Alignment-based Partitioning of Large-scale Ontologies

Fayçal Hamdi, Brigitte Safar, Chantal Reynaud and Haïfa Zargayouna

Abstract Ontology alignment is an important task for information integration systems that can make different resources, described by various and heterogeneous ontologies, interoperate. However very large ontologies have been built in some domains such as medicine or agronomy and the challenge now lays in scaling up alignment techniques that often perform complex tasks. In this paper, we propose two partitioning methods which have been designed to take the alignment objective into account in the partitioning process as soon as possible. These methods transform the two ontologies to be aligned into two sets of blocks of a limited size. Furthermore, the elements of the two ontologies that might be aligned are grouped in a minimal set of blocks and the comparison is then enacted upon these blocks. Results of experiments performed by the two methods on various pairs of ontologies are promising.

1 Introduction

The fast development of internet technology engendered a growing interest in research on sharing and integrating sources in a distributed environment. The Semantic Web [1] offers possibility for software agents to exploit representations of the sources contents. Ontologies have been recognised as an essential component for knowledge sharing and the realisation of the Semantic Web vision. By defining the concepts of specific domains, they can both describe the content of the sources to be integrated and explain the vocabulary used by users in requests. However, it is very

Fayçal Hamdi, Brigitte Safar and Chantal Reynaud
LRI, Université Paris-Sud, Bât. G, INRIA Saclay - Île-de-France, 2-4 rue Jacques Monod, F-91893 Orsay, France e-mail: firstname.lastname@lri.fr

Haïfa Zargayouna
LIPN, Université Paris 13 - CNRS UMR 7030, 99 av. J.B. Clément, 93440 Villetaneuse, France. e-mail: haifa.zargayouna@lipn.univ-paris13.fr

unlikely that a single ontology covering whole distributed systems can be developed. In practice, ontologies used in different systems are developed independently by different communities. Thus, if knowledge and data must be shared, it is essential to establish semantic correspondences between the ontologies of these systems. The task of alignment (search for mappings between concepts) is thus particularly important for integration systems because it allows several heterogeneous systems, which each has its own ontology, to be used jointly. This research subject has resulted in numerous works [12].

The current techniques of alignment are usually based upon similarity measures between pairs of concepts, one from each ontology. These measures are mostly based on the lexical characteristics of the concept labels and/or on the structural characteristics of the ontologies [10], [9], [11] which involve comparing the description of each concept in one ontology with the description of all concepts in the other. Theses techniques are often tested on small ontologies (a few hundred concepts). When ontologies are very large, for example in Agronomy or Medicine, ontologies include tens of thousands of concepts (AGROVOC[1] : 28 439, NALT [2] : 42 326, NCI[3] : 27 652), and the effectiveness of the automatic alignment methods decreases considerably in terms of execution time, size of memory used or accuracy of resulting mappings. A possible solution to this problem is to try to reduce the number of concepts given to the alignment tool, and for this purpose to partition both ontologies to be aligned into several blocks, so the processed blocks have a reasonable size.

We propose two methods of partitioning guided by the task of alignment. These methods are partially inspired by co-clustering techniques, which consist in exploiting, besides the information expressed by the relations between the concepts within one ontology, the information which corresponds to the inter-concept relations which can exist across both ontologies. The fact that concepts of both ontologies can have exactly the same label and can be connected by a relation of equivalence is an example of relation easy to calculate even on large ontologies, and which we will use to our benefit. Our methods will thus start by identifying, with a similarity measure strict and inexpensive to calculate, the couples of concepts from the ontologies which have identical labels, and will base itself on these concepts, called *anchors*, to make the partitions.

The rest of the paper is organized as follow. In the next section, we present the context of our work and some related works in the domain of partitioning, and then we detail more precisely the algorithm of partitioning PBM used by the alignment system FALCON [5, 6] on which we based our propositions. In Section 3 we detail our two methods of partitioning. In Section 4 we present and analyse the experimental results which demonstrate the relevance of these methods. Finally, we conclude and we give some perspectives in section 5.

[1] http://www4.fao.org/agrovoc/

[2] http://agclass.nal.usda.gov/agt/

[3] http://www.mindswap.org/2003/CancerOntology/

2 Context and state of the art

The problem which we are interested in is the scalability of the ontologies alignment methods.

2.1 Context

An ontology corresponds to a description of an application domain in terms of concepts characterized by attributes and connected by relations. The ontology alignment task consists in generating in the most automatic way relations between the concepts of two ontologies. The types of these matching relations can be equivalence relations *isEq*, subsumption relations *isA* or proximity relations *isClose*. When the ontologies are very large, the efficiency of automatic alignment methods decreases considerably. The solution which we consider is to limit the size of the input sets of concepts given to the alignment tool. In order to do this we partition both ontologies to be aligned into several blocks, so only blocks of reasonable size are processed. The two sets of blocks obtained will then be aligned in pairs, each pair made from a block from each set, and the objective consists in minimizing the number of pairs to be aligned.

Our contribution is the elaboration of a partitioning algorithm adapted to the task of alignment and usable on all ontologies containing a hierarchy of labelled concepts. It only exploits the relations of subsumption between concepts and their labels. Partitioning a set E consists in finding disjoined subsets $E_1, E_2, ..., E_n$, of elements semantically close i.e. connected by an important number of relations. The realisation of this objective consists in maximizing the relations within a subset and in minimizing the relations between different subsets.

The quality of the result of a partitioning will be appreciated according to the following criteria:

- The size of generated blocks: blocks must be smaller than the maximum number of elements that the alignment tool can handle.
- The number of generated blocks: this number must be as low as possible to limit the number of pairs of blocks to be aligned.
- The degree of blocks cohesiveness: a block will have a strong cohesiveness if the structural relations are strong inside the block and weak outside. This degree groups the elements which can possibly match into a minimal number of blocks and thus reduces the number of comparisons to be made.

The fact that the partitioning algorithm only uses, in a light treatment, the subsumption relationships between the concepts allows very large ontologies to be partitioned. It is thus a scalable approach.

2.2 State of the art

In real application domains the ontologies are becoming increasingly large and many works as [13], [2] and [5] are interested in ontology partitioning.

Thus the work reported in [13] aims at decomposing ontologies into independent sub-blocks (or *islands*), in order to facilitate different operations, such as maintenance, visualisation, validation or reasoning, on the ontologies. This method is not adapted to our problem because the process of blocks generation imposes a constraint on the minimal size of the generated blocks which is not appropriate for alignment. In addition, it builds many small blocks, which has a negative impact on the final step of alignment. Works presented in the Modular Ontology conference [7] focus specifically on the problems of reasoning and seek to build modules centred on coherent sub-themes and self-sufficient reasoning. For example, the work of [2] are very representative of this issue, and guarantee that all the concepts connected by links of subsumption are grouped together into a single module. For ontologies containing tens of thousands of subsumption relations (as AGROVOC and NALT) this type of constraint can lead to the creation of blocks with badly distributed sizes, unusable for alignment. However, this technique is used by the MOM system to align (theoretically) large ontologies, but the tests presented in [15] are only applied on ontologies of less than 700 concepts.

In our knowledge, only PBM Partition-based Block Matching system, integrated into the ontology matching system FALCON [5, 6] has been created in order to align ontologies, but we will see that its method of decomposition does not take completely into account all the constraints imposed by this context, in particular the fact of working simultaneously with two ontologies.

2.3 The PBM Method

The PBM[4] method proposed in [5] consists in decomposing into blocks each ontology independently, by the clustering ROCK algorithm [3], and then by measuring the proximity of each block of an ontology with every block of the other ontology in order to align only the pairs of concepts belonging to the closest blocks.[5] To make the partition, while ROCK considers that the links between the concepts all have the same value, PBM introduced the concept of *weighted links* mainly based on a structural similarity between concepts.

[4] The description of the PBM algorithm we present here is based upon the implementation available at: http://iws.seu.edu.cn/projects/matching/

[5] The blocks are built as sets of concepts, and an intermediate step, used by PBM but which we are not describing here, is needed to retransform them into ontology fragments.

2.3.1 Weighted links

Let c_i, c_j be two concepts of the same ontology O, c_{ij} their smallest common ancestor and $depthOf(c)$ the distance in number of edges between the concept c and the root of O. PBM measures the value of the link connecting c_i and c_j called $Link_s(c_i, c_j)$ using the measure of Wu and Palmer [16]:

$$Link_s(c_i, c_j) = \frac{2 * depthOf(c_{ij})}{depthOf(c_i) + depthOf(c_j)}$$

To prevent high calculation cost of similarity between each pair of concept, PBM considers only the concepts which satisfy the following relation:

$$\left| depthOf(c_i) - depthOf(c_j) \right| \leq 1$$

2.3.2 Partitioning Algorithm

For partitioning two ontologies in blocks, PBM is based on two essential notions: the *cohesiveness* within a block and the *coupling* between two separate blocks. Cohesiveness is a measure of the weight of all links connecting concepts belonging to the same block, and coupling is a measure of the weight of all links connecting concepts of two different blocks. Theses notions are calculated with the same measure called *goodness*:

$$goodness(B_i, B_j) = \frac{\sum_{c_i \in B_i, c_j \in B_j} Link_s(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)}$$

$Cohesiveness(B_i) = goodness(B_i, B_i)$, $Coupling(B_i, B_j) = goodness(B_i, B_j)$ where $B_i \neq B_j$.

Given an ontology O, the algorithm takes for input the set B of n blocks to partition, where each block is initially reduced to a single concept of O, and a k desired number of output blocks or a parameter ε_1 limiting the maximum number of concepts in each block. It first initialises the cohesiveness value of each block as well as the coupling value. For each iteration, the algorithm chooses the block which has the maximum cohesiveness value and the block which has the maximum coupling value with the first block. It replaces theses two blocks by the result of their fusion and updates coupling values of all blocks by taking this new block into account. The algorithm stops when it reaches the desired number of blocks or when all blocks have reached the size limit or there is no block whose cohesiveness is larger than zero.

2.3.3 Identification of pairs of blocks to align

Once the separate partitioning of both ontologies is achieved, the evaluation of the proximity between blocks is based on *anchors*, i.e. from previously known mappings between the terms of both ontologies, defined by string comparison techniques or defined by an expert. The more two blocks contain common anchors, the more they are considered close.

Let k (resp. k') be the number of blocks generated by the partitioning of an ontology O (resp. O') and B_i (resp. B'_j) be one of these blocks. Let the function $anchors(B_u, B'_v)$ that calculates the number of anchors shared by two blocks B_u and B'_v and let $\sum_{v=1}^{k'} anchors(B_i, B'_v)$ be the number of anchors contained in a block B_i. The *Proximity* relation between two blocks B_i and B'_j is defined as follows:

$$Proximity(B_i, B'_j) = \frac{2.anchors(B_i, B'_j)}{\sum_{u=1}^{k} anchors(B_u, B'_j) + \sum_{v=1}^{k'} anchors(B_i, B'_v)}$$

The aligned pairs of blocks are all the pairs whose proximity is greater than a given threshold $\varepsilon_2 \in [0,1]$. A block may be aligned with several blocks of the other ontology or with none, depending on the value chosen for this threshold.

Example We applied the PBM algorithm, available online, to two toy ontologies to vizualise its behaviour and facilitate later comparison with our own methods.

Fig.1 shows these two ontologies after a partitioning achieved with the control variable representing the maximum size of merged blocks fixed at 3 concepts, i.e. a block exceeding this size cannot be merged. So the blocks thus generated contain at most 6 concepts, and as O_S has 13 concepts, this value ascertained we would get at least 3 blocks.

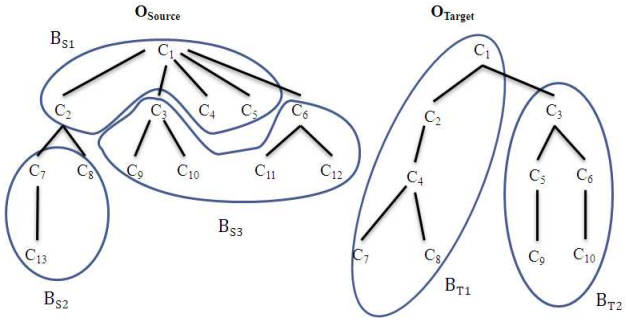

Fig. 1 The blocks built by PBM

Fig.2 shows the anchors which are supposed to be shared between both ontologies. Block B_{S1} contains 2 anchors, one of which is shared with B_{T1} while the other

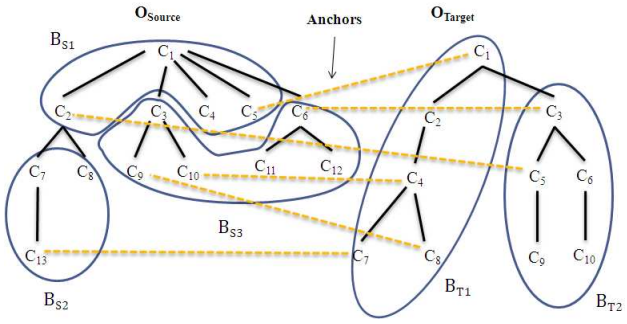

Fig. 2 Anchors identification

is shared with B_{T2}. Block B_{S2} only contains one anchor, shared with B_{T1}. Block B_{S3} contains 3 anchors two of which are shared with B_{T1} while the third is shared with B_{T2}.

Shared-anchors based proximity calculations must be performed on every possible pairs of blocks (6 pairs in this case). As the (B_{S1}, B_{T1}) pair only has one common anchor while the blocks have, in order, 2 and 4 common anchors, Proximity(B_{S1}, B_{T1}) = 0.33. The other results are: Proximity(B_{S1}, B_{T2}) = 0.5, Proximity(B_{S2}, B_{T1}) = 0.4, Proximity(B_{S2}, B_{T2}) = 0, Proximity(B_{S3}, B_{T1}) = 0.57, Proximity(B_{S3}, B_{T2}) = 0.4.

The number of pairs actually aligned varies according to the threshold value. Lowering the threshold multiplies alignments and the chances one has of finding mappings, but also increases runtime costs. With a high threshold, less time is spent aligning far blocks but this can result in the loss of potential mappings. When the threshold is set to 0.4, the (B_{S1}, B_{T1}) pair is not aligned and the common anchor is not discovered in the mappings. When the threshold is set to 0.33, all the anchors are discovered in the mappings, but every possible pair of blocks, except anchorless (B_{S2}, B_{T1}) has to be aligned.

This method allows PBM to decompose very large ontologies. Nevertheless this decomposition is made a priori, without taking into account the objective of alignment, because it is applied to each ontology independently from the other. Partitioning is done blindly, some anchors may not be in the blocks finally aligned and the resulting alignment does not necessarily include all desired mappings. Finally, the calculation of the relevant blocks to be aligned is expensive (in processing time).

Despite these criticisms, the decomposition algorithm PBM is, among all existing partitioning algorithms, the most adapted to the task of alignment since it allows control of the maximum size of the generated blocks.

We propose two methods that reuse this algorithm by modifying how it generates blocks. Our idea is to consider, as soon as possible during the partitioning, all the existing data relative to the alignment between the concepts of both ontologies and to try to simulate, at least in the second method, co-clustering.

3 Alignment Oriented Partitioning Methods

To take into account as soon as possible the objective of alignment, our methods are going to lean on two facts: on one hand the couples of concepts stemming from both ontologies which have exactly the same label and can be connected by a relation of equivalence and on the other hand the possible structural asymmetry of the two ontologies to be aligned.

Even on large ontologies, it is possible to identify, with a similarity measure strict and inexpensive to calculate, concepts which have a label in common across ontologies. As in PBM, we call these couples concept *anchors* but we will use them to generate partitions.

The structural asymmetry of both ontologies is used to order their partitioning and to choose the method to do it: if one ontology is more structured than the other, it will be easier to decompose it into blocks with a strong internal cohesiveness and its decomposition can serve as a guide for the decomposition of the other ontology. In what follows, the most structured ontology will be called the *target*, O_T and the less structured, the *source*, O_S. The first method that we propose, called PAP (*Partition, Anchor, Partition*), consists in beginning by decomposing the target O_T, then by using identified anchors, to force the partitioning of O_S to follow the pattern of O_T. In so doing, this first method partially breaks the structure of the source O_S. This is not a problem when the source is poorly structured.

However, if O_S is well-structued, the PAP method is inadequate and we suggest another partitioning method, called APP (*Anchor, Partition, Partition*) which follows more closely the structure of both ontologies. The APP method partitions O_T by favoring the fusion of blocks sharing anchors with O_S, and partitions O_S by favoring the fusion of blocks sharing anchors with the same block generated from O_T.

3.1 The PAP *Method*

The PAP method consists in beginning by decomposing the target O_T, then by forcing the partitioning of O_S to follow the pattern of O_T. To achieve this, the method identifies for each block B_{Ti} from O_T all the anchors belonging to it. Each of these sets will constitute the kernel or *center* CB_{Si} of a future block B_{Si} to be generated from the source O_S. The alignment of the pairs of blocks allows to find, in the final step of alignment, all the equivalence relations between anchors. The PAP method consists of four steps besides the calculation of anchors:

Partition O_T into several blocks B_{Ti}. Partitioning is done according to the PBM algorithm.

Identify the centers CB_{Si} of the future blocks of O_S. The centers of O_S are determined from two criteria: the pairs of anchors identified between O_S and O_T, and the blocks B_{Ti} built from the target ontology O_T.

Let the function $Anchor(E, E')$, whose arguments E and E' are each an ontology or a block, returns all concepts of E which have the same label as the concepts of E'. For each block B_{Ti} built in the previous step, the centers of future blocks of O_S are calculated as follows:

$$CB_{Si} = Anchor(O_S, B_{Ti})$$

Partition the source O_S around the centers CB_{Si}. After identifying the centers of the future blocks O_S, we apply the PBM algorithm with the following difference. Instead of inputing the set of the m concepts as m blocks, each reduced to a single concept, we introduce the n centers identified in the previous step, as distinct blocks but with several concepts and other concepts of O_S that have no equivalents in O_T, each one in an individual block. The cohesiveness of the blocks representing the centers O_S is initialized with the maximum value.

Identifying the pairs of blocks to align. Each block B_{Si} built from a center is aligned with the corresponding block B_{Ti}. The algorithm can lead to the constitution of B_{Sj} blocks containing no anchors and which, in the current state of our implementation, are not taken into account in the matching process. The treatment of these blocks without anchors is a perspective of this work, still under study.

Example On the toy example presented earlier, fig. 3 shows first the decomposition of O_T achieved by the PBM algorithm, then the identification of the centers CB_{Si} of the future blocks of O_S. These will be built from the blocks generated for target B_{T1} and B_{T2}. $CB_{S1} = \{c_5, c_9, c_{10}, c_{13}\}$ and $CB_{S2} = \{c_2, c_6\}$.

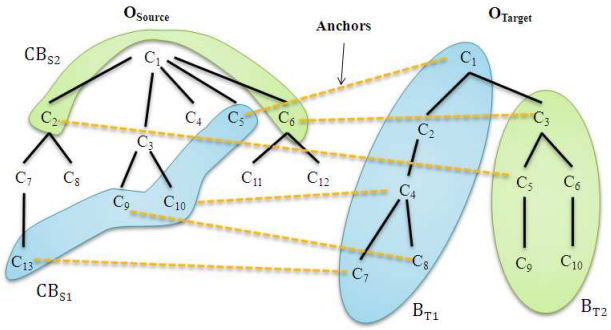

Fig. 3 The centers CB_{Si} identified from B_{Ti}

Fig. 4 shows O_S blocks resulting from the partition of O_S around the centers. The test on the maximum size of constructed blocks being performed according to *PAP* method after the initial block grouping, block B_{S1} becomes larger than the size limit so no other block can be grouped with it. It is the same for B_{S2}. Thus this partioning

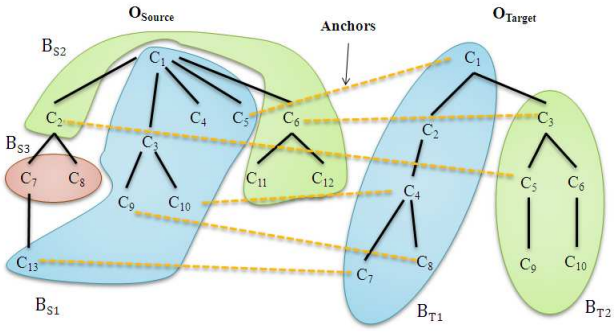

Fig. 4 Partition of O_S around the centers CB_{Si} identified in precedent step

reveals an anchorless block, B_{S3}, which will not be aligned. The aligned pairs are (B_{S1}, B_{T1}) and (B_{S2}, B_{T2}), immediately identifiable by construction.

3.2 The APP *Method*

The idea of this method is to partition both ontologies at the same time, i.e. to co-cluster. The problem is that we cannot really treat these ontologies in parallel because of their large size. To simulate the parallelism, we partition the target ontology by favoring the fusion of blocks sharing anchors with the source, and we partition the source by favoring the fusion of blocks sharing anchors with the same block generated from the target. Then we take into account the equivalence relations between ontologies identified since the partitioning of O_T, which makes the search for resembling blocks easier and improves alignment results. Unlike the PBM algorithm and our PAP method, this partitioning method is alignment-oriented: it simplifies the subsequent task of aligning both ontologies. The APP method has three steps:

Generate O_T blocks. To generate blocks of the target O_T, we use the PBM algorithm by modifying the definition of the *goodness* measure to take into account the equivalence relations between both ontologies. We add a coefficient representing the proportion of anchors that are shared in a block B_j of O_T. The more anchors a block contains, the more this coefficient increases its cohesiveness or its coupling value respectively to other blocks. As a result, during the generation of blocks, the choice of the block that has the maximum value of cohesiveness or coupling depends not only upon relations between concepts inside or outside the blocks of O_T, but also upon the anchors shared with O_S.

Let $\alpha \in [0,1]$, B_i and B_j be two blocks of O_T, $|Anchor(B_j, O_S)|$ represents the number of anchors in B_j and $|Anchor(O_T, O_S)|$ represents the total number of anchors. The *goodness* equation becomes:

$$goodness(B_i, B_j) = \alpha \left(\frac{\sum_{c_i \in B_i, c_j \in B_j} Link_S(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)} \right) + (1-\alpha) \frac{|Anchor(B_j, O_S)|}{|Anchor(O_T, O_S)|}$$

Generate O_S blocks. Again we modify the *goodness* measure to take into account at the same time the values of links between O_S concepts, the anchors shared between both ontologies and the blocks built for O_T. Let the block B_i of O_S be the block with the maximum value of cohesiveness and the block B_k of O_T be the block which shares the highest number of anchors with B_i. The new calculation of *goodness* favors the fusion of B_i with B_j, which contains the highest number of anchors in common with B_k. This gathers in a single source block the anchors shared with one target block.

Let $\alpha \in [0,1]$, B_i and B_j be two distinct blocks of O_S. Let B_k be the block of O_T which shares the highest number of anchors with B_i. The *goodness* equation becomes:

$$goodness(B_i, B_j) = \alpha \left(\frac{\sum_{c_i \in B_i, c_j \in B_j} Link_S(c_i, c_j)}{sizeOf(B_i) \cdot sizeOf(B_j)} \right) + (1-\alpha) \frac{|Anchor(B_j, B_k)|}{|Anchor(O_T, O_S)|}$$

Identification of blocks pairs. The alignment is done between the blocks sharing the highest number of anchors; a block of O_S can only align itself with a single block of O_T.

Example Fig. 5 and 6 also display results obtained upon our toy example. Fig.5 shows the blocks from O_T built according to the APP method, favoring anchor grouping. Fig. 6 shows the blocks built in O_S, favoring the construction of blocks sharing anchors with these of O_T while taking the structure of O_S into account.

Every source block is only aligned once with the block with which it has the greatest number of common anchors, identified by construction. So we align the pairs (B_{S1}, B_{T1}), (B_{S3}, B_{T1}) and (B_{S2}, B_{T2}).

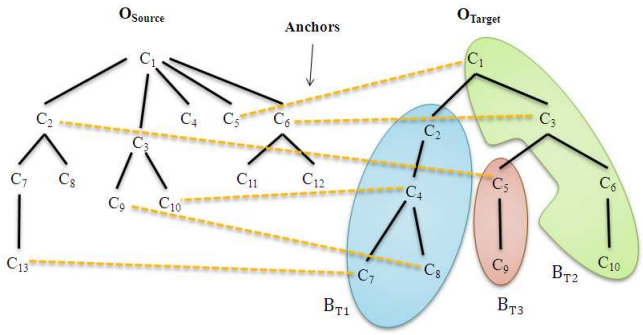

Fig. 5 Built blocks from O_T by the APP method

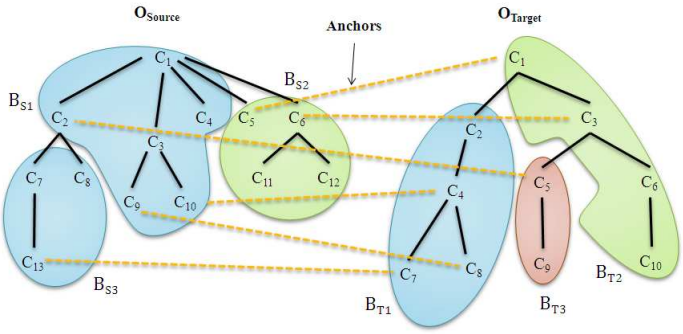

Fig. 6 Built blocks from O_S by the APP method

B_{T3} takes no part in the alignment process because it shares its single common anchor with B_{S1} and B_{S1} has more anchors in common with B_{T1} than with B_{T3}. This results in the loss of an anchor match, (c_2, c_5), but reduces alignment runtime. We can hope that the co-clustering building of the blocks takes inter-ontologies relationships more into account.

4 Experiments

We have implemented the two methods presented previously and experiments were made on various ontologies in order to compare partitioning methods through their suitability for alignment. Generated blocks were aligned by pairs using the alignment software developed within our team, *TaxoMap* [4].

The experiments were first realised on ontologies in the geographical area, supplied by COGIT[6]. These ontology sizes are limited so it is possible to align them directly - without having to partition - and to obtain reference mappings. They are also well known in the team which enabled us to analyse the semantic relevance of the generated blocks. Other experiments were then made on two pairs of large ontologies which our tool fails to align because of scalability problems.

4.1 Experiments on geographic ontologies

Target ontology BDTopo, is composed of 612 concepts related by subsumption links in a hierarchy seven levels deep. *Source* ontology BDCarto includes 505 concepts

[6] The COGIT laboratory (Conception Objet et Généralisation de l'Information Topographique), National Geographical Institute

in a hierarchy of depth 4. The results of the direct alignment carried out without ontologies partitioning are presented in Table 1.

Table 1 Relations identified by aligning BDCarto to BDTopo

Ontologies	Target Size	Source Size	isEq	isClose	isA	Σ
BDTopo-BDCarto	612	505	197	13	95	305

To make the partitions, the maximum size of merger blocks was fixed at 100 concepts, i.e. a block exceeding this size cannot be merged. So the blocks thus generated contain at most 200 concepts. Table 2 lists the number of blocks generated for each ontology.

Table 2 Partitioning of BDTopo and BDCarto with the different methods

		Target Ontology BDTopo			Source Ontology BDCarto		
Methods	Anchors	Generated blocks	Isolated concepts	Largest block	Generated blocks	Isolated concepts	Largest block
PBM	191	5	0	151	25	22	105
PAP	191	5	0	151	10	16	143
APP	191	6	0	123	10	16	153

Target ontology BDTopo is the main ontology for COGIT. It is well constructed, compact and highly structured. The root is only linked to two direct children of depth 1, which are direct parents to a limited number of nodes. It is easy to partition into semantically relevant blocks, whether by the PBM method which is mainly based on the structural relations between concepts, by the PAP method, which uses the PBM algorithm for the partitioning of the target and so gives the same results for it, or by the APP method. Both possible decompositions, consisting of 5 or 6 blocks, are relevant.

On the opposite side, source ontology BDCarto is less structured and much dispersed. The root is linked to almost thirty direct children, and many sub-trees contain no more than about ten elements. Decomposition is more delicate. The PBM algorithm generates a big number of small blocks comprising no more than 5 or 6 concepts, 19 blocks do not contain anchors, and 22 blocks contain only one isolated concept. By using the information on the shared anchors, our methods allow to aggregate to larger blocks more than half of these small blocks and many isolated concepts, while maintaining its semantic consistency. The generated partition, less dispersed, is therefore more understandable for the humans and more efficient for the next phase of blocks alignment.

The choice of the pairs of blocks to align differs according to the method used:

PBM: among the 25 generated blocks only 6 source blocks contain anchors. Theses 6 blocks are aligned with the target blocks for which the ratio of shared anchors

on the sum of anchors present in the two blocks is higher than a given threshold ε, fixed here at 0.1. This threshold is reached by 9 pairs of blocks, 9 alignments are made.

PAP: the 5 source blocks, built starting from the 5 blocks of the target which contain anchors, lead in all to 5 alignments.

APP: the 7 selected pairs are those which maximize the number of shared anchors of the 7 source blocks containing anchors and which each participates only in one alignment.

Table 3 shows the number of mappings we obtain by matching the different pairs of blocks chosen by our alignment tool, *TaxoMap*. The results presented show that even by matching fewer pairs of blocks than in the PBM method, matching blocks generated by our methods give better results in number of identified mappings.

Table 3 Relations identified by the alignment of blocks generated by different methods

Methods	Aligned Pairs	isEq	isClose	isA	Σ	Precision	Recall
PBM	9	118	13	52	183	0.96	0.57
PAP	5	192	10	55	257	0.97	0.81
APP	7	147	11	61	219	0.97	0.69

If we analyse the results[7] of the two classical alignment measures to compare the relevance of the techniques, the precision (the number of correct mappings identified after partition compared to the full number of returned mappings after partition) and the recall (the number of correct mappings identified after partition compared to the number of reference mappings), we see that our methods have a much better recall. Indeed, these methods take into account the equivalence relations between the labels in the partitioning process, which brings together the concepts that have relations between them in blocks which will be considered thereafter as pairs to align, while the PBM method partitions ontologies independently from each other and makes only a posteriori alignment. The PAP method allows in particular, by construction, to find all mappings corresponding to the anchors and thus has a higher recall. We are currently working upon heuristics which could be applied, after the partitioning step, on isolated blocks and which would increase the recall of our methods.

The fact that the different methods have a precision lower than 1. means that all three of them find mappings which had not been identified by the alignment of the unpartitioned ontologies. Although these mappings are here considered to be invalid, they are not necessarily wrong. Indeed, for every source concept, our tool produces a single mapping with one concept of the target ontology, that which it

[7] These results were calculated automatically by the API of alignments evaluation available on the Web, http://oaei.ontologymatching.org/2008/align.html, by providing in reference the file generated by direct alignment by *TaxoMap* without partition.

considers the best, even if several concepts of the target could be matched. If the two concepts involved in a reference mapping are no longer compared because they are divided into non-aligned blocks, another mapping, which will not necessarily be uninteresting, can be found for the source concept. The study of the quality of these new mappings, as well as more advanced analysis of the relative qualities of our two methods, will be carried out in complementary work.

4.2 Experiments on large ontologies

We tested the two different methods on two pairs of large ontologies (Library and FAO). These pairs of ontologies are used as test in the evaluation campaign OAEI (*Ontology Alignment Evaluation Initiative*) in which alignment tools compete each year on ontologies of diverse sizes and domains.

For both tests (Library and FAO), the comparison between our methods and the PBM method is complex because the FALCON system was not a participant to the 2008 OAEI campaign and we did not participate to the FAO test in the 2007 campaign. Furthermore, as the FAO pair of ontologies was not a test case provided by the 2008 campaign, we did not access the reference mappings. Despite of this, we present in this section, two kinds of experiments. First, we provide a comparison between our results and those obtained by the participants having done the Library test in 2008. Second, we use the FAO test to compare the number of blocks generated by our methods and the PBM algorithm.

4.2.1 Library test

The Library set of tests is made of two thesauri, GTT and Brinkman, in Dutch. These two thesauri are used by the National Library of the Netherlands to indexed the books of two large collections. GTT thesaurus contains 35,194 concepts and Brinkman contains 5,221 concepts. Each concept has (exactly) one preferred label, but also synonyms (961 for Brinkman, 14,607 for GTT). The organisers of the test in 2007 showed that both thesauri have similar coverage (2,895 concepts actually have exactly the same label) but differ in granularity and that the thesauri structural information was very poor. GTT (resp. Brinkman) contains only 15,746 (resp 4,572) hierarchical *broader links*. Its structure being particularly poor (it has 19,752 root concepts), GTT thesaurus was considered as the source in our experiments.

As both ontologies are very imbalanced and as the number of retrieved anchors was limited to 3,535, which is not much with respect to the size of the source, we only experimented with the PAP method. We set the maximum size for a block to be grouped to 500.

The PAP method returned 227 blocks for Brinkman, the larger of which had 703 concepts, and 2,041 blocks for GTT, the larger of which had 517 concepts. 16,265 concepts of GTT remained isolated.

Table 4 Partitioning of Brinkman and GTT

		Target Thesaurus Brinkman			Source Thesaurus GTT		
Methods	Anchors	Generated blocks	Isolated concepts	Largest block	Generated blocks	Isolated concepts	Largest block
PAP	3 535	227	0	703	2 041	16 265	517

As over 1,800 blocks of GTT contained no anchors, we only aligned 212 pairs and identified 3,217 matches, only 1,872 of which were equivalence relations (ExactMatch).

Table 5 Relations identified by the alignment of blocks generated by the PAP method

Methods	Aligned Pairs	isEq	isGeneral	isClose	isA	Σ	Precision	Recall
PAP	212	1 872	40	274	1 031	3 217	0.88	0.41

The reason why so few equivalence relations were returned, with respect to the number of identified anchors, is that both thesauri contain a large number of synonyms. We identified 3,535 anchors while only 2,895 concepts were supposed to have the same label. This means that at least 640 anchors concern source concepts, among which at least 2 labels are considered equivalent to 2 other target labels, which are not necessarily associated to the same concept. The problem here is that if a source concept is anchored to 2 distinct target concepts, at best both these target concepts belong to the same block, and the target concept is linked by an ExactMatch relation to only one of these concepts. In the worst case, the 2 target concepts belong to distinct blocks and the PAP method does not know to which block the source concept should be linked. So the PAP method sets it to become an isolated concept.

Table 6 Results of the systems taking part in the Library test

Participant	ExactMatch	Precision	Coverage
DSSim	2 930	0.93	0.68
TAXOMAP	1 872	0.88	0.41
Lily	2 797	0.53	0.37

Even though several anchors have disappeared, precision and coverage evaluated only upon equivalence relations (ExactMatch) by the organisers of the test, and presented in Table 6, place our system TAXOMAP running the PAP method, in rea-

sonable position. Among the other two participants, DSSim [8] [8] got better results than us but Lily [9] [14] did worse.

4.2.2 FAO test

The FAO set of tests (2007) comprises two ontologies : AGROVOC and NALT, which consist respectively of 28 439 and 42 326 concepts. AGROVOC is a multilingual ontology built by FAO (Food and Agriculture Organization). It covers the fields of agriculture, forestry, fisheries, environment and food. NALT is the thesaurus of NAL (National Agricultural Library) on the same subject.

The most important ontology, NALT, is used as the target and AGROVOC is used as the source. The maximum size of merger blocks is fixed at 2 000 concepts.

Table 7 Partitioning of AGROVOC and NALT

		Target Ontology NALT			Source Ontology AGROVOC		
Methods	Anchors	Generated blocks	Isolated concepts	Largest block	Generated blocks	Isolated concepts	Largest block
PBM	14 787	47	4	3 356	318	492	2 830
PAP	14 787	47	4	3 356	252	199	2 939
APP	14 787	47	4	3 118	95	199	3 534

Despite there are no reference mappings which make possible to analyse the quality of produced alignments, we nethertheless present the results of partitionning in Table 7 because they seem us relevant. Table 7 shows that in this experiment, as in the previous one, partitioning according to our methods minimised the number of isolated concepts, and in particular according to the APP method, minimised the number of generated blocks, leading to partitions that might be less dispersed.

Among the 47 blocks built for O_T according to the PAP method, only 42 contain anchors. So 210 of the 252 blocks built for O_S take no part in the alignment process which matches 42 pairs of blocks. The APP method matches 25 pairs of blocks.

5 Conclusion

As current tools for ontology alignment lose their effectiveness on large ontologies, the objective of this work was to study the techniques of ontology partitioning oriented towards the alignment task.

[8] The authors of DSSim say they partition the ontologies but do not explain how.

[9] The authors of Lily say they process the ontologies according to a method which is not based upon partitioning but they refer to a yet unpublished article.

The two methods we propose take the PBM algorithm for ontology partitioning, developed for the alignment system, but instead of applying the algorithm, as PBM, successively and independently on each ontology, we try to take into account as soon as possible in the partitioning process the context of the alignment task.

Our methods are applied on two ontologies simultaneously, and use alignment-related data. These alignment-related data are easy to extract, even from large ontologies. They include pairs of concepts, one concept from each ontology, which have the same label, and structural information on the ontologies to align.

The PAP method is well suited for ontologies of a dissymmetrical structure. It starts by decomposing the best structured ontology and then forces the decomposition of the second ontology following the same pattern. The APP method can be applied when both ontologies are well structured. It favors the generation of blocks of concepts, which are related, from one ontology to the other, by equivalence links.

The fact that the partitioning algorithms only use data easy to extract, in a light treatment, allows very large ontologies to be partitioned. It is thus a scalable approach.

Our methods were tested on different ontology couples. The results presented here show that they can build partitions less dispersed by limiting the number of generated blocks and isolated concepts. For the experiment where we have reference mappings, we have been able to see that our partitions lost fewer mappings.

We are currently working upon heuristics which could be applied, after the partitioning step, on isolated blocks and which would increase the recall of our methods.

We currently continue the experiments to analyse the qualities of our two methods when both ontologies are heavily unbalanced (in terms of size and structure) or when the number of concepts with identical labels is limited.

6 Acknowledgement

This research was supported by the French National Research Agency (ANR), through the GeOnto project ANR-O7-MDCO-005 on Creation, Comparison and Exploitation of Heterogeneous Geographic Ontologies (http://geonto.lri.fr/) and through the WebContent project ANR RNTL.

References

1. Berners-Lee, T., J. Hendler and O. Lassila (2001). The semantic web. In The Scientific American 284(5), pp. 34-43.
2. Grau, B. C., B. Parsia, E. Sirin and A. Kalyanpur (2005). Automatic partitioning of owl ontologie using e-connections. In Proceedings of 18th International Workshop on Description Logics, Edinburgh, UK.
3. Guha, S., R. Rastogi and K. Shim (2000). ROCK : A robust clustering algorithm for categorical attributes. Information Systems 25, pp. 345-366.

4. Hamdi, F., H. Zargayouna, B. Safar and C. Reynaud (2008). TaxoMap in the OAEI 2008 Alignment Contest. In Workshop on Ontology Matching, located at the International Semantic Web Conference - ISWC.
5. Hu,W., Y. Zhao and Y. Qu (2006). Partition-based block matching of large class hierarchies. In Proceedings of Asian Semantic Web Conference- ASWC, pp. 72-83.
6. Hu,W., Y. Qu, and G. Cheng (2008). Matching large ontologies: A divide-and-conquer approach. Data & Knowledge Engineering Revue 67, pp. 140-160.
7. ModularOntology-ISWC, W. (2006). Proceedings of the workshop on Modular Ontologies. located at the International Semantic Web Conference- ISWC.
8. Naggy M., M. Vargas-Vera and P. Stolarski (2008) DSSim Results for OAEI 2008. In Workshop on Ontology Matching.
9. Noy, N. F. and M. A. Musen (2000). PROMPT : Algorithm and tool for automated ontology merging and alignment. In Proceedings of AAAI/IAAI, pp. 450-455.
10. Rahm, E. and P. A. Bernstein (2001). A survey of approaches to automatic schema matching. VLDB Journal 10(4), pp. 334-350.
11. Reynaud, C. and B. Safar (2007). Techniques structurelles d'alignement pour portails web. Revue des Nouvelles Technologies de l'Information, RNTI W-3, Fouille du Web, Cepadués.
12. Shvaiko, P. and J. Euzenat (2005). A survey of schema-based matching approaches. Journal on Data Semantics IV, pp. 146-171.
13. Stuckenschmidt, H. and M. Klein (2004). Structured-based partitioning of large concept hierarchies. In International Semantic Web Conference - ISWC, pp. 289-303.
14. Wang, P. and B. Xu (2008). Lily: Ontologiy Alignment Results for OAEI 2008. In Workshop on Ontology Matching, located at the International Semantic Web Conference- ISWC.
15. Wang Z., Y. Wang, S. Zhang , G. Shen and T. Du, (2006) Matching Large Scale Ontology Effectively. In Proceedings of Asian Semantic Web Conference - ASWC. pp. 99-105.
16. Wu, Z. and M. Palmer (1994). Verb semantics and lexical selection. In 32nd. Annual Meeting of the Association for Computational Linguistics, pp. 133-138.