Automatic Fusion of knowledge stored in Ontologies

Alma-Delia Cuevas[1] and Adolfo Guzman-Arenas[2]

[1] Dirección General de Educación Superior Tecnológica Av. Patriotismo 711 Edif. B Col. San Juan Mixcoac Del. Benito Juárez C.P. 03730 México D.F.almadeliacuevas@gmail.com

[2] Centro de Investigación en Computación, Instituto Politecnico Nacional, Mexico City, MEXICO. a.guzman@acm.org

Abstract. A person adds new knowledge to his/her mind, taking into account new information, additional details, better precision, synonyms, homonyms, redundancies, apparent contradictions, and inconsistencies between what he/she knows and new knowledge that he/she acquires. This way, he/she incrementally acquires information keeping it at all times consistent. This information can be represented by Ontologies. In contrast to human approach, algorithms of Ontologies fusion lack these features, merely being computer-aided editors where a person solves the details and inconsistencies. This article presents a method for Ontology Merging (OM), its algorithm and implementation to fuse or join two ontologies (obtained from Web documents) in an automatic fashion (without human intervention), producing a third ontology, and taking into account the inconsistencies, contradictions, and redundancies between both ontologies, thus delivering a result close to reality. The repeated use of OM allows acquisition of much information about the same topic.

Keywords. Ontology, Artificial Intelligence, Knowledge representation, Semantic Web, Ontology fusion.

1 Introduction

A person accrues information across his/her life by adding new knowledge (concepts, relations, typical values…) to the information (s)he has in his/her mind (in his/her ontology or knowledge structure), identifying redundancies, new information, small and large contradictions, antonymous and

synonyms among others cases. Nowadays, computers could do the same process (joining knowledge which comes from two different ontologies) through an editor [§1.2] that makes preliminary alignment of concepts, and lets a person finally decide. It is a computer-aided fusion. The problem to solve is how to mechanize that fusion.

1.1 Outline of paper

This article presents an algorithm (OM, Ontology Merging) and its implementation to fuse two ontologies in an automatic form, obtaining a third one, taking into account inconsistencies, synonymous, precision rates, contradictions and discrepancies between them, in such manner that the result is close to reality. The result, a knowledge ontology, can become quite useful if it is the fusion of many general and specific ontologies. The article is based on a Ph. D. thesis [6].
Current similar works appear in §1.2. Some examples of the results produced by OM are shown in §3 and §4. Future plans for enriching OM are in §6. One of these plans is the creation of a system [17] that takes text documents and converts them to ontologies, after which OM will join them in order to create larger and larger ontologies containing detailed knowledge about a given topic area. This large ontology (yet to be built) can be used, for instance, by a system [2] that answers non-trivial questions.
OM resembles CyC [19], in that both pursue to produce a large common knowledge ontology. CYC envisions to manually building such large ontology, while OM does it mechanically.

1.2 Related works

In distinction to the manual creation of ontologies [i.e., 19], OM performs such creation by getting pieces of knowledge (small ontologies) and joining them carefully (verifying inconsistencies, joining synonyms, etc) without human intervention. Also, OM is not tuned to special or specific knowledge areas; it can be used to merge ontologies in any knowledge area (perhaps after modifying its initial knowledge basis, §2.2).
Current encyclopedias, such as Wikipedia or Encyclopedia Britannica, contain knowledge held in written documents, inserted by hand into the encyclopedia and related to each other by hand, too. Inconsistency and contradiction among documents is controlled by restricting who publishes (inserts) the documents, and by a "final authority" (the Editor). In contrast to this, ontologies can be produced electronically, by repeated use of OM. Relations are placed (among the *concepts* of the ontology) by OM, who also resolves discrepancies and disagreements.

Current methods of fusing ontologies are computer-aided, not fully automated processes. PROMPT [9], Chimaera [16], OntoMerge [7], IF-Map and ISI [Internet reference 1] require that a user solve problems presented during the fusion. FCA-Merge [20] uses Formal Concept Analysis for the representation of ontologies, forcing them to be mutually consistent. But the majority of ontologies in Web present inconsistencies when compared to other ontologies. A recent fuser is HCONE-merge [15], which uses the semantic data base WordNet [8] as intermediary information for the fusion, requiring less user support. It is an important advance in computer-aided ontology fusion.

Ontologies facilitate the search of the Web for the right concepts. If each element of knowledge in Internet were translated (located, placed) into a (piece inside an) ontology, this way of structuring knowledge would be more efficient for computer search. OM will find its work easier, too. For example, a comprehensive ontology about Albert Einstein's life is obtained from 50 biographies, and now these extensive descriptions have to be hand built. It is our hope that, with OM's help, these large ontologies could be machine built.

1.3 Information Management

Internet contains huge amounts of information in billions of documents located in Web sites, text libraries, doorway services, music blogs, photographic maps, etc. When we access them through searchers (Google, CiteSeer,…), only a small part of the available information is recovered, because the search is performed in a syntactic form (through labels, words and phrases); that is, through lexicographic comparisons. More over, the answer is a large list of documents that does not always contain the information sought. In addition, the desired information must be deduced or extracted by manually processing each of the documents (that is, by *reading* them) by a person, perhaps adding knowledge from several of them.

If a large structure of knowledge about a given topic could be found in the Web (as an ontology, for instance; Cf.§2.1), then an alternative form to obtain a desired complex information would be to query (by an "intelligent" query) such ontology. Sure this will be less painful than the actual procedure. To achieve this, two tools must be constructed: one to smartly join small ontologies into larger ones; another to pose intelligent queries to a complex or large ontology.

This article is focused on the first of these tools: an automatic knowledge fuser. This fusion must consider not only the syntax of the word and phrases (contained in the description of the concepts forming the ontologies), but their semantics, too (the neighboring words or concepts, synonyms, homonyms, and so on).

2. OM elements

The behavior of OM is better perceived through examples. To this end, documents were taken from Web sites. From them, ontologies were manually created, but their fusion was entirely constructed by OM.

2.1 Ontology definition

In Philosophy, Ontology or "being theory", is the study of being, what is it, how it is and how it becomes [Internet reference 2]. In Computer Science, Ontology is a data structure, a notation used to share and reuse knowledge between Artificial Intelligence systems [10]. Once a common vocabulary is defined, knowledge can be represented by an ontology. Then, an ontology is a set of definitions, classes, relations, functions and other objects of speech [6].
Mathematically, it does not exist a satisfactory theory which characterizes formally the ontology definition. However, some attempts have appeared [14]. From the Logics viewpoint, an ontology is a pair

$$O = (Ç, R)$$

Where:
Ç is a set of nodes (representing concepts), some of which are relations.
R is a set of restrictions, of the form (r; c_1; c_2;…; c_k) between the relation r and concepts c_1 until c_k (lower c is used to refer to each concepts of set Ç, while a semicolon separates members in the restrictions). For example: (cut; `scissor`; `sheet`), (print; `printer`; `document`; `ink`). In these examples, the concepts that are relations too, are cut and print. The restrictions are not limited to have two members besides the relation. Therefore, an ontology is a hypergraph with Ç the set of nodes and R the set of hyper-relations.

Ontology definition languages are defined in order to (manually) code the information in the form of nodes, relations, and word definitions. This is necessary if these ontologies are to be shared. These languages' purpose is to express the required Semantics. OM uses a special notation (a language based on XML) to represent ontologies, not explained here.

Restrictions could also be used to represent "behavior" of concepts (such as logical restrictions among several concepts, or how the heart works); for this to be achieved, we must be able to (1) use the OM notation to represent events, passing of time, and causal behavior, (2) the OM notation should represent logical restrictions, too, and (3) tell OM how to *fuse* two of these restrictions (that is, two of these behaviors) into a third one. Work on these issues is hard (Cf. §6, point 2).

Figure 1 shows a hierarchy drawn like a semantic net. The nodes represent concepts, while the links represent relations among them

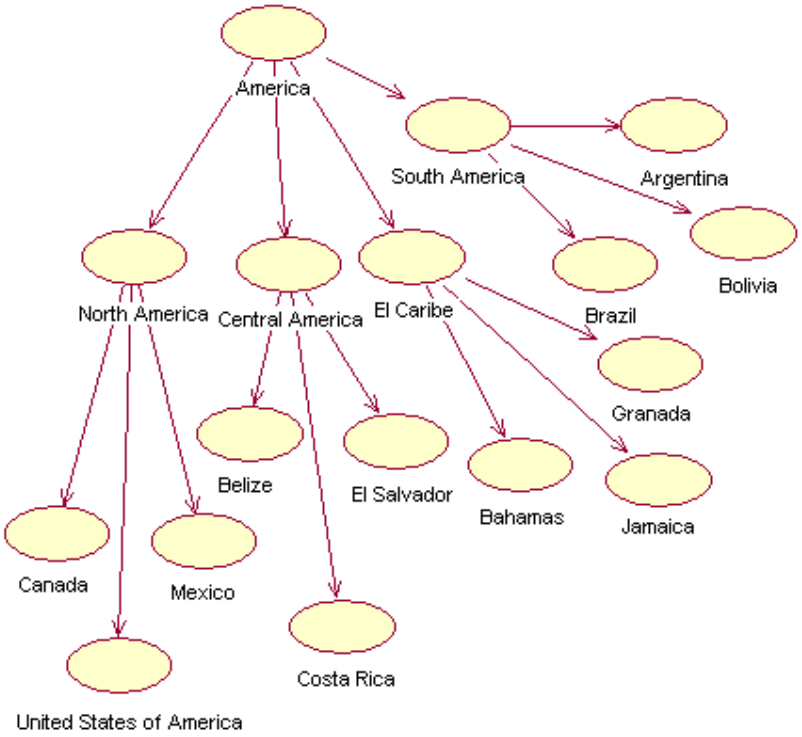

Figure 1. The ontology of the American Continent (called "America", not to be confused with USA, a country). Only some countries are shown. As we will see in §8, this ontology, because it contains only "part of," "member of" and "subset of" relations, can also be regarded as a *hierarchy*

2.2 Initial knowledge used by OM

OM is supported by some initial (built-in) knowledge bases and resources. These are:

1. - Articles and linking words like (*in*, *the*, *to*, *this*, *and*, *or*, etc.), that are ignored in the name or description of the concept.

2. - Words that change or reject concepts in the relation's name, such as: *except*, *without*. For example: *Poppy without Petiole*. This means that the concept Petiole does not form part of the concept Poppy.
3. - Hierarchies of concepts. A hierarchy is a tree of concepts where each node is a concept or a set; if it is a set, then its subsets are a partition of it. The hierarchy represents a taxonomy of related terms. It is used to compute the *confusion* (§8.1). We exploit it to detect synonyms and "false inconsistencies" due to degree of detail.

2.3 Ontology fusion using OM

OM was developed at the Centro de Investigación en Computación - Instituto Politécnico Nacional (CIC, or Center for Computing Research, National Polytechnic Institute), as a product of a doctoral thesis. OM is *strong* because it can join ontologies with inconsistencies (See examples in Table 1). In the process, two ontologies A and B are fused to form a third ontology C. In general we have:

$C = A \cup \{c_B' , r_B \mid c_B' , r_B = ext(r_A, r_B) \ \forall \ c_A \in A \}$

The resulting ontology C is the original ontology A plus some concepts and relations of B that the function *ext* gets.
Where:
c_A is a concept in ontology A, r_A are the relations of c_A present in A; r_B are relations of c_B that exist in B; c_B is the most similar concept *cms* (Cf. §8.2) in B to c_A; and $c_B' \in B$ is explained below.
$\cup$ means ontology joining. It is a carefully joining, somewhat different to set union.
ext(r_A, r_B) is the algorithm that completes the relations r_B which are not in A with those c_B' (which are in B) that do not contradict knowledge from A. That is, for each node $c_A \in A$, all its relations in A are retained in C, and only *some* relations in B of c_B (the concept most similar in B to c_A) are added to C, as well as their "target" concept c_B'. For instance, if the restriction (r_B c_B c_B') = (religion; `Juárez`; `catholic`) is in B, and it does not contradict knowledge from A, then restriction (r_A c_A c_B') = (religion; `Benito Juárez`; `catholic`) is added to r_A in C. [Here we assume that c_A = `Benito Juárez` and its most similar concept in B is c_B = `Juárez`]. This will become clear in the examples below.

Due to the application of *ext* to each concept c_A of A, the OM algorithm takes from B the additional knowledge not present in A and adds it to C. This extraction must be carefully, so as not to introduce inconsistencies, mistakes or redundant information in C. The algorithm *ext* is large, and it is partially explained in §3.2 to §3.6.

3 How OM fuses ontologies

3.1. General description

In order to fuse ontologies A and B into the result C, OM performs the following steps:
1. Copy ontology A to C.
2. Starting from root concept c_{Root} in C, and progressing depth-first:
3. Look in B for its most similar concept c_B (using COM, explained in §8.2). The most similar concept is also known as *cms*.
4. If there is a *cms* in B, new relations of that *cms* in B can be added to C, as well as new concepts, as follows:
4.A. Subsets become partitions (§3.2), if appropriate;
4.B. Redundant relations are verified and rejected (not copied to C), explanation and example in §3.3;
4.C. Synonyms are verified and properly fused (§3.4);
4.D. Homonyms are detected and handled separately (§3.5);
4.E. Partitions from B not in A are added to C (§3.6), when suitable;
4.F. Some inconsistencies are detected and solved (§8.1), using Confusion theory.
5. If there is not a *cms*, then take the next concept c_C depth-first and go back to step 3.
If in step 4 inconsistencies are not solved, then the relation prevailing in A is conserved in C (the conflicting relation in B is discarded).

OM is supported by two important recent developments, briefly explained in the Appendix:
- Confusion Theory (§8.1), that obtains the *confusion* (a number) when concept *r* is used instead of concept *s*. We use it to properly handle redundancies (for instance, "Juárez was born in Mexico" versus "Juárez was born in Guelatao"). Finally, if the inconsistency can not be solved, then OM prefers the knowledge of A. In presence of confusions, OM gives more importance to knowledge acquired earlier. It could be desirable for OM to produce a symmetric fusion; nevertheless, we believe that human learning also has this property (Cf. point 7 of §5.3).
- Comparer of Mixed Ontologies COM (§8.2), that considers a concept c_A in ontology A and looks for the most similar concept c_B in ontology B.

3.2 Promoting subsets to partitions

In Ontology A (Figure 2), `Hotel Finca Santa Marta` [Internet reference 3] finds its most similar concept in B to be `Finca Santa Marta` [Internet reference 4], which has the partition Hotel Amenities. The members of this

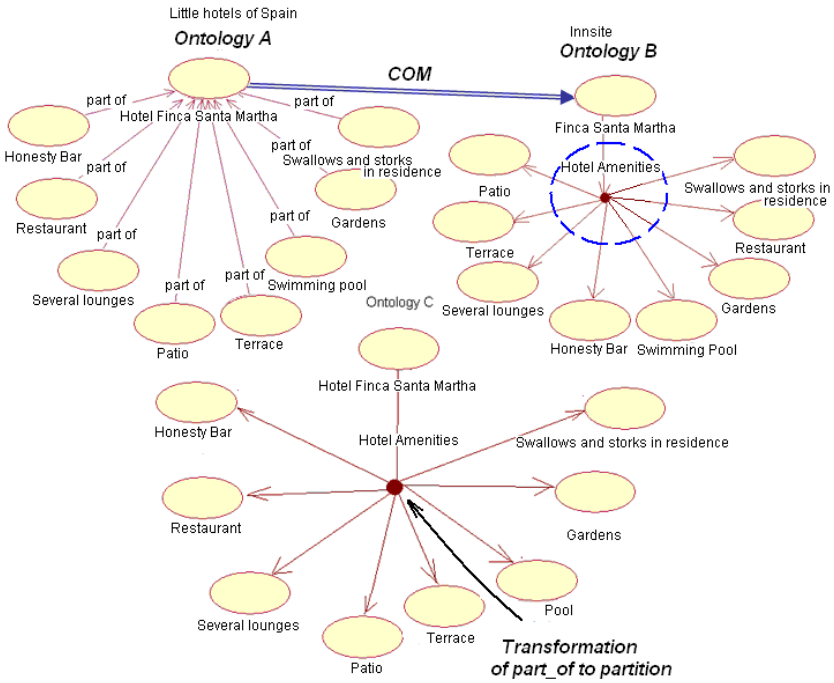

Figure 2. Partitions and Subsets. In B, partition Hotel Amenities of Finca Santa Marta prevails over the subsets indicated by relation "parts of" of A. This is because concepts Honesty Bar, Restaurant, Several lounges, Patio, Terrace, Swimming Pool, Gardens and Swallows and storks in residence are identical to corresponding parts of Hotel Finca Santa Marta in A

partition are precisely the same as the parts (part of) of `Hotel Finca Santa Marta` in A. For this reason, OM selects for the resulting ontology C the relation partition Hotel Amenities, which has a more precise meaning than the corresponding knowledge in A.

3.3 Expunging redundant relations

As an example (see Figure 3), let ontology A have the information of Little hotels of Spain [Internet reference 3] and B have information of InnSite [Internet reference 4]. C shows the results of fusing A and B. The sons of Finca Santa Marta in B are compared with the concept Farm in A, but the sons of Farm are more similar to the sons of Finca Santa Marta (double arrows). During the fusion we have that: Farm1 and Olive oil producing farm 1 are similar, and the same holds for Farm 2 and Olive oil producing Farm 2. In the resulting ontology C we can see that Farm 1 and Farm 2 have two links of the type part of: part of Farm and part of Finca Santa Marta, therefore the relation Farm 1 part of Finca Santa Marta is eliminated from C.

3.4 Identification and merging of synonyms

In this example (Figure 4), a company sells oil (ontology A), while B shows an agent that requests information. The resulting ontology will allow the base knowledge of the company to understand more requests. Synonymy is detected through the word description Ingredient (item, ingredient) in A. Therefore, the Item partition will not be copied into the resulting ontology C, since the Ingredient partition is already in C.

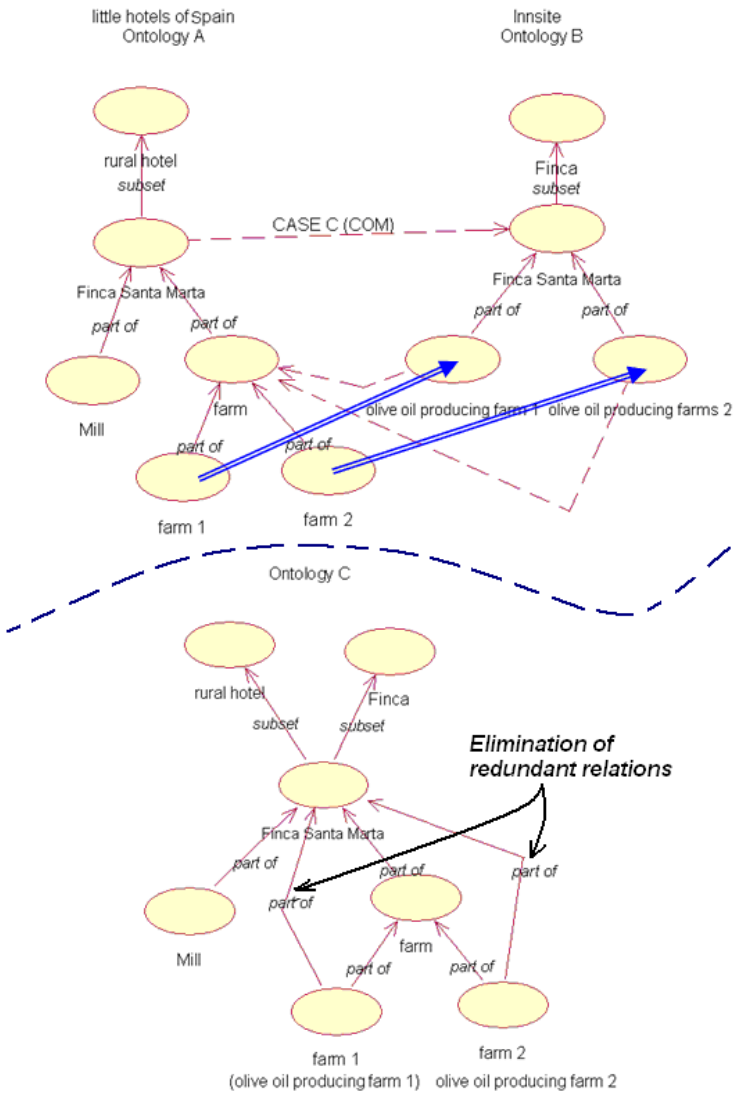

Figure 3. Expunging redundant relations. In C the node `Farm 1` is a part of `Farm` and also a part of `Finca Santa Marta` (redundancy that is eliminated). For that same reason, the restriction (part of; `farm2`; `Finca Santa Marta`) is removed, too

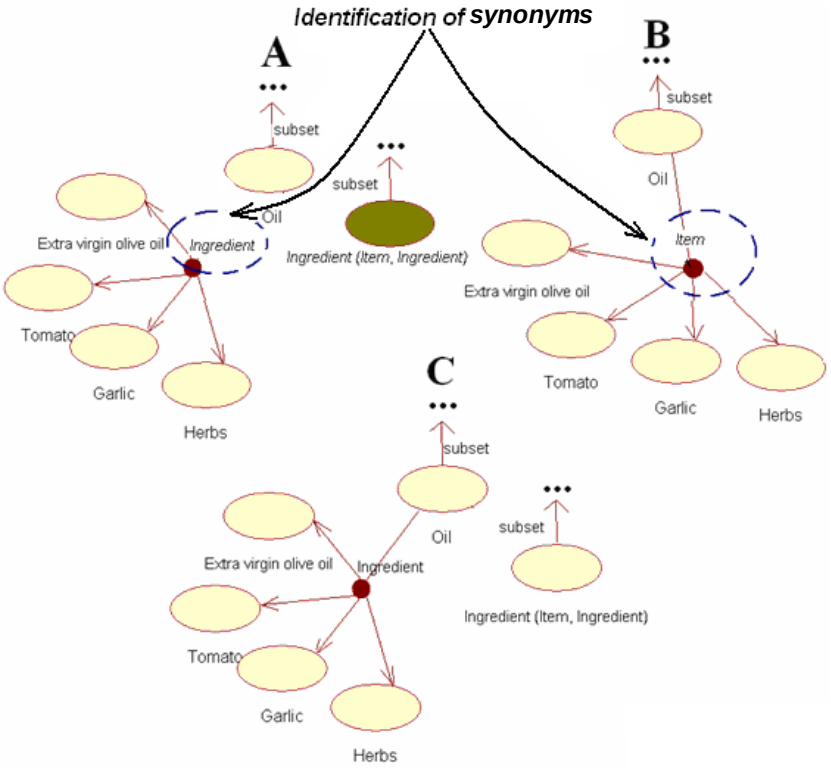

Figure 4. In ontologies A and B, `Ingredient` and `Item` concepts are identified as synonyms. C is the result of fusing A and B

The same process of identification of synonyms concepts is applied to the identification of synonym relations.

3.5 Indentifying and separating homonyms

Concepts `printer` in A (Figure 5) and `printer` in B have the same syntax (same word descriptions), but different semantics. OM finds them different, because their ancestors do not coincide (using COM), neither their relations coincide. If they had descendants, these will be compared. Thus, OM considers them as different concepts; both are added to C in distinct parts of the ontology. Descendants of both concepts are copied to C, if they exist.

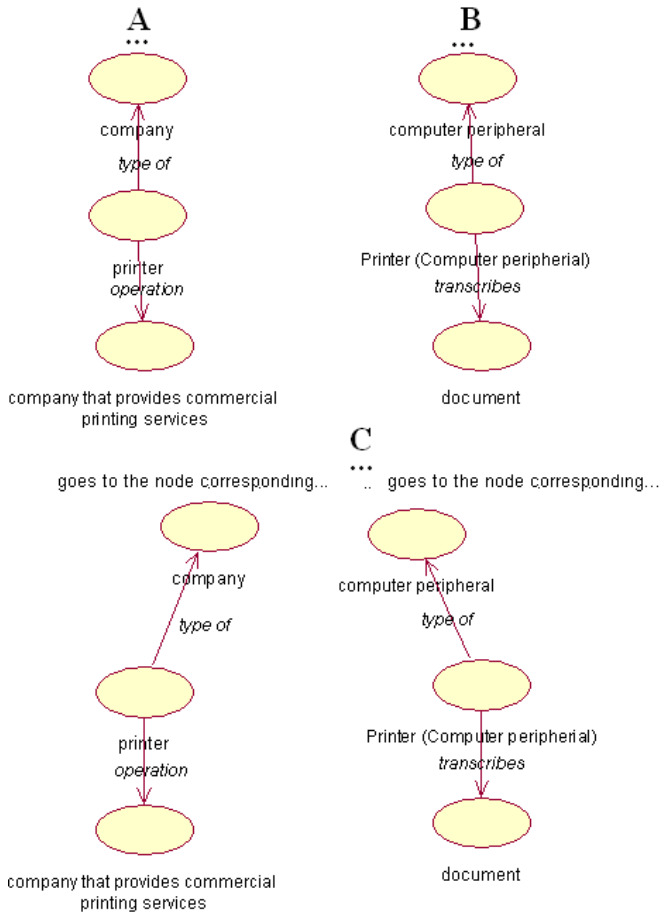

Figure 5. Homonyms. Concepts `printer` in A and `printer` in B are found not to be the same. They both go to C as two different concepts with the same name and same word description.

3.6 Adding a new partition

Figure 6 shows the ontology A with concept `Oil`. Through COM, `Oil` in A finds the most similar concept in B to be `Oil`, too. OM takes the partition Production and will copy it into the resulting ontology C, since it finds no disabling contradictions in A's knowledge. Now the concept `Extra virgin Olive Oil`

has two links: one of them is a part of partition Ingredient and the other one is a subset of `Vegetal Oil`.

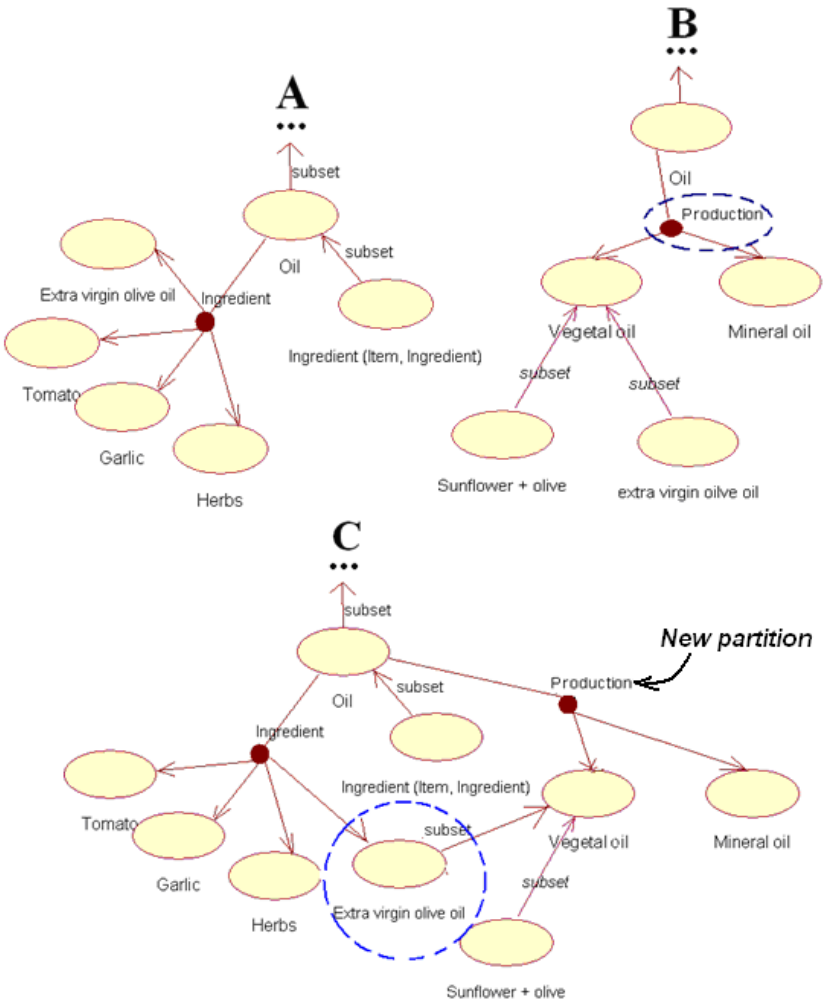

Figure 6. A new partition is learned. The partition Production in B is copied into the resulting ontology C since no contradictions to this knowledge are found in A. The text from which ontology A was drawn said that “extra virgin olive oil is an ingredient”, too; therefore, it was reflected in A

3.7 Comparison with current fusion methods

As briefly outlined in §1.2, current fusion methods are manually driven. They are in fact, computer-aided fusion methods: a human being must do the final decisions and correct mistakes. These fusion methods are just "aligners" that suggest what concepts in ontology A to fuse with which others of B, the user being the final judge. Also, in them the user settles all arising inconsistencies, while OM solves automatically *some* inconsistencies. An approach closer to OM is [15], that uses Wordnet to achieve a less human-intensive fusion. *Thus, OM is the first fully automatic fuser.*

OM was tried on ontologies fused by Protegé and Chimera (using the language OWL), obtaining the same results. These ontologies were "easy" for OM, since their relations are binary, their classes and concepts are "shallow", with few relations (most of their meaning is in their names, Cf. point 1 of §5.3), and they do not use partitions. Also, they were small in general (about 15 nodes).

In addition, OM does not try to eliminate concepts belonging to just one ontology (they are considered "garbage" by some aligners). So, if ontology B in figure 2 says that "ducks and geese" are some "amenities" of Finca Santa Marta, they will be added to the result, in spite the fact that A does not mention them. OM was not designed to have doubts about its inputs (see point 4 of §5.3).

OM resembles current ontology aligners in that both do not know how to handle processes, things that change with time, or that occur in different locations (Cf. points 2 and 3 of §6).

4 Results

Table 1 presents some results. A and B, the ontologies to be fused, were manually constructed from different documents found in Internet. Each pair of ontologies to be joined describes the same topic, for example, both ontologies about `Oaxaca` were built from documents found in different Web sites. Each ontology describes `Oaxaca` in its own way. The ontologies thus obtained were joined entirely by OM. The results produced by OM were verified (compared) against a result obtained by manual fusion of the formant ontologies A and B. The results in general were quite good.

The first column of the Table 1 presents some ontologies and the time that OM took to fuse them. The slowest fusion is *One hundred years of loneliness* because it has more relations and OM verifies carefully elements of each relation, as explained in §2 and §3.1. The second column shows the results of fusing the relations. In the third column we observe the results of fusing the concepts; it also shows the result of the manual fusion compared with OM's fusion. The fourth column shows the numeric error and the last one, the efficiency of OM.

Let T be the total number of relations and concepts in C, produced by manual fusion. Then, the formulas used for error and efficiency are:

Error = (number of relations and concepts wrongly copied to C) / T

Efficiency = (number of relations and concepts wrongly copied to C) / T

5 Discussion

Current ontology fusion methods (excepting HCONE-Merge [15], which uses WordNet to find the meaning of the concepts to be aligned) share two features: (A) the aligning and fusion are done by (syntactically) comparing names and labels of concepts (we call them "the word description of a concept") and their neighborhood; and (B) they resort to human intervention for final acceptance of their suggestions. Thus, they are computer-aided fusers.

OM uses the definition of the concepts, their neighborhood and their characteristics (relations; that is, restrictions in which they participate). These restrictions are verified by a recursive process, since each of them can be also a concept, or point to a concept [in the sense that restriction (religion `Juárez catholic`) can be regarded as an arrow labeled religion from `Juárez` pointing to concept `catholic`], each of them suffers the same OM verification before fusion. This recursive process can be interpreted as a semantic analysis of the concepts. That is, all possible knowledge in A and B about the concept to be merged into C is taken into account. Thus, this version of "semantic analysis" has more possibilities, as OM shows, than the usual syntactic analysis or matching.

5.1 Verifying the fusion

Currently, the fusion is checked manually (§4) against a hand-computed result. It could be machine verified by a program that performs deductions (up to a point, since the "right" answer to a complex question involving context may be subjective) using the question-answering program in [2] (not yet finished). This work is designed to query the result of the fusion of heterogeneous databases; thus, it will have to be adapted to handle ontologies. The adaptation has not been done.

Table 1. Results of using OM in some examples from real cases.

Source ontologies A and B	Relations in the result C	Concepts in the result C	Error	% Effic.
Neurotransmission (A) and Schizophrenia[1] (B) (2 sec.)	79 relations of A were added and fused correctly to 51 relations of B, producing 127 relations in C. The manual method gave 129. Expressed as 79A +51B =127C / 129C. (2 of 129 were not copied).	56 concepts of A were added and fused correctly to 26 of B, getting 77 nodes in C. The manual method gave 79. Expressed as 56A + 26B =77C /79C. (2 of 79 concepts were missing).	0.019	98
Solar System (4 sec.)	45B[i] + 56A[ii] = 59C. All correct.	60B +79A =125C. All correct.	0	100
Continent (3 sec.)	40B[iii] +34A[iv] = 46C.	54B +50A = 66C. All correct.	0	100
Inconsistent ontologies (1 sec.)	3B +4A =7C. There were two inconsistencies (2 relations that were not defined, one on each ontology). C had 1 inconsistence, OM solved the other.	5B +6A = 9C. There were 5 inconsistencies (3 concepts in A and 2 in B wrongly classified). C had the same 5 inconsistencies.	0	100
100 Years of loneliness (10 minutes)	283b + 231A =420C /432C. (12 of 432 were not copied) in C.	126B +90A =141C /148C. (8 of 149 concepts were missed).	0.034	96.5
Oaxaca (5 min.)	43B +61A =96C. All correct.	117B +234A =309C /310C. (1 of 310 concepts was not copied).	0.002	99.7

[1] We thank Paola Nery Ortiz www.geocities.com/paolaneriortiz/ because she manually converted two Spanish documents into ontologies: neurotransmisor: es.wikipedia.org/wiki/Neurotransmisor and esquizofrenia: www.nimh.nih.gov/publicat/spSchizoph3517.cfm Her ontologies were fused automatically by OM.

Results of Table 1 are surprisingly good, since even people are not that good at merging ontologies. This is probably due to additional knowledge used in the human fusion: when a person merges the concepts found in A and B, his or her previous knowledge about dog, cat, or garlic oil influences the result C. Also, (s)he is somewhat at discomfort if the resulting C does not agree with reality, with the real world. OM does not do such a complex job: it just adds knowledge in B to knowledge in A (OM has just a tiny previous knowledge, cf. §2.2), resolves the problems explained in §3, and yields the result. OM does not check that indeed C's knowledge is consistent with the real world: it has no way to do that (nor it is its purpose). A better understanding of OM's behavior will arise as larger and more realistic ontologies are merged.

5.2 Applications of OM

OM has not been applied to large practical problems, yet. The largest of them is *One Hundred Years of Loneliness*, with 432 relations and 148 concepts (Table 1), or Oaxaca with 310 concepts. A problem is that few ontologies are available (Some relief will come from [17]). Many of them are *shallow*, with few relations, built mainly for use by people (not by machine).

Some applications of OM for business interactions are in [5]. An analysis of the semantic web is outside both the scope of OM (it just fuses ontologies) and the scope of this paper.

Be aware that OM can not solve (nor it intends to) the following problems:

A. Find the truth. Find out what is right in real life. It is impossible for OM to find out if the Earth is flat, if light is a wave or a stream of photons… OM only "knows" the knowledge in A and B (plus its tiny internal knowledge basis, §2.2). Its duty is to produce a merge C = A∪B. If both A and B are wrong (they tell lies), but A is consistent with B, so it will be C.

B. To select, from contradictory facts, the right one [Cf. previous point (A)]. Some contradictions are detected as "differing in precision" (§8.1) and solved. Other contradictions are just homonyms, and thus solved (§3.5). Also, given a bag of assertions or facts $\{f_1, f_2 \ldots f_k\}$, the Theory of Inconsistency [13] (not yet used by OM) will find the most likely assertion, in the sense of a "best guess," given the evidence in the bag. But beware that truth is not obtained by majority voting.

C. In what order the ontologies should be fused. If you consider OM as a young intelligence "learning" by amassing bodies of knowledge from many sources, then the order of presentation should be carefully chosen (by the user or "teacher"), starting from good well-constructed general ontologies, gradually adding details and diversity, etc.

5.3 Methodology for the construction of OM

The erection of OM has been influenced by these premises and considerations:
1. We found most of hand-built ontologies available to be "shallow", because they were designed for human use; hence, their meaning was mostly in the names (words) of the nodes. Thus, we preferred to create our own from Web texts. We also used our own OM notation, to have more flexibility in representing new hyper relations, instead of the traditional ontology languages (Cf. §1.2).
2. The ontologies to be fused come from documents containing "assertions considered to be true and important," for instance, pieces that could be used for teaching: how the planets are; how neurotransmission occurs; what is Oaxaca and where it is. Facts that could be taught at school. No opinions, beliefs, poetry, allegories nor metaphors… Also, our current OM does not handle well the passage of time (See §6 Future Work), so it fuses best unchanging facts.
3. A and B are assumed to be self-consistent ontologies. Of course, some knowledge in A may contradict some knowledge in B, and it is the duty of OM to try to solve the inconsistencies.
4. There will be no attempt to see if knowledge in sources A and B are "true" or agree with the real world or the known facts. Thus, A and B are assumed to be "good" or "true" by definition. No sense in discussing their validity. If A says that "garlic oil" is a fine piece of furniture, OM will not challenge that assertion, although B may contradict it. For instance, both ontologies A and B of Figure 2 classify storks and swallows as "hotel amenities."
5. No initial knowledge is used by OM (well, just a tiny bit, see §2.2). Why is this? As more fusions occur, result(s) of previous fusions could be used as previous knowledge by OM. That is, C_1 =OM(A, B) initially. Then, C_2 =OM(C_1, C), then C_3 =OM(C_2, D)… and keep accruing knowledge. Also, we plan to add important sources of knowledge to ease the fusion; see §6 Future Work.
6. No attempt shall be made to see if C is "right" in the real world sense. If A says that the earth is flat, and B too, the result in C "the earth is flat" is considered fine. That is, C shall only be consistent (as much as possible) with A and B.
7. If A and B do not contradict each other, OM(A, B) = OM(B, A). But if they are contradictory, no attempt was made to "fix" the fact that OM(A, B) ≠ OM(B, A). That is, to ascertain which of A or B is "right" and agrees with the real world.

6 Future works

OM could be completed with a pair of additional tools:

- The parser or converter of texts documents into ontologies [17]. It can be regarded as a "pre-processor" to OM. It will (automatically) produce the ontologies that OM fuses. Work in construction.

- The question-answering program, mentioned in §5.1, which will allow us to exploit or to use to a practical purpose that knowledge that OM* join. Work in construction.

In addition, more fusions with larger ontologies need to be performed, to better understand the behavior of OM and to advance its performance.

Furthermore, we have in mind some improvements:

1. – Use of linguistic resources (WordNet, WordMenu, etc). with the purpose of:

1. a. To disambiguate [1]; that is, to map a word into its corresponding concept (according to its context); to give it its right sense. [3] nearly accomplishes this.

1. b. To know more about the relation "part" (part of). That is, to know which concepts are part of other concepts, example, a `boat` is composed of `prow`, `stern`, `starboard`, `port`, `shelter`, `propeller`… (using WordMenu).

2. - Processes. Rules to manage events that happen through time. Some considerations are:

2. a. Some events don't mention when it happened, for example: "*He left Santa Cruz seminary.*"

2. b. Other events mention the time approximately, for example: "He visited the greater part of the Mexican Republic carrying national documents after he had been moved from his position office" or similar form.

2. c. Others are "always" or eternal, for example: "Rosa is Benito's sister," "Benito Juárez is from Oaxaca."

The begin date and an end date of an event can be known (constant) or not known (variables), for example: "*He left Santa Cruz Seminary at **t**.*" Other example: "*He got to the University to study Law at the **u** moment*" where ***t*** and ***u*** are unknown. Later, time relations can be placed among those variables (***t*** > 1812, ***t*** < ***u***). A relevant work is [18].

3. - Similar to (**2**), but to manage events that occur in space (located in space). Geospatial ontologies could be relevant, although I doubt it. Additions **1**, **2**, **3** not yet built.

7 Conclusions

With the advent of OM, we can fuse two ontologies in automatic form, without human intervention. The progress made can be gauged from the quality of the results obtained (Table 1). OM detects and solves some inconsistencies, detects synonyms, homonyms, redundant information and different degrees of detail or precision.

Missing are: a) An analyzer that converts documents from natural language into ontology, and b) a question-answerer (using the resulting ontology of OM) to answer difficult (i.e., "intelligent" or "tough") questions.

Acknowledgments

Work reported was supported in part by grant 43377 Conacyt, and by SNI.

8 Appendix. Work that supports OM

OM exploits, among other things, the following algorithms, previously published. §8.1 introduces Confusion Theory and its algorithm, which yields a number (the *confusion*) when concept *r* is used instead of *s*. §8.2 shows, for ontologies A and B, the algorithm COM, that takes a concept $c_A \in A$ and finds the most similar concept $c_B \in B$.

8.1 Confusion Theory

This is a brief summary of work presented in [11].
Let be *r* and *s* are two values (nodes) in a hierarchy with height[2] *h* and let *r'* be any predecessor of *r*.

The *absolute confusion* that results by using *r* instead of *s* (the expected value) is:
CONF(*r*, *r*) = CONF(*r*, *s*) = 0 when *s* is one of the predecessor of *r*;
CONF(*r*, *s*) = 1 + CONF (*r*, *father_of*(*s*)) in other case.
To compute the value of CONF(*r*, *s*), travel from *r* to *s* in the hierarchy, and count all the *descending* links.
Example. Consider Figure 1 to represent a hierarchy. Then, CONF(`Costa Rica`, `America`) = 0, while CONF(`America`, `Costa Rica`) = 2.
CONF(*r*, *s*) can have values larger than 1, that change by inserting additional nodes (providing more detailed descriptions) in the path between *r* and *s*. To avoid this, we will define the *relative confusion*, conf, by:

Definition. The *relative confusion* (or just *confusion*) conf(r, s) that results by using *r* instead of *s* is:

$$conf(r,s) = \frac{CONF(r,s)}{h}$$

conf(*r*, *s*) is the absolute confusion CONF(*r*, *s*) divided by *h*, the height of the hierarchy.

[2] The height of a tree is the number of links that exist in the way of its root to the most distant leaf. Example: the height of the tree of Figure 1 is 3.

conf finds the confusion caused when a concept *r* is used instead of the intended or correct concept *s*. conf is not a symmetric function. The value of *conf*(*r*, *s*) depends on the hierarchy on which *r* and *s* sit. The hierarchy provides a *context* on which the vicinity of the symbolic values *r* and *s* can be gauged.
Example: if in a hierarchy appears (part of; `San Pablo Guelatao`; `Ixtlan mountains`); (part of; `Ixtlan mountains`; `Oaxaca`); (part of; `Oaxaca`; `Mexico`), then CONF(`San Pablo Guelatao`, `Mexico`) =0; CONF(`Mexico`, `San Pablo Guelatao`) =2. OM deduces that `San Pablo Guelatao` is more specific than `Mexico`. Thus, when considering the fact in ontology A = (was born in; `Benito Juárez`; `San Pablo Guelatao`) and the fact in B = (was born in; `Benito Juárez`; `Mexico`), OM has to decide if an inconsistency has been found, since `Mexico` is not the same as `San Pablo Guelatao` and was born in can only have a single value (somebody can not be born in two different places). OM detects that such contradiction does not exist, since `San Pablo Guelatao` is a more accurate description of the birth place of `Juárez`, is it a "refinement" of `Mexico`. Hence, `San Pablo Guelatao` is kept in C, thus: (was born in; `Benito Juárez`; `San Pablo Guelatao`). The most precise result goes to C.

8.2 The Mixed ontologies comparator COM

This section briefly explains the COM algorithm, reported elsewhere [4, 12], which OM uses extensively. Let C_A be a node (a concept) in ontology A and P_A its predecessor. COM wants to find the most similar (to C_A) node C_B in ontology B. Let us call P_B the predecessor of this (yet to be found) node C_B. The algorithm has four cases:
1.- Case A: the concept C_A matches with C_B in B and the predecessors P_A and P_B too (C_B and P_B are found in B).
2.- Case B: P_A matches P_B but no match occurs between C_A and C_B (P_B was found in B, but no C_B can be found).
3.- Case C: C_A matches C_B but there is no match between P_A and P_B (no P_B can be found in B).
4.- Case D: C_A does not match with C_B and P_A does not match P_B. (no C_B nor P_B can be found in B).

8.2.1 Case A: C_A matches with C_B and P_A matches with P_B

Given C_A and P_A, COM looks in B for two concepts: C_B and P_B, such that the definition of P_B matches the majority of the words that define P_A, and the majority of the words that define C_B match the definition of C_A. In that case, the algorithm returns:

- The C_B (known as *cms* too) $\in$ B,

• The *vs* value, a number between 0 and 1, where 0 means no match, 1 means strong match.

8.2.2 Case B: P_A finds a matching P_B , but C_A does not

P_B is found but no C_B. In this case, COM is called recursively with P_A as parameter to confirm that P_B is an predecessor of C_A. If a cousin of P_B or candidate found (P_B') happens to be the root of the ontology (O_{BRoot}) then the algorithm finishes without success. If that doesn't happen, then C_A is looked in B through each son of P_B. That son with the majority of its properties and values matching those of C_A (making repeated calls to COM) will be returned. That means, the P_B son is searched in B possessing the majority of properties and values of C_A. If the candidate C_B' has sons, it is verified that they match (making repeated calls to COM) with C_A sons. If a C_B' is found with the expected properties, the algorithm finishes successfully returning such C_B'. Otherwise, COM tries to find C_B' among father's sons (in B) of P_B; that means, among the brothers of P_B. If that doesn't happen, OM looks among the grandchildren of P_B. If C_B' is not found, then the closest match to C_A is an (unknown, not present) son of P_B, therefore COM returns "son of P_B" (that means that a P_B son that doesn't exist yet into B is the most similar node to C_A) and then the algorithm finishes.
Example. Let `Furniture` ∈ A with son `kitchen dinette`, while `Furnishings` ∈ B exists with sons `table`, `chair`, `bed` and `chest`, but with no `kitchen dinette`. When C_A = `kitchen dinette` is searched in B, COM returns C_B = "son of `Furnishings`", meaning that no match to C_A was found among the furnishings of B, but that COM is aware that the match to `kitchen dinette` ∈ A is a son of `Furnishings` ∈ B.

8.2.3 Case C: C_B is found but P_B can not be found

If C_B is found but not P_B, then COM checks if the grandfather of C_B in B is similar to P_A, or if the great-grandfather of C_B in B is similar to P_A (this was mentioned in Case A). If this is the case, then the most similar concept of P_A in B is the grandfather or great-grandfather of C_B and the algorithm finishes. If it is not found, then COM verifies if the majority of the relations and their values of C_A match those of C_B and if the majority of the C_A sons match the majority of C_B's sons; if the properties and sons indeed match, then the answer is C_B and the algorithm finishes, even though P_B (which corresponds to the concept P_A on A) was not found on B. If just a part of properties and sons match then the answer is "probably C_B" and the algorithm finishes. If no properties nor sons match, then the answer is "doesn't exist" and the algorithm concludes. This case considers the concepts that share the same label or word description but different meanings, for

example: in the ontology A there is a concept `Bureau` with the predecessor `Office` and the ontology B a concept `Bureau` with predecessor `Furniture`.

8.2.4 Case D: Neither C_B nor P_B can be found in B

If C_B doesn't exist and neither P_B exists, then the answer of COM is "doesn't exist" and the algorithm ends. This situation arises when we are considering two different ontologies, for example: an ontology about `Information Systems` and another ontology about `Natural Resources`.

References

1. **BanerJee, S.,** and **Pedersen T**. Extended Gloss Overlaps as Measure of Semantic Relatedness. *Proc. of IJCAI-03*, pp. 805-810. México. 2003
2. **Botello, A**. Infering relations among autonomous data bases. CIC-IPN. Ph. D. thesis in progress. 2009. (In Spanish)
3. **Colorado, F.** *Mappiung words to concepts: disambiguation.* CIC-IPN, M. Sc. Thesis, C (In Spanish). Available in http://www.divshare.com/download/6096165-b6d
4. **Cuevas, A**., and **Guzmán, A**. Improving the Search for the Most Similar Concept in other Ontology. *Proc. XVIII Congreso Nacional y IV Congreso Internacional of Informática y Computación*. Torreón Coah. México. October 2005.
5. **Cuevas, A**., and **Guzman, A**. A Language and Algorithm for Automatic Merging of Ontologies, a chapter of the book "Handbook of Ontologies for Business Interactions," Peter Rittgen, ed. Idea Group Inc, Publishers. Hershey, PA, USA. 2008
6. **Cuevas, A.** Union of ontologies using semantic properties. Ph. D. thesis. CIC-IPN. Dec. 2006. (In Spanish). Available in: http://www.divshare.com/download/6096305-b18
7. **Dou, D., McDermott, D**., and **Qi, P**. Ontology Translation by Ontology Merging and Automated Reasoning. *Proc. EKAW Workshop on Ontologies for Multi-Agent Systems*. 2002.
8. **Fellbaum, C**. WordNet, An Electronic Lexical Database. Library of Congress Cataloging in Publication Data. 1999.
9. **Fridman, N.,** and **Musen, M**. PROMPT: Algorithm and Tool for Automated Ontology Merging and Alignment. *Proc. Seventeenth National Conference on Artificial Intelligence*. pp 450-455, Austin, TX, USA, 2000.
10. **Gruber, T**. Toward principles for the design of ontologies used knowledge sharing. Originally in N. Guarino & R. Poli, (Eds.), *International Workshop onf Formal Ontology*, Padova, Italy. 1993.
11. **Guzmán, A**., and **Levachkine, S**. Hierarchies Measuring Qualitative Variables. *Lecture Notes in Computer Science* LNCS **2945** [Computational Linguistics and Intelligent Text Processing], Springer-Verlag. 262-274. ISSN 0372-9743. 2004.
12. **Guzmán, A**., and **Olivares, J**. Finding the Most Similar Concepts in two Different Ontologies. *Lecture Notes in Artificial Intelligence* LNAI **2972**, Springer-Verlag. 129-138. ISSN 0302-9743. 2004.
13. **Guzman-Arenas, A**., **Jimenez, A**. Obtaining the inconsistency and consensus among a set of assertions on a qualitative attribute. Submitted to *Journal Expert Systems with Applications.*

14. Kalfoglou, Y., and **Schorlemmer, M**. Information-Flow-based Ontology Mapping. *Proc. of the 1st International Conference on Ontologies, Databases and Application of Semantics* (ODBASE'02), Irvine, CA, USA. 2002.

15. Kotis, K., and **Vouros, G**., **Stergiou, K**. Towards Automatic of Domain Ontologies: The HCONE-merge approach. *Elsevier's Journal of Web Semantic (JWS)*, vol. **4**:1, pp 60-79. 2006. Available in http://authors.elsevier.com/sd/article/S1570826805000259

16. McGuinness, D., Fikes, R., Rice, J., and **Wilder, S**. The Chimaera Ontology Environment Knowledge. *Proc. of the Eighth International Conference on Conceptual Structures Logical, Linguistic, and Computational Issues*. Darmstadt, Germany. 2000.

17. Nery, P. Parser for the conversion of text documents into ontologies. Thesis in progress. CIC-IPN. México. 2009. (In Spanish)

18. Puscasu, G., Ramirez Barco P, *et al*. On the identification of temporal clauses. *LNAI **4293***, 911-921 (MICAI 06). 2006.

19. Reed, S. L., and **Lenat, D.** Mapping Ontologies into CyC. *Proc. of AAAI Workshop on Ontologies and the Semantic Web*, Edmonton, Canada. 2002.

20. Stumme, G., Maedche, A. Ontology Merging for Federated ontologies on the semantic web. In: E. Franconi, K. Barker, D. Calvanese (Eds.): *Proc. Intl. Workshop on Foundations of Models for Information Integration (FMII'01)*, Viterbo, Italy, 2001. lNAI, Springer 2002.

References in Internet.

1. Loom http://www.isi.edu/isd/LOOM/LOOM-HOME.html
2. wikipedia http://es.wikipedia.org/wiki/Ontolog%C3%ADa
3. Hotel Finca Santa Martha http://www.littlehotelsofspain.co.uk/santamarta.php
4. Finca Santa Marta http://www.innsite.com/inns/A004065.html

---

[i] Ontology B is obtained from: http://www.solarviews.com/span/solarsys.htm (The site shows an article, B was hand-made from such article).

[ii] Ontology A is obtained from: http://es.wikipedia.org/wiki/Sistema_Solar

[iii] B comes from: http://es.wikipedia.org/wiki/Redefinici%C3%B3n_de_planeta_de_2006

[iv] Ontology A is obtained from: http://es.wikipedia.org/wiki/Continente