.26GHz E5607 processors and 32GB of RAM, under Linux ProxMox 2 (Debian). The runtimes for each system can be found in Table 11. The measurements are based on 1 run. We can observe large differences between the time required for a system to complete the 900 matching tasks. While RiMOM requires around 13 minutes, WeSeE takes around 41 hours. As we have used this year a different setting from the one in 2012, we are not able to compare runtime measurements over the campaigns.

7.3 Evaluation results

Overall results Before discussing the results per pairs of languages, we present the aggregated results for the test cases within type (i) and (ii) matching task. Table 11 shows the aggregated results. Systems not listed in this table have generated empty alignments, for most test cases (ServOMap) or have thrown exceptions (CroMatcher, XMapGen, XMapSiG). For computing these results, we do not distinguish empty and erroneous alignments. As shown in Table 11, we observe significant differences between the results obtained for each type of matching task (specially in terms of precision). Most of the systems that implement specific cross-lingual techniques – YAM++ (.40), WikiMatch (.27), RiMOM2013 (.21), WeSeE (.15), StringsAuto (.14), and MapSSS (.10) – generate the best results for test cases of type (i). For the test cases of type (ii), systems non specifically designed for cross-lingual matching – MaasMatch and OntoK – are in the top-5 F-measures together with YAM++, WikiMatch and WeSeE. Concerning CIDER-CL, this system in principle is able to deal with a subset of languages, i.e., DE, EN, ES, and NL.

Overall (for both types i and ii), in terms of F-measure, most systems implementing specific cross-lingual methods outperform non-specific systems: YAM++ (.50), WikiMatch (.22), RiMOM (.17), WeSeE (.15) – with MaasMatch given its high scores on cases (ii) – and StringsAuto (.10).

Comparison with previous campaigns In the first year of evaluation of MultiFarm, we have used a subset of the whole data set, where we omitted the ontologies edas and ekaw, and suppressed the test cases where Russian and Chinese were involved. Since 2012, we have included Russian and Chinese translations, but still have not included edas and ekaw. In the 2011.5 intermediary campaign, 3 participants (out of 19) used specific techniques – AUTOMSv2, WeSeE, and YAM++. In 2012, 7 systems (out of 24) implemented specific techniques for dealing with ontologies in different natural languages – ASE, AUTOMSv2, GOMMA, MEDLEY, WeSeE, WikiMatch, and YAM++. This year, as in 2012, 7 participants out of 21 use specific techniques: 2 of them have been participating since 2011.5 (WeSeE and YAM), 1 since 2012 (WikiMatch), 3 systems (CIDER-CL, RiMOM2013 and MapSSS) have included cross-lingual approaches in their implementations, and 1 new system (StringsAuto) has participated.

Comparing 2012 and 2013 results (on the same basis), WikiMatch improved precision for both test case types – from .22 to .34 for type (i) and .43 to .65 for type (ii) – preserving its values of recall. On the other hand, WeSeE has decreased both precision – from .61 to .22 – and recall – from .32 to .12 – for type (i) and precision – from .90 to .56 – and recall – from .27 to .09 – for type (ii).

			Different ontologies (i)			Same ontologies (ii)		
	System	Runtime	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.
Cross-lingual	CIDER-CL	110	.03	.03	.04	.18	.06	.04
	MapSSS	2380	.27	.10	.07	.50	.06	.03
	RiMOM2013	13	.52	.21	.13	.87	.14	.08
	StringsAuto	24	.30	.14	.09	.51	.07	.04
	WeSeE	2474	.22	.15	.12	.56	.16	.09
	WikiMatch	1284	.34	.27	.23	.65	.18	.11
	YAM++	443	.51	**.40**	.36	.91	**.60**	.50
Non specific	AML	7	.14	.04	.03	.35	.03	.01
	HerTUDA	46	.00	.01	1.0	.02	.03	1.0
	HotMatch	17	.00	.00	.00	.41	.04	.02
	IAMA	15	.15	.05	.03	.58	.04	.02
	LogMap	18	.18	.05	.03	.35	.03	.01
	LogMapLite	6	.13	.04	.02	.25	.02	.01
	MaasMatch	70	.01	.02	.03	.62	**.29**	.19
	ODGOMS	44	.26	**.10**	.06	.47	.05	.03
	OntoK	1602	.01	.01	.05	.16	.16	.15
	Synthesis	67	.30	.05	.03	.25	.04	.03

Table 11. MultiFarm aggregated results per matcher, for each type of matching task – types (i) and (ii). Runtime is measured in minutes (time for completing the 900 matching tasks).

Language specific results Table 12 shows the results aggregated per language pair, for the the test cases of type (i). For the sake of readability, we present only F-measure values. The reader can refer to the OAEI results web page for more detailed results on precision and recall. As expected and already reported above, the systems that apply specific strategies to match ontology entities described in different natural languages outperform the other systems. For most of these systems, the best performance is observed for the pairs of language including Dutch, English, German, Spanish, and Portuguese: CIDER-CL (en-es .21, en-nl and es-nl .18, de-nl .16), MapSSS (es-pt and en-es .33, de-en .28, de-es .26), RiMOM2013 (en-es .42, es-pt .40, de-en .39), StringsAuto (en-es .37, es-pt .36, de-en .33), WeSeE (en-es .46, en-pt .41, en-nl .40), WikiMatch (en-es .38, en-pt, es-pt and es-fr .37, es-ru .35). The exception is YAM++ which generates its best results for the pairs including Czech : cz-en and en-pt .57, cz-pt .56, cz-nl and fr-pt .53. For all specific systems, English is present in half of the top pairs.

For non-specific systems, most of them cannot deal at all with Chinese and Russian languages. 7 out of 10 systems generate their best results for the pair es-pt (followed by the pair de-en). Again, similarities in the language vocabulary have an important role in the matching task. On the other hand, although it is likely harder to find correspondences between cz-pt than es-pt, for some systems Czech is on pairs for the top-5 F-measure (cz-pt, for AML, IAMA, LogMap, LogMapLite and Synthesis). It can be explained by the specific way systems combine their internal matching techniques (ontology structure, reasoning, coherence, linguistic similarities, etc).

	AML	CIDER-CL	HerTUDA	HotMatch	IAMA	LogMap	LogMapLite	MaasMatch	MapSSS	ODGOMS	OntoK	RiMOM2013	StringsAuto	Synthesis	WeSeE	WikiMatch	YAM++
cn-cz		.00	.01								.01	.12				.12	.36
cn-de		.00	.01					.00			.01	.18				.15	.37
cn-en		.01	.01								.01	.25				.16	.43
cn-es		.00	.01	.01				.00			.01	.17				.24	.19
cn-fr		.01	.01	.01				.01			.01	.17			.01	.10	.42
cn-nl		.00	.01	.01				.00			.01	.16				.19	.31
cn-pt		.01	.01								.01	.10				.18	.32
cn-ru		.00	.01	.01						.00	.01				.01	.23	.34
cz-de	.10	.00	.01		.10	.09	.09	.01	.14	.13	.01	.24	.22	.11	.02	.26	.48
cz-en	.04	.00	.01		.12	.05	.04	.02	.25	.24	.01	.25	.29	.06	.08	.18	.57
cz-es	.11	.00	.01		.11	.11	.11	.02	.18	.18	.03	.24	.22	.14	.09	.29	.19
cz-fr	.01	.00	.01	.01	.01	.01	.01	.01	.15	.07	.02	.17	.17	.02	.04	.22	.52
cz-nl	.04	.00	.01		.05	.04	.04	.01	.12	.12	.03	.32	.18	.04	.01	.23	.53
cz-pt	.12	.01	.01		.13	.13	.13	.01	.18	.20	.02	.24	.23	.16	.04	.25	.56
cz-ru		.01	.01								.01					.25	.48
de-en	.20	.12	.01		.21	.22	.20	.05	.28	.30	.01	.39	.33	.22	.33	.32	.50
de-es	.07	.15	.01		.07	.09	.06	.02	.26	.24	.01	.31	.29	.08	.28	.29	.19
de-fr	.05	.01	.01		.04	.04	.04	.02	.08	.06	.01	.29	.23	.04	.32	.26	.44
de-nl	.05	.16	.01		.04	.04	.04	.04	.17	.21	.01	.30	.19	.05	.31	.27	.40
de-pt	.07		.01		.08	.07	.07	.02	.13	.11	.02	.27	.18	.09	.34	.26	.41
de-ru		.00	.01	.01				.01			.01					.31	.47
en-es	.06	.21	.01		.05	.15	.04	.03	.33	.30	.01	.42	.37	.05	.46	.38	.23
en-fr	.06		.01	.01	.10	.06	.04	.05	.17	.15	.01	.32	.29	.04	.32	.33	.50
en-nl	.07	.18	.01		.07	.07	.10	.05	.22	.24	.02	.35	.24	.08	.40	.32	.52
en-pt	.06		.01		.06	.06	.06	.02	.18	.19	.02	.36	.30	.07	.41	.37	.57
en-ru		.00	.01					.00			.01					.24	.50
es-fr	.03		.01		.06	.06	.01	.03	.17	.14	.02	.36	.29	.02	.25	.37	.20
es-nl		.18	.01					.02	.07	.03	.01	.29	.15		.34	.32	.16
es-pt	.29		.01		.29	.24	.23	.05	.33	.33	.04	.40	.36	.25	.33	.37	.25
es-ru		.00	.01								.02					.35	.19
fr-nl	.12	.00	.01	.01	.12	.13	.12	.06	.16	.13	.03	.30	.26	.15	.24	.34	.46
fr-pt	.01		.01	.01	.01			.04	.10	.11		.26	.19		.34	.28	.53
fr-ru		.01	.01					.00			.01					.33	.46
nl-pt	.01		.01		.02	.01	.01	.02	.03	.04	.01	.15	.07	.02	.31	.27	.51
nl-ru		.00	.01					.01			.01					.33	.44
pt-ru		.00	.01					.00			.01					.29	.47

Table 12. MultiFarm results per pair of languages, for the test cases of type (i). In this detailed view, we distinguished empty alignments, represented by empty cells, from wrong ones (.00)

7.4 Conclusion

As expected, systems using specific methods for dealing with ontologies in different languages work much better than non specific systems. However, the absolute results

are still not very good, if compared to the top results of the original Conference data set (approximatively 75% F-measure for the best matcher). For all specific cross-lingual methods, the techniques implemented in YAM++, as in 2012, generate the best alignments in terms of F-measure (around 50% overall F-measure for both types of matching tasks). All systems privilege precision rather than recall. Although we count this year on 4 new systems implementing specific cross-lingual methods, there is room for improvements to achieve the same level of compliance as in the original data set.

8 Library

The library test case was established in 2012[13]. The test case consists of matching of two real-world thesauri: The Thesaurus for the Social Sciences (TheSoz, maintained by GESIS) and the Standard Thesaurus for Economics (STW, maintained by ZBW). The reference alignment is based on a manually created alignment from 2006. As additional benefit from this test case, the reference alignment is constantly improved by the maintainers by manually checking the generated correspondences that have not yet been checked and that are not part of the reference alignment[14].

8.1 Test data

Both thesauri used in this test case are comparable in many respects. They have roughly the same size (6,000 resp. 8,000 concepts), are both originally developed in German, are today both multilingual, both have English translations, and, most important, despite being from two different domains, they have significant overlapping areas. Not least, both are freely available in RDF using SKOS[15]. To enable the participation of all OAEI matchers, an OWL version of both thesauri is provided, effectively by creating a class hierarchy from the concept hierarchy. Details are provided in the report of the 2012 campaign [1]. As stated above, we updated the reference alignment with all correct correspondences found during the 2012 campaign, it now consists of 3161 correspondences.

8.2 Experimental setting

All matching processes have been performed on a Debian machine with one 2.4GHz core and 7GB RAM allocated to each system. The evaluation has been executed by using the SEALS infrastructure. Each participating system uses the OWL version.

To compare the created alignments with the reference alignment, we use the Alignment API. For this evaluation, we only included equivalence relations (skos:exactMatch). We computed precision, recall and F_1-measure for each matcher. Moreover, we measured the runtime, the size of the created alignment and checked

[13] There has already been a library test case from 2007 to 2009 using different thesauri, as well as other thesaurus test cases like the food and the environment test cases.

[14] With the reasonable exception of XMapGen, which produces almost 40.000 correspondences.

[15] `http://www.w3.org/2004/02/skos`

whether a 1:1 alignment has been created. To assess the results of the matchers, we developed three straightforward matching strategies, using the original SKOS version of the thesauri:

- MatcherPrefDE: Compares the German lower-case preferred labels and generates a correspondence if these labels are completely equivalent.
- MatcherPrefEN: Compares the English lower-case preferred labels and generates a correspondence if these labels are completely equivalent.
- MatcherPref: Creates a correspondence, if either MatcherPrefDE or MatcherPrefEN or both create a correspondence.
- MatcherAllLabels: Creates a correspondences whenever at least one label (preferred or alternative, all languages) of an entity is equivalent to one label of another entity.

8.3 Results

Of all 21 participating matchers (or variants), 12 were able to generate an alignment within 12 hours. CroMatcher, MaasMatch, RiMOM2013, WeSeE and WikiMatch did not finish in the time frame, OntoK had heap space problems and CiderCL, MapSSS and Synthesis threw an exception. The results can be found in Table 13.

Matcher	Precision	F-Measure	Recall	Time (ms)	Size	1:1
ODGOMS	0.70	0.76	0.83	27936433	3761	-
YAM++	0.69	0.74	0.81	731860	3689	-
MatcherPref	0.91	0.74	0.63	-	2190	-
ServOMap	0.70	0.74	0.78	648138	3540	-
AML	0.62	0.7	0.88	39366	4433	-
MatcherPrefDE	0.98	0.73	0.58	-	1885	-
MatcherAllLabels	0.61	0.72	0.88	-	4605	-
LogMap	0.78	0.70	0.64	98958	2622	-
LogMapLite	0.65	0.70	0.77	20312	3775	-
HerTUDA	0.52	0.67	0.92	11228741	5559	-
HotMatch	0.73	0.65	0.58	12128682	2494	√
MatcherPrefEN	0.88	0.57	0.42	-	1518	-
XmapSig	0.80	0.45	0.32	2914167	1256	-
StringsAuto	0.77	0.30	0.19	1966012	767	√
IAMA	0.78	0.08	0.04	18599	166	-
XmapGen	0.03	0.06	0.37	3008820	38360	-

Table 13. Results of the Library test case (ordered by F-measure).

The best systems in terms of F-measure are ODGOMS and YAM++. These matchers also have a higher F-measure than MatcherPref. ServOMap and AML are below this baseline but better than MatcherPrefDE and MatcherAllLabels. A group of matchers including LogMap, LogMapLite, HerTUDA and HotMatch are above the MatcherPrefEN baseline. Compared to last year evaluation with the updated reference alignment, the matchers clearly improved: in 2012, no matcher was able to beat MatcherPref and MatcherPrefDE, only ServOMapLt was better than MatcherAllLabels. Today, two

matchers outperformed all baselines; further two matchers outperformed all baselines but MatcherPref. This is remarkable, as the matchers are still not able to consume SKOS and therefore neglect the distinction between preferred and alternative labels. The baselines are tailored for very high precision by design, while the matchers usually have a higher recall. This is reflected in the F-measure, where the highest value increased from 0.72 to 0.76 by almost 5 percentage points since last year. The recall mostly increased, e.g. YAM++ from 0.76 to 0.81 (without affecting the precision negatively, which also increased from 0.68 to 0.69).

Like in the previous year, an additional intellectual evaluation of the alignments established automatically was done by a domain expert to further improve the reference alignment. Unsurprisingly, the matching tools predominantly detected matches based on the character string. This included the term alone as well as the term's context. Especially in the case of short terms, this could easily lead to wrong correspondences, e.g., "tea" $\neq$ "team", "sheep" $\neq$ "sleep". Except for its sequence of letters the term's context was not taken into account.

This sole attention to the character string was a main source of error in cases in which on the term as well as on the context level similar terminological entities appeared, e.g., "Green revolution" subject category: "Development Politic" $\neq$ "permanent revolution" subject category: "Political Developments and Processes".

Additionally, identical components of a compound frequently lead to incorrect correspondences, e.g., "prohibition of interest" $\neq$ "prohibition of the use of force". Moreover, terms in different domains might look similar, but in fact have very different meanings. An illustrative example is "Chicago Antitrust Theory" $\neq$ "Chicago School", where indeed the same Chicago is referenced, but without any effect on the (dis-)similarity of both concepts.

8.4 Conclusion

The overall performance improvement is encouraging in this test case. While it might not look impressive to beat simple baselines as ours at first sight, it is actually a notable achievement. The baselines are not only tailored for very high precision, benefitting from the fact that in many cases a consistent terminology is used, they also exploit additional knowledge about the labels. The matchers are general-purpose matchers that have to perform well in all OAEI test cases. Nonetheless, there does not seems to be matchers who understand SKOS in order to make use of the many concept hierarchies provided on the Web.

Generally, matchers still rely too much on the character string of the labels and the labels of the concepts in the immediate vicinity. During the intellectual evaluation process, it became obvious that a multitude of incorrect matches could be prevented if the subject categories, respectively the thesauri's classification schemes be matched beforehand. In many cases, misleading candidate correspondences could be discarded by taking these higher levels of the hierarchy into account. It could be prevented, for example, to build up correspondences between personal names and subject headings. A thesaurus, however, is not a classification system. The disjointness of two subthesauri is therefore not easy to establish, let alone to detect by automatic means. Nonetheless,

thesauri oftentimes have their own classification schemes which partly follow classification principles. We believe that further exploiting this context knowledge could be worthwhile.

9 Interactive matching

The interactive matching test case was evaluated at OAEI 2013 for the first time. The goal of this evaluation is to simulate interactive matching [23], where a human expert is involved to validate mappings found by the matching system. In the evaluation, we look at how user interaction may improve matching results.

For the evaluation, we use the conference data set 5 with the ra1 alignment, where there is quite a bit of room for improvement, with the best fully automatic, i.e., non-interactive matcher achieving an F-measure below 80%. The SEALS client was modified to allow interactive matchers to ask an oracle, which emulates a (perfect) user. The interactive matcher can present a correspondence to the oracle, which then tells the user whether the correspondence is right or wrong.

All matchers participating in the interactive test case support both interactive and non-interactive matching. This allows us to analyze how much benefit the interaction brings for the individual matchers.

9.1 Results

Overall, five matchers participated in the interactive matching test case: AML and AML-bk, Hertuda, LogMap, and WeSeE-Match. All of them implement interactive strategies that run entirely as a post-processing step to the automatic matching, i.e., take the alignment produced by the base matcher and try to refine it by selecting a suitable subset.

AML and AML-bk present all correspondences below a certain confidence threshold to the oracle, starting with the highest confidence values. They stop adding references once the false positive rate exceeds a certain threshold. Similarly, LogMap checks all questionable correspondences using the oracle. Hertuda and WeSeE-Match try to adaptively set an optimal threshold for selecting correspondences. They perform a binary search in the space of possible thresholds, presenting a correspondence of average confidence to the oracle first. If the result is positive, the search is continued with a higher threshold, otherwise with a lower threshold.

The results are depicted in Table 14. Please note that the values in this table slightly differ from the original values in the conference test case, since the latter uses micro average recall and precision, while we use macro averages, so that we can compute significance levels using T-Tests on the series of recall and precision values from the individual test cases. The reason for the strong divergence of the results for WeSeE to the conference test case results is unknown. Altogether, the biggest improvement in F-measure, as well as the best overall result (although almost at the same level as AML-bk), is achieved by LogMap, which increases its F-measure by four percentage points. Furthermore, LogMap, AML and AML-bk show a statistically significant increase in recall as well as precision, while all the other tools except for Hertuda show a significant

	AML	AML-bk	Hertuda	LogMap	WeSeE
Non-Interactive Results					
Precision	0.88	0.88	0.77	0.83	0.57
F-measure	0.69	0.71	0.62	0.68	0.49
Recall	0.59	0.61	0.53	0.62	0.46
Interactive Results					
Precision	[1]**0.91**	[1]**0.91**	**0.79**	[1]**0.90**	[1]**0.73**
F-measure	[1]**0.71**	[1]**0.73**	0.58	[1]**0.73**	0.47
Recall	[5]**0.61**	[5]**0.63**	0.50	[5]**0.64**	0.40
Average Number of Interactions					
Positive	1.43	1.57	1.95	2.57	1.67
Negative	5.14	5.05	10.33	1.76	3.81
Total	6.57	6.67	12.33	4.33	5.48

Table 14. Results of the interactive matching test case. The table reports both the results with and without interaction, in order to analyze the improvement that was gained by adding interactive features. Improvements of the interactive variants over the non-interactive variants are shown in bold. Statistically significant differences are marked with $^5(p < 0.05)$ and $^1(p < 0.01)$. Furthermore, we report the average number of interactions, showing both the positive and negative examples presented to the oracle.

increase in precision. The increase in precision is in all cases however higher than the increase of recall. It can be observed for AML, AML-bk and LogMap that a highly significant increase in precision also increases F-measure at a high significance level, even if the increase in recall is less significant.

At the same time, LogMap has the lowest number of interactions with the oracle, which shows that it also makes the most efficient use of the oracle. In a truly interactive setting, this would mean that the manual effort is minimized. Furthermore, it is the only tool that presents more positive than negative examples to the oracle.

On the other hand, Hertuda and WeSeE even show a decrease in recall, which cannot be compensated by the increase in precision. The biggest increase in precision (17 percentage points) is achieved by WeSeE, but on an overall lower level than the other matching systems. Thus, we conclude that their strategy is not as efficient as those of the other participants.

Compared to the results of the non-interactive conference test case, the best interactive matcher (in terms of F-measure) is slightly below the best matcher (YAM++) with a F-measure value of 0.76 (using macro averages). Except for YAM++, the interactive versions of AML-bk, AML and LogMap achieve better F-measure scores than all non-interactive matchers.

9.2 Discussion

The results show that current interactive matching tools mainly use interaction as a means to post-process an alignment found with fully automatic means. There are, however, other interactive approaches that can be thought of, which include interaction at an earlier stage of the process, e.g., using interaction for parameter tuning [25], or determining anchor elements for structure-based matching approaches using interactive

methods. The maximum F-measure of 0.73 achieved shows that there is still room for improvement.

Furthermore, different variations of the evaluation method can be thought of, including different noise levels in the oracle's responses, i.e., simulating errors made by the human expert, or allowing other means of interactions than the validation of single correspondences, e.g., providing a random positive example, or providing the corresponding element in one ontology, given an element of the other one.

So far, we only compare the final results of the interactive matching process. In [23], we have introduced an evaluation method based on *learning curves*, which gives insights into how quickly the matcher converges towards its final result. However, we have not implemented that model for this year's OAEI, since it requires more changes to the matchers (each matcher has to provide an intermediate result at any point in time).

10 Instance matching

The instance matching track aims at evaluating the performance of different matching tools on the task of matching RDF individuals which originate from different sources but describe the same real-world entity [16].

10.1 RDFT test cases

Starting from the experience of previous editions of the instance matching track in OAEI [15], this year we provided a set of RDF-based test cases, called RDFT, that is automatically generated by introducing controlled transformations in some initial RDF data sets. The controlled transformations introduce artificial distortions into the data, which include data value transformations as well as structural transformations. RDFT includes blind evaluation. The participants are provided with a list of five test cases. For each test case, we provide training data with the accompanying alignment to be used to adjust the settings of the tools, and contest data, based on which the final results will be calculated. The evaluation data set is generated by exploiting the same configuration of the RDFT transformation tool used for generating training data.

The RDFT test cases have been generated from an initial RDF data set about well-known computer scientists data extracted from DBpedia. The initial data set is composed by 430 resources, 11 RDF properties and 1744 triples. Some descriptive statistics about the initial data set are available online. Starting from the initial data set, we provided the participants with five test cases, where different transformations have been implemented, as follows:

- **Testcase 1: value transformation.** Values of 5 properties have been changed by randomly deleting/adding chars, by changing the date format, and/or by randomly change integer values.
- **Testcase 2: structure transformation.** The length of property path between resources and values has been changed. Property assertions have been split in two or more assertions.
- **Testcase 3: languages.** The same as Testcase 1, but using French translation for comments and labels instead of English.

	tescase01			tescase02			tescase03			tescase04			tescase05		
system	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.	Prec.	F-m.	Rec.
LilyIOM	1.00	**1.00**	0.99	0.74	0.74	0.74	0.94	0.93	0.92	0.71	0.72	0.73	0.71	0.58	0.49
LogMap	0.97	0.80	0.69	0.79	0.88	0.99	0.98	0.84	0.73	0.95	0.80	0.70	0.92	0.74	0.62
RiMOM2013	1.00	**1.00**	1.00	0.95	0.97	0.99	0.96	**0.98**	0.99	0.94	**0.96**	0.98	0.93	**0.96**	0.99
SLINT+	0.98	0.98	0.98	1.00	**1.00**	1.00	0.94	0.92	0.91	0.91	0.91	0.91	0.87	0.88	0.88

Table 15. Results for the RDFT test cases.

- **Testcase 4: combined.** A combination of value and structure transformations using French text.
- **Testcase 5: cardinality.** The same as Testcase 4, but now part of the resources have none or multiple matching counterparts.

10.2 RDFT results

An overview of the precision, recall and F_1-measure results for the RDFT test cases is shown in Table 15.

All the tools show good performances when dealing with singular type of data transformation, i.e., Testcases 1-3, either value, structural, and language transformations. Performances drop when different kinds of transformations are combined together, i.e., Testcases 4-5, except for RiMOM2013, which still has performances close to 1.0 for both precision and recall. This suggests that a possible challenge for instance matching tools is to work in the direction of improving the combination and balancing of different matching techniques in a single, general-purpose, configuration scheme.

In addition to precision, recall and F_1-measure results, we performed also a test based on the similarity values provided by participating tools. In particular, we selected the provided mappings by different thresholds on the similarity values, in order to monitor the behavior of precision and recall[16]. Results of this second evaluation are shown in Figure 3.

Testing the results when varying the threshold used for mapping selection is useful to understand how robust are the mappings retrieved by the participating tools. In particular, RiMOM2013 is the only tool which has very good results with all the threshold values that have been tested. This means that the retrieved mappings are generally correct and associated with high levels of similarity. Other tools, especially LilyIOM and LogMap, retrieve a high number of mappings which are associated with low levels of confidence. In such cases, when we rely only on mappings between resources that are considered very similar by the tool, the quality of results becomes lower.

Finally, as a general remark suggested from the result analysis, we stress the opportunity of working toward two main goals in particular: one one side, on the integration of different matching techniques and the need of conceiving self-adapting tools, capable of self-configuring the most suitable combination of matching metrics according to the nature of data heterogeneity that needs to be handled; on the other side, the need for

[16] This experiment is partially useful in the case of SLINT+, where the similarity values are not in the range [0,1] and are not necessarily proportional to the elements similarity.

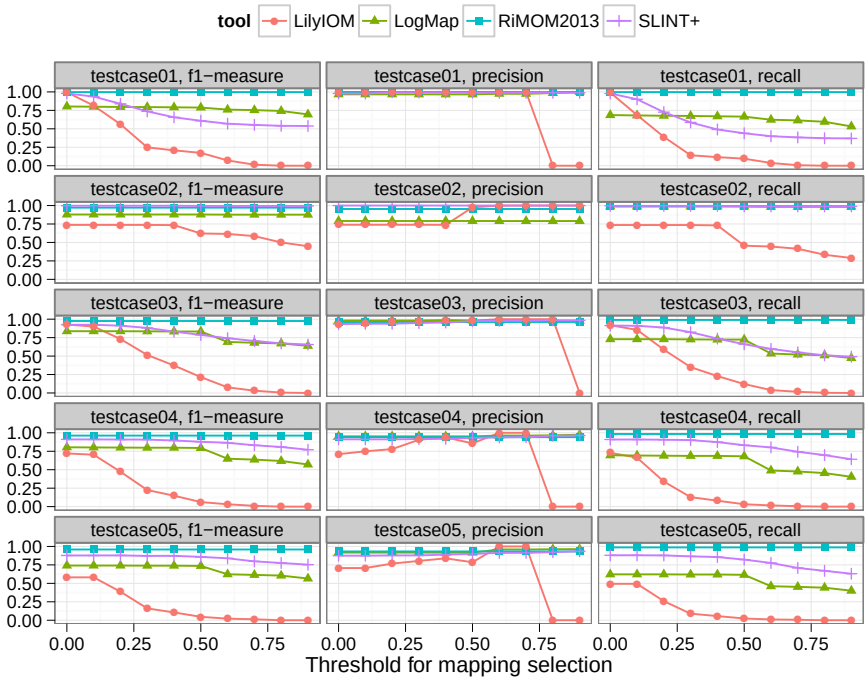

Fig. 3. Evaluation test based on the similarity values.

tools capable of providing a degree of confidence which could be used for measuring the reliability of the provided mappings.

11 Lesson learned and suggestions

There are, this year, very few comments about the evaluation execution:

A) This year indicated again that requiring participants to implement a minimal interface was not a strong obstacle to participation. Moreover, the community seems to get used to the SEALS infrastructure introduced for OAEI 2011. This might be one of the reasons for an increasing participation.
B) Related to the availability of the platform, participants checked that their tools were working on minimal tests and discovered in September that they were not working on other tests. For that reason, it would be good to set the preliminary evaluation results by the end of July.
C) Now that all tools are run in exactly the same configuration across all test cases, some discrepancies appear across such cases. For instance, benchmarks expect only class correspondences in the name space of the ontologies, some other cases expect something else. This is a problem, which could be solved either by passing parameters to the SEALS client (this would make its implementation heavier) or by post processing results (which may be criticized).
D) [24] raised and documented objections (on validity and fairness) to the way reference alignments are made coherent with alignment repair techniques. Appropriate measures should be taken to mitigate this.
E) Last years we reported that we had many new participants. The same trend can be observed for 2013.
F) Again and again, given the high number of publications on data interlinking, it is surprising to have so few participants to the instance matching track.

12 Conclusions

OAEI 2013 saw an increased number of participants and most of the test cases performed on the SEALS platform. This is good news for the interoperability of matching systems.

Compared to the previous years, we observed improvements of runtimes and the ability of systems to cope with large ontologies and data sets (testified by the largebio and instance matching results). This comes in addition to progress in overall F-measure, which is more observable as the test case is more recent. More preoccupying was the lack of robustness of some systems observed in the simple benchmarks. This seems to be due to an increased reliance on the network and networked resources that may time-out systems.

As usual, most of the systems favour precision over recall. In general, participating matching systems do not take advantage of alignment repairing system and return sometimes incoherent alignments. This is a problem if their result has to be taken as input by a reasoning system. They do not generally use natural language aware strategies, while the multilingual tests show the worthiness of such an approach.

A novelty of this year was the evaluation of interactive systems, included in the SEALS client. It brings interesting insight on the performances of such systems and should certainly be continued.

Most of the participants have provided a description of their systems and their experience in the evaluation. These OAEI papers, like the present one, have not been peer reviewed. However, they are full contributions to this evaluation exercise and reflect the hard work and clever insight people put in the development of participating systems. Reading the papers of the participants should help people involved in ontology matching to find what makes these algorithms work and what could be improved. Sometimes participants offer alternate evaluation results.

The Ontology Alignment Evaluation Initiative will continue these tests by improving both test cases and testing methodology for being more accurate. Matching evaluation still remains a challenging topic, which is worth further research in order to facilitate the progress of the field [27]. Further information can be found at:

`http://oaei.ontologymatching.org`.

Acknowledgements

We warmly thank the participants of this campaign. We know that they have worked hard for having their matching tools executable in time and they provided insightful papers presenting their experience. The best way to learn about the results remains to read the following papers.

We are very grateful to STI Innsbruck for providing the necessary infrastructure to maintain the SEALS repositories.

We are also grateful to Martin Ringwald and Terry Hayamizu for providing the reference alignment for the anatomy ontologies and thank Elena Beisswanger for her thorough support on improving the quality of the data set.

We thank Christian Meilicke for help with incoherence evaluation within the conference and his support of the anatomy test case.

We also thank for their support the other members of the Ontology Alignment Evaluation Initiative steering committee: Yannis Kalfoglou (Ricoh laboratories, UK), Miklos Nagy (The Open University (UK), Natasha Noy (Stanford University, USA), Yuzhong Qu (Southeast University, CN), York Sure (Leibniz Gemeinschaft, DE), Jie Tang (Tsinghua University, CN), Heiner Stuckenschmidt (Mannheim Universität, DE), George Vouros (University of the Aegean, GR).

Bernardo Cuenca Grau, Jérôme Euzenat, Ernesto Jimenez-Ruiz, Christian Meilicke, and Cássia Trojahn dos Santos have been partially supported by the SEALS (IST-2009-238975) European project in the previous years.

Ernesto and Bernardo have also been partially supported by the Seventh Framework Program (FP7) of the European Commission under Grant Agreement 318338, "Optique", the Royal Society, and the EPSRC projects Score!, ExODA and MaSI^3.

Cássia Trojahn dos Santos and Roger Granada are also partially supported by the CAPES-COFECUB Cameleon project number 707/11.

References

1. José Luis Aguirre, Bernardo Cuenca Grau, Kai Eckert, Jérôme Euzenat, Alfio Ferrara, Robert Willem van Hague, Laura Hollink, Ernesto Jiménez-Ruiz, Christian Meilicke, An-

driy Nikolov, Dominique Ritze, François Scharffe, Pavel Shvaiko, Ondrej Sváb-Zamazal, Cássia Trojahn, and Benjamin Zapilko. Results of the ontology alignment evaluation initiative 2012. In *Proc. 7th ISWC ontology matching workshop (OM), Boston (MA US)*, pages 73–115, 2012.

2. Ana Armas Romero, Bernardo Cuenca Grau, and Ian Horrocks. MORe: Modular combination of OWL reasoners for ontology classification. In *Proc. 11th International Semantic Web Conference (ISWC), Boston (MA US)*, pages 1–16, 2012.
3. Benhamin Ashpole, Marc Ehrig, Jérôme Euzenat, and Heiner Stuckenschmidt, editors. *Proc. K-Cap Workshop on Integrating Ontologies*, Banff (Canada), 2005.
4. Olivier Bodenreider. The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic Acids Research*, 32:267–270, 2004.
5. Caterina Caracciolo, Jérôme Euzenat, Laura Hollink, Ryutaro Ichise, Antoine Isaac, Véronique Malaisé, Christian Meilicke, Juan Pane, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Vojtech Svátek. Results of the ontology alignment evaluation initiative 2008. In *Proc. 3rd ISWC ontology matching workshop (OM), Karlsruhe (DE)*, pages 73–120, 2008.
6. Jérôme David, Jérôme Euzenat, François Scharffe, and Cássia Trojahn dos Santos. The alignment API 4.0. *Semantic web journal*, 2(1):3–10, 2011.
7. Jérôme Euzenat, Alfio Ferrara, Laura Hollink, Antoine Isaac, Cliff Joslyn, Véronique Malaisé, Christian Meilicke, Andriy Nikolov, Juan Pane, Marta Sabou, François Scharffe, Pavel Shvaiko, Vassilis Spiliopoulos, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Cássia Trojahn dos Santos, George Vouros, and Shenghui Wang. Results of the ontology alignment evaluation initiative 2009. In *Proc. 4th ISWC ontology matching workshop (OM), Chantilly (VA US)*, pages 73–126, 2009.
8. Jérôme Euzenat, Alfio Ferrara, Christian Meilicke, Andriy Nikolov, Juan Pane, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2010. In *Proc. 5th ISWC ontology matching workshop (OM), Shanghai (CN)*, pages 85–117, 2010.
9. Jérôme Euzenat, Alfio Ferrara, Robert Willem van Hague, Laura Hollink, Christian Meilicke, Andriy Nikolov, François Scharffe, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, and Cássia Trojahn dos Santos. Results of the ontology alignment evaluation initiative 2011. In *Proc. 6th ISWC ontology matching workshop (OM), Bonn (DE)*, pages 85–110, 2011.
10. Jérôme Euzenat, Antoine Isaac, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2007. In *Proc. 2nd ISWC ontology matching workshop (OM), Busan (KR)*, pages 96–132, 2007.
11. Jérôme Euzenat, Christian Meilicke, Pavel Shvaiko, Heiner Stuckenschmidt, and Cássia Trojahn dos Santos. Ontology alignment evaluation initiative: six years of experience. *Journal on Data Semantics*, XV:158–192, 2011.
12. Jérôme Euzenat, Malgorzata Mochol, Pavel Shvaiko, Heiner Stuckenschmidt, Ondrej Svab, Vojtech Svatek, Willem Robert van Hage, and Mikalai Yatskevich. Results of the ontology alignment evaluation initiative 2006. In *Proc. 1st ISWC ontology matching workshop (OM), Athens (GA US)*, pages 73–95, 2006.
13. Jérôme Euzenat, Maria Rosoiu, and Cássia Trojahn dos Santos. Ontology matching benchmarks: generation, stability, and discriminability. *Journal of web semantics*, 21:30–48, 2013.
14. Jérôme Euzenat and Pavel Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2nd edition, 2013.
15. Alfio Ferrara, Andriy Nikolov, Jan Noessner, and François Scharffe. Evaluation of instance matching tools: The experience of OAEI. *Journal of Web Semantics*, 21:49–60, 2013.

16. Alfio Ferrara, Andriy Nikolov, and François Scharffe. Data linking for the semantic web. *International Journal on Semantic Web and Information Systems*, 7(3):46–76, 2011.
17. Ernesto Jiménez-Ruiz and Bernardo Cuenca Grau. LogMap: Logic-based and scalable ontology matching. In *Proc. 10th International Semantic Web Conference (ISWC), Bonn (DE)*, pages 273–288, 2011.
18. Ernesto Jiménez-Ruiz, Bernardo Cuenca Grau, Ian Horrocks, and Rafael Berlanga. Logic-based assessment of the compatibility of UMLS ontology sources. *J. Biomed. Sem.*, 2, 2011.
19. Ernesto Jiménez-Ruiz, Christian Meilicke, Bernardo Cuenca Grau, and Ian Horrocks. Evaluating mapping repair systems with large biomedical ontologies. In *Proc. 26th Description Logics Workshop*, 2013.
20. Yevgeny Kazakov, Markus Krötzsch, and Frantisek Simancik. Concurrent classification of EL ontologies. In *Proc. 10th International Semantic Web Conference (ISWC), Bonn (DE)*, pages 305–320, 2011.
21. Christian Meilicke. *Alignment Incoherence in Ontology Matching*. PhD thesis, University Mannheim, 2011.
22. Christian Meilicke, Raúl García Castro, Frederico Freitas, Willem Robert van Hage, Elena Montiel-Ponsoda, Ryan Ribeiro de Azevedo, Heiner Stuckenschmidt, Ondrej Sváb-Zamazal, Vojtech Svátek, Andrei Tamilin, Cássia Trojahn, and Shenghui Wang. MultiFarm: A benchmark for multilingual ontology matching. *Journal of web semantics*, 15(3):62–68, 2012.
23. Heiko Paulheim, Sven Hertling, and Dominique Ritze. Towards evaluating interactive ontology matching tools. In *Proc. 10th Extended Semantic Web Conference (ESWC), Montpellier (FR)*, pages 31–45, 2013.
24. Catia Pesquita, Daniel Faria, Emanuel Santos, and Francisco Couto. To repair or not to repair: reconciling correctness and coherence in ontology reference alignments. In *Proc. 8th ISWC ontology matching workshop (OM), Sydney (AU)*, page this volume, 2013.
25. Dominique Ritze and Heiko Paulheim. Towards an automatic parameterization of ontology matching tools based on example mappings. In *Proc. 6th ISWC ontology matching workshop (OM), Bonn (DE)*, pages 37–48, 2011.
26. Emanuel Santos, Daniel Faria, Catia Pesquita, and Francisco Couto. Ontology alignment repair through modularization and confidence-based heuristics. *CoRR*, abs/1307.5322, 2013.
27. Pavel Shvaiko and Jérôme Euzenat. Ontology matching: state of the art and future challenges. *IEEE Transactions on Knowledge and Data Engineering*, 25(1):158–176, 2013.
28. York Sure, Oscar Corcho, Jérôme Euzenat, and Todd Hughes, editors. *Proc. ISWC Workshop on Evaluation of Ontology-based Tools (EON), Hiroshima (JP)*, 2004.
29. Cássia Trojahn dos Santos, Christian Meilicke, Jérôme Euzenat, and Heiner Stuckenschmidt. Automating OAEI campaigns (first report). In *Proc. ISWC Workshop on Evaluation of Semantic Technologies (iWEST), Shanghai (CN)*, 2010.

Oxford, Linköping Mannheim, Grenoble, Milano, Porto Alegre, Toulouse, Köln,
Walldorf, Montpellier, Trento, Prague
November 2013