

Combining statistical and logical inference for ontology alignment

Octavian Udrea and Lise Getoor

University of Maryland, College Park, MD 20742

{udrea,getoor}@cs.umd.edu

Abstract

In recent years, ontologies have proved a very attractive medium for storing domain knowledge in a variety of organizations; a few example domains are medicine¹ and genetics². The rapid evolution of representation standards has led to the creation of distinct ontology bases in overlapping domains, making it paramount for large systems to reconcile ontological data from multiple sources. A large body of work on ontology integration [Euzenat, 2004] has produced algorithms and heuristics that have proven successful in alleviating the high computation costs of the process. However, the effective use of ontology formalisms – such as rules and axioms – in the integration process remains an open question. In this paper, we present a novel ontology integration method that takes advantage of the logical inference capabilities in OWL Lite to improve the recall of the merged ontologies. Our CPI (Clustering with Partial Inference) algorithm uses (i) graph clustering techniques to improve the process of selecting candidate matching entities and (ii) a limited, adaptive reasoning process to account for the effects of introducing relationships between such entities. Our initial experiments, performed on a set of 30 pairs of publicly available ontologies, show that CPI offers a 30% improvement in recall and a 25% improvement in answer quality compared to FCA-merge [Stumme and Maedche, 2001].

1 Introduction

As ontologies become increasingly widespread, we witness the emergence of large ontologies in overlapping domains; for example, two large ontologies on biology are Tambis³ and the subset of the Cyc ontology relating to

biological data⁴. As more and more tools are built to reason within these domains, reconciliation between ontologies becomes a major challenge for the Semantic Web Community [Euzenat, 2004].

Informally, the *ontology alignment problem* can be stated as: given two ontologies S_1 and S_2 , can we determine a set of relationships (subsumption and equivalence) between entities (classes, properties and instances) in the two ontologies? This in turn relates to the *integration problem*: can we determine a new, consistent ontology S (often called the *integration witness*) and a set of mappings from entities in S_1 and S_2 to entities in S ? There are a number of differences between the two problems which are discussed in more detail in [Konieczny, 2000]; we largely ignore such cases in this paper and will often use the terms integration and alignment interchangeably.

The initial approaches to merging ontologies in the literature [Mitra *et al.*, 1999; Calvanese *et al.*, 2001; Bouquet *et al.*, 2003; McGuinness *et al.*, 2000; Stumme and Maedche, 2001] attempt to address the high computation costs of reasoning in an ontological framework by taking a Greedy heuristic approach. Specifically, the approach is to (i) locate candidate pairs of entities in an ontology, to (ii) choose the best pair according to a similarity measure, and (iii) repeat the process in a Greedy fashion. The focus is often that of finding good similarity measures and heuristics for selecting candidate pairs. There is also a smaller body of work on “global” similarity measures, – instead of looking at individual candidate pairs, in this case we focus on defining the measure at the ontology level. The initial work of Euzenat *et al.* [Euzenat and Valtchev, 2003] establishes a global similarity measures that encompasses most existing frameworks; the measure is then used in the OLA [Euzenat *et al.*, 2004] system.

For rich formal frameworks such as OWL, the question that comes to mind is whether the richer formalism (e.g., axioms, rules, etc.) can be used to guide the integration process. In this paper we give a novel method of ontology integration for OWL Lite that takes advantage of the inferential possibilities of the language and present initial

¹ <http://www.cs.man.ac.uk/~rector/ontologies/simple-top-bio/>

² <http://www.geneontology.org/>

³ <http://www.cs.man.ac.uk/~horrocks/Ontologies/tambis.daml>

⁴ <http://opencyc.sourceforge.net/daml/cyc.daml>

experimental results on 30 pairs of real-world ontologies that suggest partial logical inference greatly improves the recall of ontology alignment when compared to the FCA-merge algorithm. Our CPI (Clustering with Partial Inference) uses graph clustering [Bhattacharya and Getoor, 2006] as a framework for choosing candidate nodes, by applying standard similarity measures to clusters instead of individual nodes in the ontology. The algorithm then performs a small number of logical inference steps to account for any logical consequences of merging the two clusters. The evidence gathered from the inference is used as feedback into the initial similarity score between candidate clusters.

The rest of the paper is organized as follows. We briefly describe OWL Lite and provide examples in Section 2. We describe the CPI method and algorithm in Section 3. Our hypothesis is that using logical inference to improve the selection of candidate pairs leads to a significant improvement in the quality of the alignment at a slight computational cost; we provide experimental evidence for this hypothesis in Section 4. Some of our most important findings are that (i) CPI leads to a 30% improvement in recall and approximately 25% improvement in overall quality compared to FCA-merge, for a slight increase in running time, (ii) the best choice of parameters for the algorithm strongly depends on the specific pair of ontologies and (iii) a relatively small number of inference steps (5 steps for CPI) cause a significant rise in quality. We present a (brief) overview of related work in Section 5 and conclude in Section 6.

2 Technical preliminaries

In this section we give a short overview of OWL Lite, introducing a simplified notation that will be used throughout the paper and two small ontology examples. We assume familiarity with the basics of OWL Lite⁵.

We use \mathcal{T} to denote the set of all primitive data types (e.g. *xsd:integer*, *xsd:string*, etc) and \mathcal{D} to denote the set of all possible data values of types in \mathcal{T} . Without loss of generality, we assume \mathcal{D} is finite.

An OWL Lite ontology is a document expressing relationships between classes, instances (or individuals) and properties. It extends the inferential capabilities of the RDF schema by allowing a much richer set of axioms. We will denote an OWL ontology by $\mathcal{O} = (\mathcal{C}, \mathcal{P}, \mathcal{I}, \mathcal{A}, \mathcal{F})$, where \mathcal{C} , \mathcal{I} , \mathcal{P} are the sets of classes, individuals and properties respectively, \mathcal{F} is the set of the form $(X p Y)$ where $X, Y \in \mathcal{I}$ and $p \in \mathcal{P}$. The set of axioms \mathcal{A} includes the following:

- Restriction axioms are used to define a set of individuals that are connected by a property p to a set of instances with specific properties (e.g., all such instances are from the same class or the size of the set is at most 1). These are generally used to define new classes.

⁵ The full description of the OWL Lite syntax and semantics can be found at <http://www.w3.org/TR/2004/REC-owl-semantics-20040210/>.

- Property axioms specify sub-property, equivalence and inverseOf relations between elements of \mathcal{P} .
- Property characterization axioms specify which properties are transitive, symmetric, functional or inverse functional.
- Class axioms specify sub-class and equivalence relations, as well as allowing for new classes to be defined as intersections of existing classes.
- Instance axioms can specify whether two given individuals are identical or different.

Figure 1 shows two simple OWL Lite ontologies. The two ontologies share part of their schemas (for example, the class *Organization* is the same in both ontologies). The two ontologies also share three axioms: *headOf* is a functional property (i.e. an employee can only head one department at a time), as well as an inverse functional property (i.e. a department can only have one head) and *affiliatedWith* is a transitive property. We will use this as a running example to illustrate some of the challenges and benefits of the CPI algorithm.

To present the reasoning process in OWL Lite, we first have to introduce class and property extensions.

For a class $c \in \mathcal{C}$, the class extension of c – which we denote by $\epsilon(c)$ ⁶ –, is informally the set of instances that are of type c . In the ontology on the right of Figure 1, $\epsilon(\textit{Organization})$ is {USCB Chemical Engineering, USCB Dept of Computer Science, Verification Lab}, as all three instances have a type that is a subclass of *Organization*.

For a property $p \in \mathcal{P}$, the property extension of p – denoted by $\mu(p)$ is the set of pairs of instances (X, Y) such that the fact $(X p Y)$ can be *inferred* from the ontology (we will informally define the inference process a little later). In the left-hand side ontology in Figure 1, $\mu(\textit{affiliatedWith})$ is {(ArchLab, VLab), (VLab, UCSB CS Dept), (ArchLab, UCSB CS Dept)}. Note that we have already performed an inferential step by adding the pair (ArchLab, UCSB CS Dept), even though no edge labeled with *affiliatedWith* exists between these two nodes. The transitivity of the *affiliatedWith* property causes the pair to be part of its property extension.

Reasoning in OWL is typically performed by using a *tableaux algorithm* [Horrocks *et al.*, 2000]. However, since our goal is to perform a small number of inference steps, often localized, we attempt to incrementally build μ and ϵ , by adding facts to the initial ontology. For instance, one such step applies the *affiliatedWith* transitivity axiom to (ArchLab, VLab), (VLab, UCSB CS Dept) and adds (ArchLab, UCSB CS Dept) to $\mu(\textit{affiliatedWith})$. For reasons of space, we provide an informal description of this step-by-step process:

1. We start with $\epsilon^0(c)$ containing only the individuals that are explicitly stated to have type c in the initial ontology. $\mu^0(p)$ is the set of pairs that contain an explicit edge labeled with p .

⁶ We will use ϵ and μ as a shorthand for the original OWL notation of EC and ER.

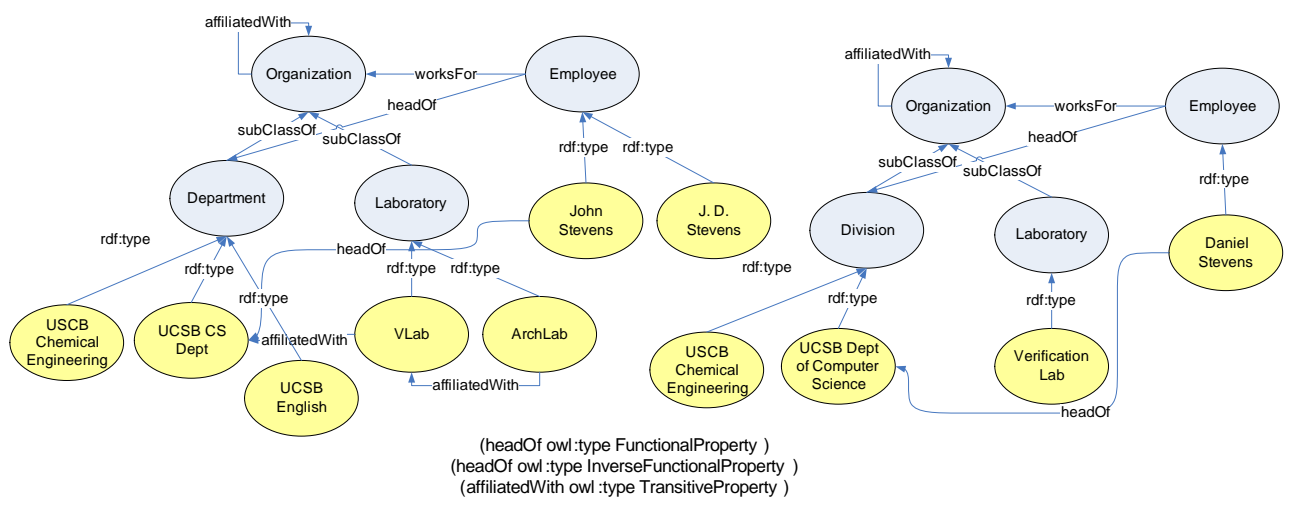


Fig. 1. Two OWL Lite ontologies

- At each step, we either add one element Z to $\epsilon(c)$ for some $c \in \mathcal{C}$ or an element (X, Y) to $\mu(p)$ for some $p \in \mathcal{P}$ by applying exactly one axiom. Informally, we attempt to compute a logical *closure* of the ontology. We denote this by $(\mu^{i+1}, \epsilon^{i+1}) \leftarrow \Delta(\mu^i, \epsilon^i)$. Δ is clearly a monotonic operator, hence this incremental closure is a fixpoint computation for Δ .

2.1 The ontology integration problem

Our goal is to find a set of mappings between classes, individuals and properties in two ontologies $\mathcal{S}_1 = (\mathcal{C}_1, \mathcal{P}_1, \mathcal{I}_1, \mathcal{A}_1, \mathcal{F}_1)$ and $\mathcal{S}_2 = (\mathcal{C}_2, \mathcal{P}_2, \mathcal{I}_2, \mathcal{A}_2, \mathcal{F}_2)$. Such mappings include subsumption or equivalence for classes and properties and equivalence for individuals. In short, we would like to locate a set of axioms A_{int} referring to entities in either \mathcal{S}_1 or \mathcal{S}_2 which contain only class, property or individual equivalence or sub-class or sub-property axioms. For now, we will ignore the differences between alignment and integration and consider this problem equivalent to that of finding $S = (\mathcal{C}_1 \cup \mathcal{C}_2, \mathcal{I}_1 \cup \mathcal{I}_2, \mathcal{P}_1 \cup \mathcal{P}_2, \mathcal{A}_1 \cup \mathcal{A}_2 \cup \mathcal{A}_{int}, \mathcal{F}_1 \cup \mathcal{F}_2)$, an integration witness for $\mathcal{S}_1, \mathcal{S}_2$.

Example 1. Figure 2 shows a possible integration witness for the ontologies in Figure 1. The bold underlined nodes represent clusters that resulted from the integration process. There is a one-to-one correspondence between such clusters and OWL Lite axioms. For instance, the cluster $\{\text{Verification Lab}, \text{VLab}\}$ corresponds to $(\text{Verification Lab owl:sameAs VLab})$.

3 Ontology integration with CPI

In this section, we provide the formal details of the Clustering with Partial Inference (CPI) approach. We start by showing how we are able to differentiate subsumption and equivalence of classes and properties, then describe how similarity scores are computed for nodes in

the ontology (classes, properties, individuals) and clusters. Finally, we show how inference can be used to refine initial similarity scores. For the purpose of this section, we assume $\mathcal{O}_1 = (\mathcal{C}_1, \mathcal{P}_1, \mathcal{I}_1, \mathcal{A}_1, \mathcal{F}_1)$ and $\mathcal{O}_2 = (\mathcal{C}_2, \mathcal{P}_2, \mathcal{I}_2, \mathcal{A}_2, \mathcal{F}_2)$ are two consistent OWL Lite ontologies and we denote by $\mathcal{O} = (\mathcal{C}, \mathcal{P}, \mathcal{I}, \mathcal{A}, \mathcal{F}) = \mathcal{O}_1 \cup \mathcal{O}_2$.

3.1 Deciding between subsumption and equivalence

An integration witness may contain either subsumption or equivalence relations between classes and properties; deciding between these two types of relations is far from trivial. We use a simple decision procedure to clearly separate the two cases. In the following, let $c_1, c_2 \in \mathcal{C}$. Let $\lambda_r \in [0, 1]$ be an arbitrary, but fixed threshold value.

- If $\frac{|\epsilon^0(c_1) - \epsilon^0(c_2)|}{|\epsilon^0(c_1)|} < \lambda_r$ and $\frac{|\epsilon^0(c_2) - \epsilon^0(c_1)|}{|\epsilon^0(c_2)|} \geq \lambda_r$, then consider the axiom $c_1 \subseteq c_2$.
- If $\frac{|\epsilon^0(c_1) - \epsilon^0(c_2)|}{|\epsilon^0(c_1)|} \geq \lambda_r$ and $\frac{|\epsilon^0(c_2) - \epsilon^0(c_1)|}{|\epsilon^0(c_2)|} < \lambda_r$, then consider the axiom $c_2 \subseteq c_1$.
- Otherwise consider the axiom $c_1 \equiv c_2$.

Intuitively, this set of rules state that if the set of instances of c_1 ($\epsilon^0(c_1)$) is “almost” a subset of the set of instances of c_2 ($\epsilon^0(c_2)$), then we should consider the subsumption relation $c_1 \subseteq c_2$. The threshold value λ_r determines how many of the instances of c_1 should also be instances of c_2 – in our experiments we have empirically determined that a value $\lambda_r = 0.2$ provides the best answer quality. We omit the description of a similar decision procedure for properties; the method can also be applied to clusters of classes $C \subseteq \mathcal{C}$ – by defining $\epsilon^0(C) = \cup_{c \in C} \epsilon^0(c)$.

Example 2. Consider the two ontologies in Figure 1 and the candidate pair (Department, Division). Then $\frac{|\epsilon^0(\text{Department}) - \epsilon^0(\text{Division})|}{|\epsilon^0(\text{Department})|} = \frac{2}{3}$ and $\frac{|\epsilon^0(\text{Division}) - \epsilon^0(\text{Department})|}{|\epsilon^0(\text{Division})|} = \frac{1}{2}$. Then if $\lambda_r = 0.6$, we

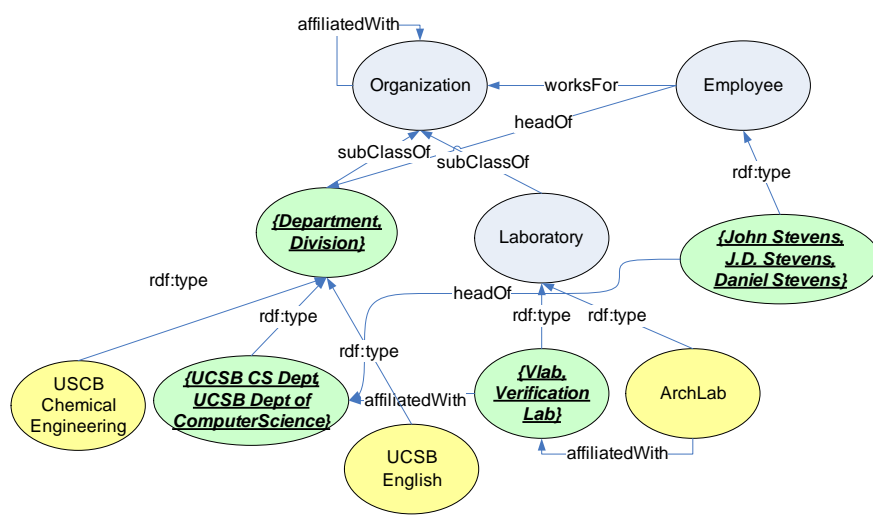


Fig. 2. A possible integration witness for the ontologies in Figure 1

would consider the relationship (Division subClassOf Department). If $\lambda_r = 0.3$, then we would choose (Division owl:equivalentClass Department).

3.2 Similarity measures

Since OWL Lite contains several types of entities, we define separate similarity measures for each. For a pair of entities (classes, properties, individuals) e_1, e_2 the similarity score is expressed as $sim(e_1, e_2) = \lambda_x sim_{lexical}(e_1, e_2) + \lambda_s sim_{structural}(e_1, e_2) + \lambda_e sim_{extensional}(e_1, e_2)$, in the same spirit as the measure defined in [Euzenat and Valtchev, 2003]. The above score is based on three components depending on the lexical aspects (name or URI), the structural aspects (neighborhoods) and the extensional aspects (ϵ, μ) of the ontology graph. We do not restrict this model by the assumption that any of the three λ parameters should have the same value in case of classes as in the case of instances; therefore, we will denote λ parameters for classes, properties and individuals with a c, p, i upper index.

Table 1 informally describes the measures used for classes, properties and instances. For lexical similarity, Jaro-Winkler[Winkler, 1994] provides the best answer quality for our dataset. Furthermore, for low Jaro-Winkler scores, WordNet is used to determine potential synonymity between entity names; we then assign a similarity score based on which synonym set the pair belongs to (synsets are ordered in WordNet by frequency of use).

For the structural similarity of classes, we consider the Jaccard distance between the immediate neighbors in the $owl : subClassOf$ hierarchy. For two multisets S_1, S_2 , the Jaccard distance is defined as $Jaccard(S_1, S_2) = \frac{|S_1 \cap S_2|}{|S_1 \cup S_2|}$. We also considered using the Jaccard distance between the sets of properties incident to the two classes; this measure proved useless during our experimental evaluation and was therefore removed. Similarly, for properties, the similarity structure is based on their

$owl : subPropertyOf$ neighborhood. In this case, we found that axioms about properties played an important part in determining similarity. As described in Section 2, a property can be *transitive, reflexive, functional* or *inverse functional*. Let H_p be the set of features (out of the aforementioned four) of p . Then $sim_{features}(p_1, p_2) = Jaccard(H_{p_1}, H_{p_2})$. For an instance $X \in \mathcal{I}$, we consider the set of pairs (p, Y) such that $(X p Y) \in \mathcal{F}$. We then compute the Jaccard distance between these sets for structural similarity. The extension similarity is computed for both classes and properties in terms of their respective extension sets; there is no extensional similarity for instances.

Given a similarity score between nodes, it can be extended to clusters of nodes in the standard way. If E_1, E_2 are clusters of entities, the cluster similarity function⁷ could be one of:

1. **Jaccard similarity:** $sim(E_1, E_2) = \frac{|E_1 \cap E_2|}{|E_1 \cup E_2|}$.
2. **Single link:**

$$sim(E_1, E_2) = \min_{x \in E_1, y \in E_2} (sim(x, y))$$

3. **Average link:** $sim(E_1, E_2) = \frac{\sum_{x \in E_1} \sum_{y \in E_2} sim(x, y)}{|E_1| \times |E_2|}$.

Even though the cluster link methods seem computational expensive, we have found that cluster sizes for CPI are relatively small, making the computation feasible.

3.3 Impact of inference

In order to account for the consequences (according to OWL semantics) of clustering a number of existing nodes together, we perform a limited number of inference steps. We denote by N the number of inference steps performed by the algorithm; we would like to keep N small, since inference has a high computational cost.

⁷ We will abuse notation and denote both functions sim for simplicity.

Table 1. Similarity measures considered

	<i>sim_{lexical}</i>	<i>sim_{structural}</i>	<i>sim_{extensional}</i>
Classes	Jaro-Winkler and Wordnet	Jaccard on <i>owl : subClassOf</i> neighbors	Jaccard on set of instances
Properties	Jaro-Winkler and Wordnet	Jaccard on <i>owl : subPropertyOf</i> neighbors multiplied by <i>sim_{features}</i>	Jaccard on set of (X, Y) pairs in property extension
Instances	Jaro-Winkler	Jaccard on sets of incident (p, Y) pairs	0

Algorithm 1 CPI-Inference

Input: Consistent ontology $\mathcal{O} = (\mathcal{C}, \mathcal{P}, \mathcal{I}, \mathcal{A}, \mathcal{F}) = O_1 \cup O_2$, axiom a representing the current integration candidate (e.g., $(C_1 \text{ owl : equivalentClass } C_2)$), initial similarity score for alignment candidate a (e.g., the similarity score between C_1 and C_2)

Output: New similarity score normalized w.r.t. the consequences of inference. 0 if the candidate alignment causes an inconsistency

- 1: $\mathcal{A} \leftarrow \mathcal{A} \cup \{a\}$
- 2: $f \leftarrow 1$
- 3: **for** $i = 1$ to N **do**
- 4: **if** a is a class equivalence/subsumption axiom **then**
- 5: $p \leftarrow$ choose a property incident to one of the classes in a
- 6: **else if** a is a property equivalence/subsumption axiom **then**
- 7: $p \leftarrow$ choose a property referenced by a
- 8: **else if** a is an instance equivalence axiom **then**
- 9: $p \leftarrow$ choose a property incident to an instances in a
- 10: **end if**
- 11: **choose** an axiom $a' \neq a$ from \mathcal{A} that involves p
- 12: $(\mu^i, \epsilon^i) \leftarrow \Delta(\mu^{i-1}, \epsilon^{i-1})$ (by applying Δ for a')
- 13: **if** there is an inconsistency **then**
- 14: **return** 0
- 15: **end if**
- 16: **recompute** the affected instance set closures
- 17: **for** all pairs of $X, Y \in \mathcal{I}$ that have become equivalent **do**
- 18: $f \leftarrow f \cdot \frac{\text{sim}(X, Y)}{1 - \text{sim}(X, Y)}$
- 19: **end for**
- 20: **end for**
- 21: **return** $\min(1, f \cdot \text{score})$

The inference process is shown in Figure 1. The algorithm firsts adds the candidate alignment to the ontology. We then proceed by performing at most N inference steps. At each step, we randomly choose a property that is either: (i) incident to the classes/individuals in the candidate alignment or (ii) one of the actual properties in the candidate alignment. The algorithm determines that the candidate axiom should not be used (returns 0) if any inconsistency occurs while computing Δ . Note that this is not entirely sound – i.e., we do not check the entire ontology for consistency. However, the heuristic has proved reliable; in our experiments, we have found inconsistencies in the final integration witness that were not detected by *CPI-Inference* in less than 0.5% of the executions.

For each pair of individuals (or clusters of individuals) that have become equivalent due to the addition of a ,

CPI-Inference computes a normalization factor based on their existing similarity (line 18); the factor is monotonic in $\text{sim}(X, Y)$. Finally, *CPI-Inference* uses these factors to compute the new similarity score for candidate alignment a ; since a full normalization is not possible unless we reach a fixpoint for Δ , we simply impose that the similarity score for a is not greater than 1 (line 21).

Example 3. Consider the example in Figure 1, in which we have already created the cluster {John Stevens, J.D. Stevens} and we are considering merging this with the cluster {Daniel Stevens}. Let us also assume that the similarity measure between {USCB CS Dept} and {USCB Dept of Computer Science} is 0.6. By performing one inference step on the axiom (headOf owl:type FunctionalProperty), we must conclude that USCB CS Dept and USCB Dept of Computer Science are in the same instance set closure. Therefore, $f = \frac{0.6}{0.4}$ and the similarity measure between the candidate clusters increases significantly.

3.4 The CPI algorithm

In this section, we present the CPI method of computing an integration witness, given the initial ontologies O_1, O_2 . In the following, λ_t is an arbitrary but fixed similarity threshold, representing the minimum score for clusters to be considered as candidates for integration.

Algorithm 2 CPI

Input: Consistent ontologies O_1 and O_2

Output: Integration of O_1 and O_2

- 1: Compute $O \leftarrow O_1 \cup O_2$
- 2: Initialize clusters to single classes, properties and individuals in O
- 3: Compute initial similarity scores for all pairs of clusters
- 4: **repeat**
- 5: Determine equivalence/subsumption axioms for all pairs of clusters with similarity greater than λ_t
- 6: Run *CPI-Inference* in parallel for all pairs of clusters with similarity greater than λ_t
- 7: Choose pair of clusters with best similarity measure and merge them into a single cluster
- 8: Recompute similarity scores for affected clusters
- 9: **until** there are no clusters with similarity measure greater than λ_t
- 10: **return** O

Note that although we have presented the *CPI-Inference* for a single candidate alignment for reasons of simplicity, with only slight modifications the inference can be performed in parallel for a set of such can-

didates. The CPI algorithm is polynomial in the size of the ontology, since (i) the computation of similarity scores between clusters is polynomial, (ii) there is at most a quadratic number of clusters and (iii) we only perform N inference steps, each of which is quadratic in the size of O . We point out that the algorithm is not sound, since it may produce an inconsistent integration witness; however, we have determined empirically that the method produced inconsistent results approximately in 0.5% of the experimental runs.

One of the important issues in designing the *CPI* method was the ordering in which we compute similarity and inference. An alternative method we considered was that of performing inference first, followed by the computation of similarity scores. This approach turned out to be computationally unfeasible in most cases since inference has a tendency to produce a large blowup in the size of node neighborhoods, which are paramount to computing structural similarity.

4 Experimental evaluation

The implementation of *CPI* consists of approximately 5000 lines of Java code; the implementation uses the Pellet⁸ API to parse OWL Lite ontologies. The experiments described in this section were performed on 30 pairs of OWL Lite ontologies from www.daml.org, SchemaBroker and OntoWeb; the ontologies span a number of domains such as organization structure, geographical information, music, etc⁹. The sizes of the ontologies vary between 754 and 1513 entities, with an average of 1015. The optimal alignments we consider as the ground truth were determined by human reviewers through the following process:

- Each randomly picked reviewer was asked to determine the best alignment she thought possible for a given pair of ontologies.
- In each successive step, another randomly picked reviewer was asked to refine the ontology alignment determined in the previous step.
- We considered the process complete when the number of changes to an existing alignment dropped below 3 to avoid cyclic changes by different reviewers.

All the experiments were performed on a P4 3.2 GHz machine with 1GB of RAM running Windows XP. The running times below do not include I/O and parsing.

Given the ground truth alignments, we are able to compute the precision, recall and *F1* quality measure ($F1 = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}$), by analyzing the resulting alignments as sets of axioms. Table 2 summarizes the parameters considered in the experiments.

In our first set of experiments we attempted to determine the clustering method that yields the best *F1* quality. We used random sampling as follows: the values

⁸ <http://www.mindswap.org/2003/pellet/>

⁹ Generally, the ontologies were chosen so that human reviewers would be familiar with the domain without special training – as is the case with gene or medical ontologies.

Table 3. *CPI* optimal parameter values

λ_g^c	λ_g^i	λ_g^p	λ_s^c	λ_s^i	λ_s^p	λ_e^c	λ_e^i	λ_e^p	λ_r
0.2	0.4	0.1	0.5	0.6	0.4	0.3	0.5	0.7	0.2

for the λ parameters were chosen uniformly at random in the interval $[0, 1]$ and the value of N was set to 0 (i.e, no inference). The process was repeated 30 times; for each execution, we computed the average quality over all 30 pairs of ontologies. In 26 runs out of 30, the complete link method outperforms the other methods; the *F1* quality ranged between 0.5 and 0.67 with a standard deviation of 0.05. We then attempted to determine the values of the λ parameters. We varied each parameter in 0.1 increments, while requiring that $\forall w \in c, p, i, \lambda_x^w + \lambda_y^w + \lambda_z^w = 1$. The implicit assumption is that the effects of N and the λ parameters are independent. For the best parameter setting, the overall recall was 0.78 and precision was 0.83, leading to a *F1* measure of 0.804. The best values of the λ parameters for each ontology pair do not coincide with the overall optimum; the difference can be between 0.1 and 0.8 for λ_s^c . The optimal parameter values suggest that:

- Lexical matching for properties is almost irrelevant; to verify this, we manually checked a few of the ontology pairs and often found that the same concepts were expressed in different terms for properties. This is also supported by the fact that many of the alignments missed by the algorithm were alignments involving properties (we discuss this in more detail below). Classes suffer from a similar problem, but to a much lower degree.
- Structural similarity has the highest relevance. This is somewhat surprising, given that structural matching is usually depicted in the literature[Euzenat, 2004] as a workaround for the semantics problem.
- We have also determined that λ_r can vary between 0.1 and 0.4 without affecting quality too much; this is as expected, given that the similarity measure is the same for either subsumption or equivalence.

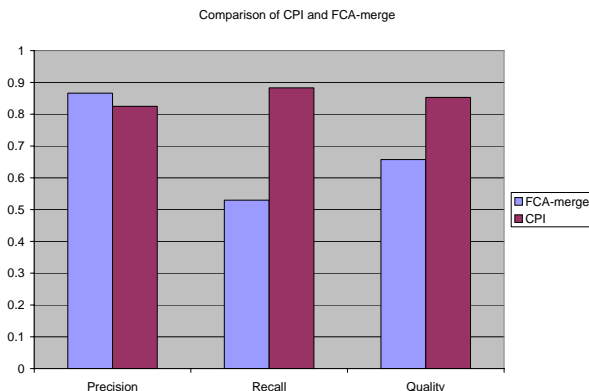


Fig. 4. Comparison between FCA-merge and *CPI*

Table 2. CPI experimental parameters

Parameter	Notes
λ_r	The threshold used in deciding on equivalence versus subsumption as a candidate relationship between classes and properties
$\lambda_e, \lambda_s, \lambda_x$	The weights for lexical, structural and extensional similarity. These are separate for classes, properties and instances. Note that λ_x^i is not relevant since instances do not have extensional similarity.
N	The inference depth.
λ_t	The overall similarity threshold.
Cluster similarity method	One of single link, complete link, average link.

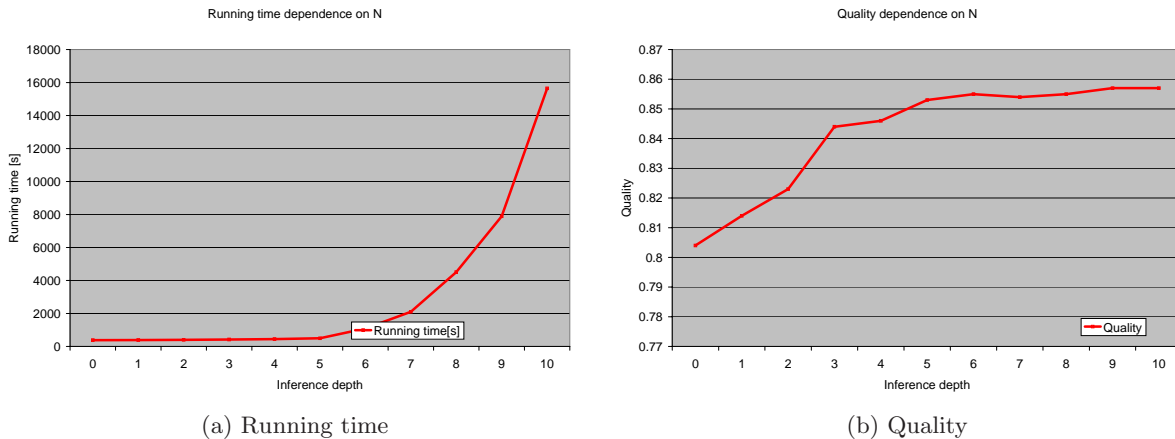


Fig. 3. Dependence of running time and quality on N

In the second set of experiments, we attempted to ascertain the impact of N on running time and quality. We varied the value of N from 0 to 10, measuring the average running time and F1 quality over the 30 pairs of ontologies. The results are displayed in Figure 3. For a good trade off in quality versus running time, we fixed $N = 5$ for the rest of the experiments.

In the third set of experiments, we compared CPI (with the aforementioned parameters) with the FCA-merge. We started by selecting articles from several US and European news sources on the topics of each ontology pair. These were fed into the FCA component that computes the formal concept lattice. Since FCA-merge requires a limited form of user interaction in its last step, the numbers reported are for the user choices which provide the best answer quality for the algorithm. The comparison in terms of precision, recall and quality is shown in Figure 4. On average, FCA-merge was able to compute the integration witness in 385 seconds (time spent interacting with the user is ignored), while CPI averaged at 496 seconds.

Finally, we looked at specializing CPI by focusing on a particular type of candidate alignments. We developed two variants of the algorithm called CPI-Instance and CPI-Class. The former focuses on creating clusters of instances first – and only after there are no such clusters that can be merged analyze class equivalence and subsumption; the latter starts by focusing on classes and then addresses instances. After running both algorithms on our dataset, we observed no significant difference in terms of quality. The quality of answer for all

pairs of ontologies were within one standard deviation of the mean for both algorithms; CPI-Instance averaged at 0.841, whereas CPI-Class averaged at 0.843. However, CPI-Instance proved relatively faster than its counterpart, averaging at 413 seconds, whereas CPI-Class took 542 seconds; this suggests that a bottom-up approach in merging ontologies is beneficial in terms of running time, since instances do not have neighborhoods as semantically rich as classes, while yielding similar answer quality.

5 Related work

We briefly mention some of the successful ontology matching systems currently in existence. Limited human interaction was found to be very helpful in guiding the merging process; systems such as Anchor-PROMPT [Noy and Musen, 2003] and Chimerae [McGuinness *et al.*, 2000] use a variety of lexical and structural based techniques to evaluate the semantic similarity of concepts and allow a human reviewer to make the final decision in difficult cases. FCA-merge [Stumme and Maedche, 2001] also takes advantage of the human element, but uses formal concept analysis based on a concept lattice extracted from text documents relevant to the ontology topics of interest. IF-map [Kalfoglou and Schorlemmer, 2003] is another system rooted in formal concept analysis that aligns two ontologies based on how they map to a third reference ontology. S-Match [Giunchiglia *et al.*, 2005] is a schema-matching system based on an extensible library of matching generators ranging from

lexical methods to SAT solvers. OLA [Euzenat *et al.*, 2004] is the only system designed specifically for OWL Lite and which uses a global similarity measure for ontology alignment. Local similarity between entities in the ontologies produces a set of equations which are iteratively solved to provide a set of mappings. However, OLA does not use the inference capabilities of OWL directly.

Our approach is closely related to the recent advances in modular ontologies. Grau *et al.* [Grau *et al.*, 2004] and Bao *et al.* [Bao *et al.*, 2006] argue that one of the crucial issues of the Semantic Web is that of reasoning with multiple distinct, but linked ontologies – a problem not yet addressed by the OWL standard. Bao *et al.* describe two classes of approaches: the use of mappings between ontology modules (Distributed Description Logics and \mathcal{E} -connections) and the use of importing (Package-based Description Logics). These approaches are complementary to ours in two ways: first, *CPI* can be used to detect some of the mappings between ontology modules automatically. Second, as a future avenue of research, *CPI* could make use of the semantics of modular ontology languages for its inference component.

6 Conclusions and future work

In this paper, we have presented a novel approach to ontology integration that improves on the existing research by efficiently using ontology formalism to improve the quality of ontology alignment. Our approach, based on clustering and limited, adaptive inference, yields a 30% improvement in answer quality compared to one of the leading integration algorithms, FCA-merge. We envision two main avenues of improvement for *CPI*: (i) a tighter integration of the inference and clustering steps of the algorithm and (ii) a better adaptive selection of inference axioms for *CPI*-Inference.

References

[Bao *et al.*, 2006] Jie Bao, Doina Caragea, and Vasant Honavar. Modular ontologies - a formal investigation of semantics and expressivity. In *ASWC*, pages 616–631, 2006.

[Bhattacharya and Getoor, 2006] I. Bhattacharya and L. Getoor. *Entity Resolution in Graphs*, chapter Mining Graph Data (L. Holder and D. Cook, eds.). Wiley, 2006.

[Bouquet *et al.*, 2003] Paolo Bouquet, Fausto Giunchiglia, Frank van Harmelen, Luciano Serafini, and Heiner Stuckenschmidt. C-OWL: Contextualizing Ontologies. . In *Proceedings of the International Semantic Web Conference*, pages 164–179, 2003.

[Calvanese *et al.*, 2001] Diego Calvanese, Giuseppe De Giacomo, and Maurizio Lenzerini. A framework for ontology integration. In *Proceedings of SWWS'01, The first Semantic Web Working Symposium*, pages 303–316, 2001.

[Euzenat and Valtchev, 2003] J. Euzenat and P. Valtchev. An integrative proximity measure for ontology alignment. In *Proceedings of the International Semantic Web Conference Workshop on Semantic Information Integration*, pages 33–38, 2003.

[Euzenat *et al.*, 2004] Jerome Euzenat, David Loup, Mohamed Touzani, and Petko Valtchev. Ontology alignment with OLA. In *Proceedings of the International Semantic Web Conference Workshop on Evaluation of Ontology-based tools (EON)*, pages 59–68, 2004.

[Euzenat, 2004] Jerome Euzenat. State of the art on ontology alignment, 2004. citeseer.ist.psu.edu/727262.html.

[Giunchiglia *et al.*, 2005] Fausto Giunchiglia, Pavel Shvaiko, and Mikalai Yatskevich. S-Match: an algorithm and an implementation of semantic matching. In *Semantic Interoperability and Integration*, number 04391 in Dagstuhl Seminar Proceedings, 2005.

[Grau *et al.*, 2004] Bernardo Cuenca Grau, Bijan Parsia, and Evren Sirin. Working with multiple ontologies on the semantic web. In *International Semantic Web Conference*, pages 620–634, 2004.

[Horrocks *et al.*, 2000] I. Horrocks, U. Sattler, and S. Tobies. Practical reasoning for very expressive description logics. *Logic Journal of the IGPL*, 8(3):239–264, 2000.

[Kalfoglou and Schorlemmer, 2003] Y. Kalfoglou and M. Schorlemmer. If-map: an ontology mapping method based on information flow theory. *Journal on Data Semantics*, 1(1):98–127, October 2003.

[Konieczny, 2000] Sbastien Konieczny. On the difference between merging knowledge bases and combining them. In *Seventh International Conference on Principles of Knowledge Representation and Reasoning (KR'00)*, pages 135–144, 2000.

[McGuinness *et al.*, 2000] Deborah L. McGuinness, Richard Fikes, James Rice, and Steve Wilder. An environment for merging and testing large ontologies. In *Proceedings of the 7th International Conference on Principles of Knowledge Representation and Reasoning*, pages 483–493, 2000.

[Mitra *et al.*, 1999] P. Mitra, G. Wiederhold, and J. Jannink. Semi-automatic integration of knowledge sources. In *Proceedings of the 2nd International Conference On Information FUSION'99*, 1999.

[Noy and Musen, 2003] Natalya F. Noy and Mark A. Musen. The PROMPT suite: interactive tools for ontology merging and mapping. *International Journal of Human-Computer Studies*, 59(6):983–1024, 2003.

[Stumme and Maedche, 2001] Gerd Stumme and Alexander Maedche. FCA-MERGE: Bottom-Up Merging of Ontologies. In *IJCAI*, pages 225–234, 2001.

[Winkler, 1994] W. Winkler. Advanced methods for record linkage., 1994. Technical report, Statistical Research Division, Washington, DC: U.S. Bureau of the Census.