Anchor-Flood: Results for OAEI 2009

Md. Hanif Seddiqui and Masaki Aono

Toyohashi University of Technology, Japan
hanif@kde.ics.tut.ac.jp, aono@ics.tut.ac.jp

Abstract. Our ontology schema matching algorithm takes the essence of the *locality of reference* by considering the neighboring concepts and relations to align the entities of ontologies. It starts off a seed point called an *anchor* (a pair of "look-alike" concepts across ontologies) and collects two blocks of neighboring concepts across ontologies. The concepts of the pair of blocks are aligned and the process is repeated for newly found aligned pairs. This year, we use a semantically reformed dynamic block of concepts starting from an anchor-concept and produce two blocks from one anchor to get alignment. We improve our memory management. The experimental results show its effectiveness against the benchmark, anatomy track and other datasets. We also extend our algorithm to match instances of IIMB benchmarks and we obtained effective results.

1 Presentation of the system

During OAEI-2008, our ontology alignment system used the *locality of reference* for collecting neighboring concepts with strong semantic arbitrary depth for aligning concepts across pair of ontologies. This year, we incorporate a process of collecting concepts with strong intrinsic semantic similarity within ontology elements considering intrinsic Information Content (IC) [6] to form a dynamic block. Hence our system forms a pair of dynamic blocks staring off an anchor across ontologies. We improve our memory management to cope large scale ontology alignment effectively. Our algorithm has shorter run time than that of the previous year. It takes less memory and even less time as well to align large ontologies. We participate in the benchmark datasets, all four tasks of anatomy track, conference and directory as well. We also take limited participation in the instance matching track. We participate only in the IIMB benchmark track of instance matching track.

1.1 State, purpose, general statement

The purpose of our Anchor-Flood algorithm [8] is basically ontology matching. However, we use our algorithm in patent mining system to classify a research abstract in terms of International Patent Classification (IPC). Containing mostly general terminologies in an abstract leads classification to a formidable task. Automatic extracted taxonomy of related terms available in an abstract is aligned with the

taxonomy of IPC ontology with our algorithm successfully.
Furthermore, we use our algorithm to integrate the multimedia resources represented by MPEG-7 [5] ontologies [11]. We have achieved good performance with effective results in the field of multimedia resource integration [7].
To be specific, we describe our Anchor-Flood algorithm, instance matching algorithm and their results against OAEI 2009 datasets here.

1.2 Specific techniques used

We have two parts of our system. One is the ontology schema matching Anchor-Flood algorithm to align concepts and properties of a pair of ontologies. Another is the instance matching approach which essentially uses our Anchor-Flood algorithm. We implement our system in Java. We create our own memory model of ontology by the ARP triple parser of Jena module.

1.2.1 Ontology Schema Matching

As a part of preprocessing, our system parses ontologies into our own developed memory model by using ARP triple parser of Jena. We also normalize the lexical description of ontology entities.

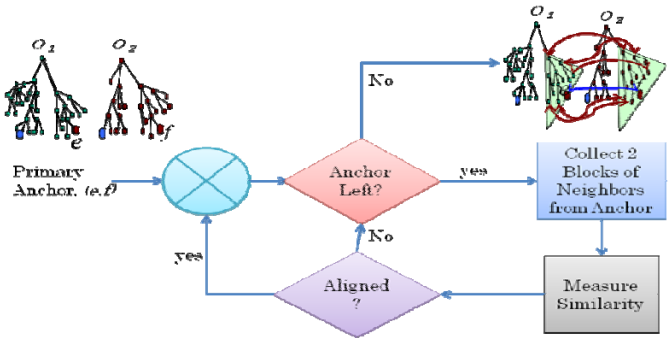

Fig.1. Ontology schema matching Anchor-Flood algorithm

Our schema matching algorithm starts off an anchor. It has a complex process of collecting small blocks of concepts and related properties dynamically by considering super-concept, sub-concept, siblings and few other neighbors from the anchor point. The size of blocks affect the running time adversely. Therefore, we incorporate semantic similarity considering intrinsic Information Content (IC) for building blocks of neighboring concepts from anchor-concepts.
Local alignment process aligns concepts and their related properties based on lexical information [2, 10, and 12], and structural relations [1, 3, 4]. Retrieved aligned pairs are considered as anchors for further processing. The process is repeated until there is no more aligned pair to be processed. Hence, it burst out with a pair of aligned fragment of the ontologies, giving the taste of segmentation [9]. Multiple anchors from different part of ontologies confirm a fair collection of aligned pairs as a whole.

1.2.2 Ontology Instance Matching

In an ontology, neither a concept nor an instance comprises its full specification in its name or URI alone. Therefore we consider the semantically linked information that includes linked concepts, properties and their values and other instances as well. They all together make an information cloud to specify the meaning of that particular instance. We refer this collective information of association as *Semantic Link cloud.* The degree of certainty is proportional to the number of semantic link associated to a particular instance by means of property values and other instances. First, pair of TBox is aligned with our Anchor-Flood algorithm. Then, we check the alignment of the type of an instance to any concept of the neighbors of the type of another instance across ABox. We measure the structural similarity among the elements available in a pair of clouds to produce instance alignment. The instance matching algorithm is depicted in Fig. 2 and in Fig. 3.

Alg. *InstanceMatch* (ABox ab_1, ABox ab_2, Alignment A)

1. for each $ins_i \in ab_1$
2. $\quad cloud_i = makeCloud(ins_i, ab_1)$
3. $\quad$ for each $ins_j \in ab_2$
4. $\quad\quad cloud_j = makeCloud(ins_j, ab_2)$
5. $\quad\quad$ if $\exists a(c_1, c_2) \in A | c_1 \in Block(ins_1.type) \wedge c_2 \in Block(ins_2.type)$
6. $\quad\quad\quad$ if $Sim_{struct}(cloud_i, cloud_j) \geq \delta$
7. $\quad\quad\quad\quad imatch = imatch \cup makeAlign(ins_i, ins_j)$

Fig. 2 Pseudo code of instance matching algorithm

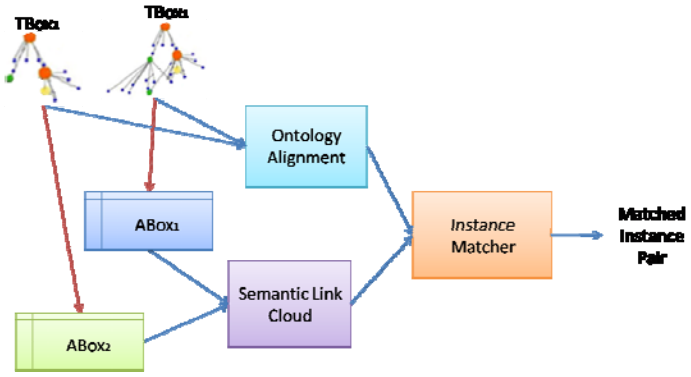

Fig. 3 The basic block diagram of our instance matching approach

1.3 Adaptations made for the evaluation

The Anchor-Flood algorithm needs an anchor to start off. Therefore, we use a tiny program module for extracting probable aligned pairs as anchors. It uses lexical information and some statistical information to extract a small number of aligned pairs from different part of ontologies. The program is essentially smaller, simpler

and faster. We also removed the subsumption module of our algorithm to keep it faster.

1.4 Link to the system and parameters file

The version of Anchor-Flood for OAEI-2009 can be downloaded from our website: http://www.kde.ics.tut.ac.jp/~hanif/res/2009/anchor_flood.zip. The parameter file is also included in the anchor_flood.zip file. I recommend readers to read the readme.txt file first. The file includes the necessary description and parameters as well in brief.

1.5 Link to the set of provided alignments (in align format)

The results for OAEI-2008 are available at our website:
http://www.kde.ics.tut.ac.jp/~hanif/res/2009/aflood.zip.

2 Results

In this section, we describe the results of Anchor-Flood algorithm against the benchmark, anatomy, directory and conferences ontologies and the IIMB instance matching benchmark provided by the OAEI 2009 campaign.

2.1 benchmark

On the basis of the nature, we can divide the benchmark dataset into five groups: #101-104, #201-210, #221-247, #248-266 and #301-304. We describe the performance of our Anchor-Flood algorithm over each of the groups below:
#101-104. Table 1 shows the perfect precision and recall in this group.
#201-210. We improve our results in this group compared to last year results as we improve our structural similarity measure.
#221-247. Our algorithm produces good precision and recall as the previous year.
#248-266. This is the most difficult group for our algorithm. However, we improve our result compared to the last year.
#301-304 Our algorithm produce almost similar result as the previous year.

Table 1. Average results against the ontology benchmarks

Datasets	Prec.	Rec.	F-Measure
101-104	1.00	1.00	1.00
201-210	0.99	0.97	0.98
221-247	0.99	1.00	0.99
248-266	0.96	0.73	0.83
301-304	0.88	0.77	0.82

2.2 anatomy

In this test, the real world cases of anatomy for Adult Mouse Anatomy (2744 classes) and NCI Thesaurus (3304 classes) for human anatomy are included. These are relatively large compared to benchmark ontologies. We participated all of the tasks of this track this year. Our algorithm produces similar result four times faster than the last year. We participate in task#2, task#3 and task#4 for the first time. We find that the run time changes adversely if the block size increases.

Table 2. Our algorithm collects alignment from anatomy ontologies quickly.

Task	**Description**	**Required Time (sec)**	**|Total Alignment|**
Task#1	Default Optimization	14.5	1149
Task#2	Increase precision	221	1228
Task#3	Increase recall	278	1416
Rask#4	Extended reference mapping	282	1460

2.3 directory & Conference Tracks

We also participate directory and conference track this year for the first time.

2.4 Instance Matching: IIMB Benchmarks

On the basis of transformation, the benchmark dataset is divided into four groups: 001-010, 011-019, 020-029 and 030-037. Table 3 shows the precision and recall for each of the groups. However, the detailed results are displayed in Annex section of this paper.

Table 3. Instance matching results against IIMB benchmarks

Datasets	**Trnasformation**	**Prec.**	**Rec.**	**F-Measure**
001-010	Value transformations	0.99	0.99	0.991
011-019	Structural transformations	0.72	0.79	0.751
020-029	Logical transformations	1.00	0.96	0.981
030-037	Several combinations of the previous transformations	0.75	0.82	0.786

3 General comments

In this section, we want to comment on the results of our system and the way to improve it.

3.1 Comments on the results

The main strength of our schema matching system is the way of minimizing the comparisons between entities, which leads enhancement in running time. In instance matching, our system shows its strength over value and logical transformations.
The weak points are: our system ignores some distantly placed aligned pairs in ontology alignment system. In instance matching, we have still rooms to work in structural transformation.

3.2 Discussions on the way to improve the proposed system

It has still rooms of improving alignments strengthening the semantic and structural analysis and adding background knowledge. We also want to incorporate complex alignment like subsumption and 1:n alignments. In instance matching, we want to improve our system against structural transformation.

4 Conclusion

Ontology matching is very important for attaining interoperability as the core of every semantic application is ontology. We implemented faster algorithm to align specific interrelated parts across ontologies, which gives the flavor of segmentation. The anatomical ontology matching shows the effectiveness of our Anchor-Flood algorithm. Our instance matching algorithm also shows its strength in value and logical transformations. In structural transformation our algorithm is also effective in spite of challenging transformation. We improved our previous Anchor-Flood algorithm in several perceptions to retrieve ontology alignment. Furthermore, we improve the versatility of using it in different applications including instance matching, patent classification and multimedia resource integration.

References

1. **Bouquet, P., Serafini, L. and Zanobini, S.:** *Semantic Coordination: A New Approach and an Application.* Proceedings of the 2nd International Semantic Web Conference (ISWC2003), Sanibel Island, Florida, USA (2003) pp. 130-145.
2. **Euzenat, J. and Valtchev, P.:** *Similarity-based Ontology Alignment in OWL-Lite.* Proceedings of the 16th European Conference on Artificial Intelligence (ECAI2004), Valencia, Spain (2004) pp. 333-337.
3. **Giunchiglia, F. and Shaiko, P.:** *Semantic Matching,* The Knowledge Engineering Review, Cambridge Univ Press, Vol. 18(3), 2004, pp. 265-280.
4. **Giunchiglia, F., Shvaiko, P. and Yatskevich, M.:** *S-Match: an Algorithm and an Implementation of Semantic Matching.* Proceedings of the 1st European Semantic Web Symposium (ESWS2004), Heraklion, Greece, (2004) pp. 61-75.
5. **Nack, F. and Lindsay, A.T.:** *Everything you wanted to know about MPEG-7 (Part I).* IEEE Multimedia, Vol. 6(3), 1999, pp. 65--77.

6. **Resnik, P.:** *Using information content to evaluate semantic similarity in a taxonomy.* Proceedings of the 14th International Joint Conference on Artificial Intelligence. Montreal, Canada (1995) pp. 448-453.
7. **Seddiqui, M.H. and Aono, M.:** *MPEG-7 based Multimedia Information Integration through Instance Matching.* Berkeley, IEEE International Conference on Semantic Computing, CA, USA (2009) pp. 618-623.
8. **Seddiqui, M.H. and Aono, M.:** An Efficient and Scalable Algorithm for Segmented Alignment of Ontologies of Arbitrary Size. *Web Semantics: Science, Services and Agents on the World Wide Web* (2008), doi:10.1016/j.websem.2009.09.001.
9. **Seidenberg, J. and Rector, A.:** *Web Ontology Segmentation: Analysis, Classification and Use.* Proceedings of the 15th International Conference on World Wide Web (WWW2006), Edinburgh, Scotland (2006) pp. 13-22.
10. **Stoilos, G., Stamou, G. and Kollias, S.:** *A String Metric for Ontology Alignment.* Proceedings of the 4th International Semantic Web Conference (ISWC2005), Galway, Ireland (2005) pp. 623-637.
11. **Troncy, R., et al.:** *Mpeg-7 based Multimedia Ontologies: Interoperability Support or Interoperability Issue.* Proceedings of the 1st International Workshop on Multimedia Annotation and Retrieval enabled by Shared Ontologies (MAReSO), Genova, Italy (2007).
12. **Winkler, W.E.:** *The State of Record Linkage and Current Research Problems.* Technical Report, Statistical Research Division, U.S. Census Bureau, Washington, USA (1999).

Annex

Schema Matching: Ontology Benchmark

Dataset	Prec.	Rec.	F-Meas.	Time (ms)
101	1.00	1.00	1.00	518
103	1.00	1.00	1.00	155
104	1.00	1.00	1.00	157
201	0.95	0.90	0.92	160
201-2	1.00	1.00	1.00	165
201-4	1.00	1.00	1.00	155
201-6	0.98	0.98	0.98	154
201-8	0.98	0.97	0.97	177
202	1.00	0.97	0.98	125
202-2	1.00	1.00	1.00	121
202-4	1.00	1.00	1.00	141
202-6	1.00	1.00	1.00	128
202-8	1.00	0.98	0.99	135
203	1.00	1.00	1.00	131
204	0.99	0.98	0.98	139
205	0.92	0.85	0.88	156
206	1.00	0.97	0.98	171
207	1.00	0.97	0.98	156
208	0.99	0.98	0.98	120
209	0.93	0.82	0.87	143
210	1.00	0.96	0.98	132
221	1.00	1.00	1.00	125
222	1.00	1.00	1.00	151
223	1.00	1.00	1.00	138
224	1.00	1.00	1.00	112
225	1.00	1.00	1.00	134
228	1.00	1.00	1.00	73
230	0.94	1.00	0.97	119
231	1.00	1.00	1.00	127
232	1.00	1.00	1.00	119
233	1.00	1.00	1.00	66
236	1.00	1.00	1.00	62
237	1.00	1.00	1.00	117
238	1.00	1.00	1.00	132
239	0.97	1.00	0.98	74
240	0.94	0.97	0.95	77
241	1.00	1.00	1.00	71
246	0.97	1.00	0.98	64
247	0.94	0.97	0.95	79
248	1.00	0.61	0.76	108
248-2	1.00	0.97	0.98	123
248-4	1.00	0.96	0.98	110
248-6	1.00	0.90	0.95	107
248-8	1.00	0.78	0.88	108
249	1.00	0.78	0.88	103
249-2	1.00	1.00	1.00	105
249-4	1.00	1.00	1.00	106
249-6	1.00	1.00	1.00	122
249-8	1.00	0.98	0.99	65
250	1.00	1.00	1.00	63
250-2	1.00	1.00	1.00	63
250-4	1.00	1.00	1.00	79
250-6	1.00	1.00	1.00	66
250-8	1.00	0.97	0.98	119
251	1.00	0.37	0.54	131
251-2	1.00	0.92	0.96	136
251-4	0.98	0.85	0.91	136

251-6	0.97	0.74	0.84	128
251-8	1.00	0.62	0.77	136
252	0.97	0.29	0.45	129
252-2	0.98	0.92	0.95	132
252-4	0.98	0.92	0.95	120
252-6	0.98	0.92	0.95	119
252-8	0.98	0.92	0.95	132
253	1.00	0.01	0.02	92
253-2	1.00	0.97	0.98	97
253-4	1.00	0.93	0.96	95
253-6	1.00	0.87	0.93	96
253-8	1.00	0.72	0.84	108
254	1.00	0.27	0.43	55
254-2	1.00	0.82	0.90	59
254-4	1.00	0.70	0.82	59
254-6	1.00	0.61	0.76	58
254-8	1.00	0.42	0.59	68
257	1.00	0.85	0.92	55
257-2	1.00	0.97	0.98	60
257-4	1.00	1.00	1.00	59
257-6	1.00	1.00	1.00	59
257-8	0.91	0.91	0.91	57
258	1.00	0.09	0.17	109
258-2	1.00	0.92	0.96	107
258-4	0.97	0.81	0.88	121
258-6	0.97	0.70	0.81	116
258-8	1.00	0.56	0.72	124
259	0.86	0.06	0.11	96
259-2	0.98	0.92	0.95	108
259-4	0.98	0.92	0.95	120
259-6	0.98	0.92	0.95	107
259-8	0.98	0.92	0.95	105
260	0.92	0.41	0.57	82
260-2	0.96	0.90	0.93	63
260-4	0.96	0.79	0.87	78
260-6	0.95	0.69	0.80	66
260-8	0.94	0.59	0.72	82
261	0.92	0.33	0.49	67
261-2	0.97	0.88	0.92	68
261-4	0.97	0.88	0.92	68
261-6	0.97	0.88	0.92	80
261-8	0.97	0.88	0.92	67
262	0.00	0.00	NaN	54
262-2	1.00	0.79	0.88	53
262-4	1.00	0.61	0.76	56
262-6	1.00	0.42	0.59	53
262-8	1.00	0.21	0.35	66
265	0.80	0.14	0.24	54
266	0.50	0.06	0.11	57
301	0.86	0.75	0.80	95
302	0.93	0.58	0.71	92
303	0.77	0.77	0.77	117
304	0.95	0.96	0.95	93

Instance Matching: IIMB Benchmarks

Data	**Prec**	**Rec**	**F-Meas.**	**Time (sec)**
001	1.00	1.00	1.00	94
002	1.00	1.00	1.00	103
003	1.00	1.00	1.00	125
004	1.00	1.00	1.00	83
005	1.00	0.95	0.97	99
006	1.00	1.00	1.00	105
007	1.00	1.00	1.00	157
008	1.00	0.99	0.99	64
009	1.00	1.00	1.00	97
010	1.00	0.94	0.97	96
011	0.82	0.62	0.71	68
012	1.00	0.96	0.98	91
013	1.00	0.99	0.99	45
014	0.89	0.66	0.76	36
015	0.99	0.95	0.97	65
016	0.93	0.80	0.86	46
017	0.67	0.40	0.50	27
018	0.77	0.54	0.63	51
019	0.88	0.55	0.68	26
020	1.00	1.00	1.00	93
021	1.00	1.00	1.00	93
022	1.00	1.00	1.00	93
023	1.00	1.00	1.00	93
024	1.00	1.00	1.00	93
025	1.00	1.00	1.00	93
026	1.00	1.00	1.00	93
027	1.00	1.00	1.00	93
028	0.46	1.00	0.63	93
029	1.00	1.00	1.00	93
030	0.82	0.57	0.67	65
031	0.83	0.60	0.70	26
032	1.00	0.95	0.97	99
033	1.00	0.93	0.96	95
034	1.00	0.98	0.99	76
035	0.93	0.69	0.79	36
036	0.99	0.86	0.92	95
037	0.83	0.44	0.58	30