Puzzling It Out: Supporting Ontology Evolution with Applications to eGovernment

Aviv Segev
Technion – Israel Institute
of Technology
Haifa 32000 Israel
asegev@tx.technion.ac.il

Avigdor Gal
Technion – Israel Institute
of Technology
Haifa 32000 Israel
avigal@ie.technion.ac.il

Abstract

In recent years, the use of ontologies in information systems has dramatically increased. Ontology design and maintenance, nonetheless, have been and still are daunting tasks. We argue that ontologies need to evolve, or else the semantic infrastructure of the information system will no longer support the organization's changing needs. Therefore, in this work we aim at tackling the problem of ontology evolution. We propose to use (machine-generated) contexts as a mechanism for quantifying relationships among concepts. To do so we compare contexts that are associated with ontology concepts. Our approach is unique in two aspects. First, we base it on a combination of ontologies and contexts, where contexts replace, to a certain extent, the role of the ontology engineer in the process. Second, we provide the ontology administrator with an explicit numeric estimation of the extent to which a modification "makes sense." We motivate our work with examples from the field of eGovernment applications and support our model with an empirical analysis, using real-world traces of news syndication.

Keywords: Ontology evolution, Contexts, Semantic interpretation, eGovernment applications

1 Introduction

In recent years, the use of ontologies in information systems has dramatically increased. Ontologies aim at providing a complete semantic understanding of a domain, enabling powerful reasoning that can serve as a basis for a semi-automatic decision making process. This strength, however, also constitutes the main obstacle to widespread adaptation of ontologies. Ontology design, for one, has been and still is a daunting task. It requires collaboration of experts within the organization with ontology engineers. The former provide the domain knowledge while the latter bring in the modeling ability. Such a collaboration may consume many organizational resources in terms of both time and monetary units. This difficulty is rooted in the need for a careful, accurate, and complete design of an ontology, a task that requires years of training and cannot be trusted to the organization expert.

In this work we address the problem of ontology evolution. We acknowledge that building an ontology from scratch requires a joint effort of experts, internal to the organization, and ontology engineers, external to it. We envision a new organizational role, called an *ontology administrator*, the parallel role to a database administrator, to handle ontology evolution. An ontology administrator is an IT professional with some training in maintaining ontologies and can therefore be trusted with evolving ontology from within the organization. In this work we provide a model for ontology evolution in which given an ontology relationship (*e.g.*, *disjoint*, representing the knowledge that an instance of one concept cannot be an instance of another) and operands (*e.g.*, two concepts or classes), an ontology administrator is provided with a quantified response regarding the extent to which the given relationship is valid for the given operands (see Figure 1).

To assist us in this task we utilize contexts (McCarthy, 1993), system-generated descriptors that serve as local views of a domain. Such a local view can be generated, for example, by analyzing a document that is relevant to an ontology concept. In this work we use contexts as a light-weight mechanism for assisting ontology administrators in making minor modifications to existing ontologies. We propose to use (machine-generated) contexts in quantifying relationships among concepts. Therefore, the validity of a relationship is measured by comparing the contexts that are associated with the operands. We believe that such a solution would significantly assist in the support of ontology evolution, to the extent that an ontol-

ogy administrator will be able to perform (at least minor) ontology evolution without the use of an ontology engineer. Our approach is unique in two aspects. First, we base it on a combination of ontologies and contexts, where contexts replace, to a certain extent, the role of the ontology engineer in the process. Second, we provide the ontology administrator with an explicit numeric estimation of the extent to which a modification "makes sense." To the best of our knowledge, none of the models for semi-automatic creation of ontologies (*e.g.*, bootstrapping (Ciravegna et al., 2003)) or for ontology evolution (*e.g.*, (Tsatsaronis et al., 2005)) make use of such quantified measures. We believe this feature is imperative in providing ontology administrators with a measure of validity in evolving ontologies.

The main contribution of this work is thus twofold. On a conceptual level, we introduce an *ontology verification* model, a **quantified** model for automatically assessing the validity of relationships in an ontology. The quantification allows the ontology administrator to define a level of validity for verifying each relation type in the ontology. On an algorithmic level, we provide a mapping of several ontology operators for defining relationships into context relationships.

The motivation of this work stems from QUALEG[1] and TERREGOV[2] eGovernment projects. In these projects, ontologies are used to drive government activities. For example, the relevance of citizen online debates to this or that role in a government can be determined by extracting the context of a debate and mapping it to an ontology of government roles (*i.e.*, a civil servant). Consider an eGovernment application describing the process by which government policies are determined and revised. An ontology is specifically designed and tailored by an ontology engineer. A change in this process, even a minor one from a design point of view, *e.g.*, adding one more government role as a participant in a meeting, may require (depending on the original ontology design) a renewed collaboration of civil servants with ontology engineers to apply changes to the ontology. Additional hurdles can be caused by multi-lingual environments (such as in the EU and Canada) and multi-cultural environments (*e.g.*, in border cities).

Throughout this paper, we motivate our work with examples from the eGovernment domain. However, due to the absence of large scale data sets for this domain, we support our model with an empirical analysis using real-world news syndication traces. The rest of the paper is organized as follows. We start with the preliminaries, formally defining ontologies and contexts in Section 2. In Section 3 we introduce the ontology verification model, followed by a proposal of a mapping of the ontology verification problem to contexts in Section 4. We then provide in Section 5 some empirical experiences. We conclude with a short summary in Section 6.

[1] http://www.qualeg.eupm.net/

[2] http://www.terregov.eupm.net

2 Ontologies And Contexts

In this work we provide a specific algorithm for automatically verifying ontological relationships from the work of (Noy and Klein, 2004). The underlying ontology model is therefore that of (Gruber, 1993): an *ontology* is an explicit specification of a conceptualization of a domain. We assume reader familiarity with basic concepts in conceptual modeling.

We define a descriptor c_i as an index term used to identify a record of information. It can consist of a word, phrase, or alphanumerical term. A weight $w_i \in \Re$ identifies the importance of descriptor c_i index term in relation to the record of information and therefore provides a refined model for evaluating the relationships between contexts. For example, we can have a descriptor $c_1 = Financial$, and $w_1 = 16$. A descriptor set is defined by a set of pairs, descriptors and weights defined by $\{\langle c_i, w_i\rangle\}_i$.

A *context* $\mathcal{C} = \{\{\langle c_{ij}, w_{ij}\rangle\}_i\}_j$ is a set of finite sets of descriptors c_{ij} from a domain $\mathcal{D}$ with appropriate weights w_{ij}, defining the importance of c_{ij}. For example, a context $\mathcal{C}$ may be a set of words (hence $\mathcal{D}$ is a set of all possible character combinations) defining a document Doc and the weights could represent the relevance of a descriptor to Doc. In classic Information Retrieval, $\langle c_{ij}, w_{ij}\rangle$ may represent the fact that the word c_{ij} is repeated w_{ij} times in Doc. Our work uses the context extraction algorithm of (Segev, 2005), which extracts contexts from documents using query expansion methods.

The context of a class is defined as a set of contexts describing instances that belong to this class. Unlike other work related to bootstrapping (Ciravegna et al., 2003), in our work contexts are not instances of ontology concept but rather rep-

resentatives thereof. Following (Segev and Gal, 2005), we define a class context $\mathcal{C}_{CL}$ of a class CL to be the union of its instance contexts.

3 Ontology Verification

Ontology verification is the process by which semantic relationships are identified by the ontology administrator. We term this process verification since we assume an ontology exists and may need to evolve. Therefore, semantic relationships in an ontology need to be continuously monitored and if necessary, revised. Here we use the work of (Noy and Klein, 2004) on ontology changes and assume a given closed set of operators OT, to be applied on a set of operands OD, taken from the set of all ontology elements. As an example, a change operator may be the *disjoint* operator, resulting in the creation of a semantic relationship called "disjoint" between two classes, given to it as operands.

Figure 1 provides a pictorial representation of the process. Formally, ontology verification is a function $OV : OT \times OD^* \rightarrow [0, 1]$. Ontology verification is given as input a hypothesis regarding the possible operator to be applied to one or more operands and returns a level of certainty μ regarding the truth in this hypothesis. A certainty of 1 indicates full certainty in the hypothesis, while a certainty of 0 means that the hypothesis is definitely incorrect. In Figure 1, the ontology verification function determines that the disjointedness of classes CL_1 and CL_2 has a certainty level of 0.9. An example of the use of the model can be that of an ontology administrator who would like to analyze a local government concept relationship. She supplies a set of documents representing two concepts, such as $Social\ Service$ and its subclass $Housing\ Service$, and receives a disjoint verification level of 0.8 in the eGovernment ontology based on this representative set of documents. A possible outcome of the verification might be adding this relationship into the ontology. In this work we aim at providing a support for the ontology administrator using automatically generated contexts.

4 Mapping Ontology Verification To Context

Having introduced ontology verification, we now focus on the details of change operators. Noy and Klein (Noy and Klein, 2004) describe a set of 22 ontology change operators and their impact on ontology elements (both classes and instances). For

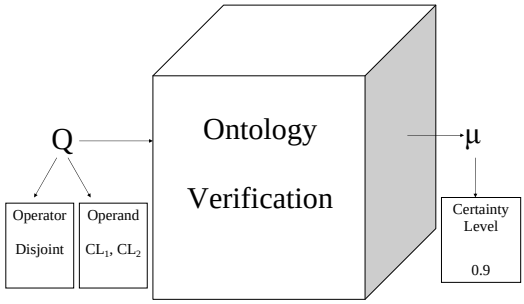

Figure 1: Ontology Verification Model

seven of these operators we show how verification can be accomplished using contexts. Verification of the remaining operators is left open for future research. The seven operators can be grouped into two groups as follows:

Hierarchy linking and restructuring In hierarchy linking one class becomes a superclass of another, making the latter a subclass of the former. Hierarchy restructuring involves reversing the roles of a superclass and a subclass.

Disjoining, merging, and splitting classes A disjoint operator involves adding a disjoint link between two classes. Merging two classes involves joining all instances of both classes into a single class. Splitting entails the partition of instances of one class into several classes.

For each operator group, we now provide an instantiation of the verification function, defining a level of certainty μ using context comparison. Two principles guide us in determining the specific formulae for computing levels of certainty. First, we ensure that extreme cases receive extreme certainty values (in particular, μ=0 and μ=1). The second principle is that of Occam's razor; thus we aim at generating as simple a formula as possible to specify intermediate values.

4.1 Hierarchy Linking and Restructuring

Let CL be a class and $I = \{I_1, I_2, ..., I_n\}$ be a set of instances. We denote by $\mathcal{C}_{CL}$ the context of a class CL and by $|\mathcal{C}_{CL}|$ the context cardinality. Similarly, for an instance I_i, we denote by $\mathcal{C}_i$ its context and by $|\mathcal{C}_i|$ the context cardinality.

Given two classes, CL_i and CL_j, if CL_i is a subclass of CL_j, then its context should be contained in the context of CL_j . This is because an instance of CL_i is also an instance of CL_j and therefore has a broader context than an instance of the superclass. Therefore, we compute the certainty of a hypothesis that CL_i is a subclass of

CL_j to be

$$\mu_{\textit{Sub-Sup}} = \frac{\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right|}{\left|\mathcal{C}_{CL_j}\right|}$$

In one extreme case, if $\mathcal{C}_{CL_j} \subset \mathcal{C}_{CL_i}$, then $\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right| = \left|\mathcal{C}_{CL_j}\right|$ and $\mu_{\textit{Sub-Sup}} = 1$. At the other extreme, if $\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j} = \emptyset$, then $\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right| = 0$ and $\mu_{\textit{Sub-Sup}} = 0$ as well. Removing a hierarchy link is a reverse operator of the linking operator. In the eGovernment ontology, we can see an example for classes $Organization$, $Hospital$, $Medical\ Social\ Center$, and $Municipal\ Center$ as having the *Sub-Sup* relation, where $Organization$ is the superclass. However, different μ values may relate to each of the subclasses of $Organization$. The ontology administrator can set a threshold for μ to determine the relationship validity.

When determining a hierarchy restructuring, we analyze the number of overlapping contexts between a subclass and a superclass, similarly to the *Sub-Sup* operator. Here, though, we reverse the roles of the subclass and superclass. Thus, given two classes CL_i and CL_j, CL_i is a subclass of CL_j, moving CL_i up the hierarchy (or symmetrically moving CL_j down the hierarchy) has a certainty level of

$$\mu_{\textit{up}} = \frac{\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right|}{|\mathcal{C}_{CL_i}|}$$

It is worth noting that $\mu_{\textit{up}}$ for moving CL_i up the hierarchy is the same as determining that CL_j is a subclass of CL_i. When moving up the hierarchy, one more decision may need to be taken. Assume that CL_j is a subclass of some third class CL_k. When we move CL_i up, we need to consider whether CL_i should become a subclass of CL_k. If this is indeed the case, then due to the transitivity of the IS-A relationship, CL_j should be disconnected from CL_k. In the previous example, when examining the $Municipal\ Center$ class an ontology administrator may decide that $Municipal\ Center$ should become a superclass of three sibling classes but not a superclass of the $Organization$ class.

4.2 Disjoining, Merging, and Splitting Classes

Two classes, CL_i and CL_j are disjoint if their contexts do not overlap. Therefore, we can compute the certainty of a *Disjoint* operator among two classes to be

$$\mu_{\textit{Disjoint}} = 1 - \frac{\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right|}{\left|\mathcal{C}_{CL_i} \cup \mathcal{C}_{CL_j}\right|}$$

If $\mathcal{C}_{CL_i} = \mathcal{C}_{CL_j}$, then $\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j} = \mathcal{C}_{CL_i} \cup \mathcal{C}_{CL_j}$ and thus $\mu_{\textit{Disjoint}} = 0$. If $\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j} = \emptyset$, then $\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right| = 0$ and thus $\mu_{\textit{Disjoint}} = 1$.

It is worth noting that while *Disjoint* is a symmetric operator, the hierarchy linking and restructuring (Section 4.1) are directional operators. This is reflected in the different methods for computing $\mu_{\textit{Sub-Sup}}$, $\mu_{\textit{up}}$, and $\mu_{\textit{Disjoint}}$, where the denominator for the former is $\left|\mathcal{C}_{CL_j}\right|$ (the super-class context cardinality) and for the latter is $\left|\mathcal{C}_{CL_i} \cup \mathcal{C}_{CL_j}\right|$.

When two classes are merged, their contexts are analyzed to identify overlappings. Thus, the merge operator can be considered the reverse operator of disjoint, yielding a certainty level computed as follows:

$$\mu_{\textit{Merge}} = 1 - \mu_{\textit{Disjoint}} = \frac{\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right|}{\left|\mathcal{C}_{CL_i} \cup \mathcal{C}_{CL_j}\right|}$$

There can be multiple variations of splitting two classes. In one variation, an ontology administrator provides the verification function and two groups of instances $I' = \{I_1, I_2, ..., I_m\}$ and $I'' = \{I_{m+1}, I_{m+2}, ..., I_n\}$, constituting a partition of the instances of a class CL. For such a variation, we need to generate a class context for I' and I'' and then determine if they are disjoint or not. Therefore, computing the certainty level of this variation of the *Split* operator is identical to that of the *Disjoint* operator.

In a second variation no instance of partitioning is given. We need first to determine the best partitioning of the class, using a statistical method such as the K-Means algorithm (with $K = 2$), and then compute $\mu_{\textit{Split}}$ as before. While the same function is used to determine class split and disjointedness, these operations are not equivalent. An ontology administrator is likely to determine to split classes with a lower level of certainty than that needed for establishing a disjoint relationship. An eGovernment example for *Merge* can be the unification of $Education$ with $School$.

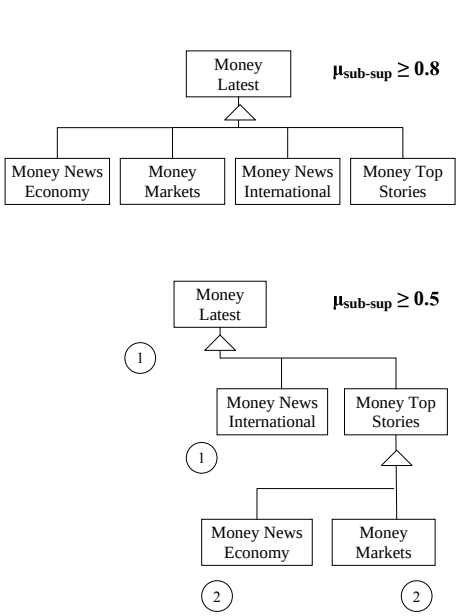

Figure 2: RSS Relations

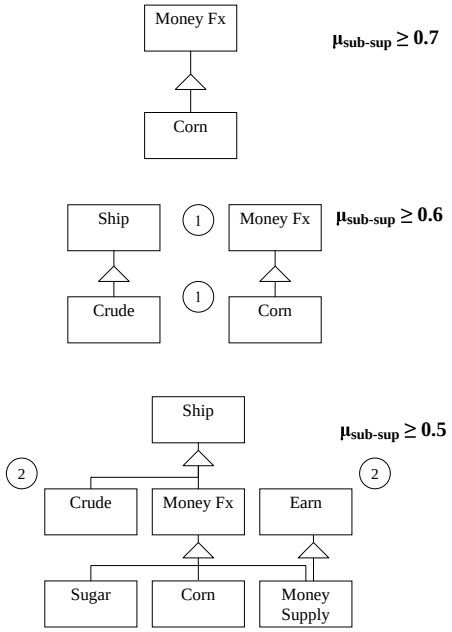

Figure 3: Reuters Relations

5 Experiences With Context Based Ontology Verification

Our experiences are based on data from Reuters and RSS news traces. In these data traces, data were originally partitioned to topics with no ontological relationships. The Reuters data set was taken from a publicly available trace (http://about.reuters.com/researchandstandards/corpus/). We chose 10 news topic categories (referred to hereafter as classes), for a total of 3,125 data, where a datum is a Reuters news article. The RSS trace was collected during August 2005 from the CNN Website. Here we chose 10 news topic categories for the total of 9,920 data, where each RSS news header or descriptor constitutes a datum. The main difference between the Reuters and RSS traces is the datum size. We generated a context for each datum and each class using an automatic context extraction algorithm (Segev, 2005). The number of context descriptors generated from each datum was set to 10. The data size per class used for RSS varied from 73 to 1,911 per class, while for Reuters it varied from 126 to 510.

In the experiment we calculated for each class the number of contexts that overlapped with the other seven classes. This asymmetric comparison gave, for any two classes CL_i and CL_j, the metric of $\left|\mathcal{C}_{CL_i} \cap \mathcal{C}_{CL_j}\right|$ and $\left|\mathcal{C}_{CL_i} \cup \mathcal{C}_{CL_j}\right|$. Given two classes and their associated contexts, we analyzed the certainty level of all the operators for all the class pairs, using the formulae given in Section 4.

We first analyze linking. Figure 2 presents the RSS class relations hierarchy created for $\mu_{\textit{Sub-Sup}} \geq 0.8$ and $\mu_{\textit{Sub-Sup}} \geq 0.5$. Similarly, Figure 3 presents the Reuters class relations hierarchy created for $\mu_{\textit{Sub-Sup}} \geq 0.7$, $\mu_{\textit{Sub-Sup}} \geq 0.6$, and $\mu_{\textit{Sub-Sup}} \geq 0.5$. As the value of $\mu_{\textit{Sub-Sup}}$ decreases, the hierarchy and the relations between the classes become more elaborated. For example, in the RSS data for $\mu_{\textit{Sub-Sup}} \geq 0.8$ the superclass $Money\ Latest$ has four subclasses. If we examine the same classes for a lower verification level of $\mu_{\textit{Sub-Sup}} \geq 0.5$ we receive a three level hierarchy. Similar results were observed for the Reuters data.

Analyzing the content of the data in the different classes of the Reuters data set, we notice that the data set $Corn$ actually discusses sale prices and purchase quantities and thus matches the relationship with $Money\ Fx$. Similarly, $Crude$ is discussed here as raw material, which has no direct relation with the class of $Earn$ but does have a relation with $Ship$, the shipment of $Crude$ material.

Comparing the results to the content analysis at a level of $\mu_{\textit{Sub-Sup}} \geq 0.5$, we see that $Ship$ is a superclass for all the shipment topics displayed, which include $Money\ Fx$ and $Crude$. $Money\ Fx$ includes another set of shipment and economic topics that form the subclasses $Sugar$, $Corn$, and $Money\ Supply$. $Earn$, conversely, includes only the subclass of $Money\ Supply$. $Money\ Fx$,

Class Sets	Merge	Super/Subclass	Disjoint
Money Latest	19.2%	86.7%	80.8%
Money News International	19.2%	19.8%	80.8%
Money News Economy	12.1%	19.5%	87.9%
Money Markets	12.1%	24.3%	87.9%

Table 1: Operator μ Verification RSS

Class Sets	Merge	Super/Subclass	Disjoint
Money Fx	15.4%	75.5%	84.6%
Corn	15.4%	37.9%	84.6%
Crude	12.8%	6.2%	87.2%
Earn	12.8%	42.3%	87.2%
Ship	22.0%	66.7%	78.0%
Crude	22.0%	23.0%	78.0%

Table 2: Operator μ Verification Reuters

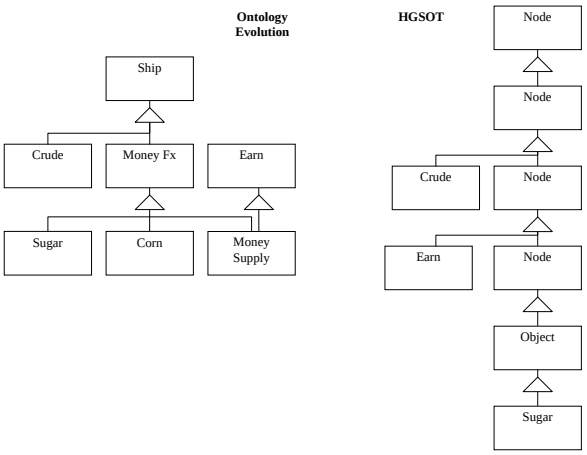

Figure 4: Reuters HGSOT Comparison

Money Foreign Exchange currency, is related to prices of $Corn$ and prices of $Sugar$ which are represented as concepts. However, $Crude$, raw material, has no direct relation to $Money$ although both concepts are related to $Ship$, shipping of material worldwide.

Table 1 compares the certainty level of three operators, namely *Merge*, *Superclass-Subclasss*, and *Disjoint*, for two class pairs in the RSS data set. When evaluating the classes $Money\ Latest$ and $Money\ News\ International$, there is a high $\mu_{\textit{Sub-Sup}}$ level and a low $\mu_{\textit{Merge}}$ level. Although there is also a high $\mu_{\textit{Disjoint}}$ level, it is not as high as $\mu_{\textit{Sub-Sup}}$ and thus it is likely that these two classes should be in a superclass-subclass relationship. When examining two sibling classes in the hierarchy, $Money\ News\ Economy$ and $Money\ Markets$, the $\mu_{\textit{Disjoint}}$ level is significantly higher than the other certainty levels, indicating a possible *Disjoint* relationship between the two classes.

Table 2 details results received for the Reuters data. It is worth noting that here, when comparing $Money\ FX$ and $Corn$, although there is a high $\mu_{\textit{Sub-Sup}}$ value, there is even a higher $\mu_{\textit{Disjoint}}$ level, which could indicate that these two classes should not be in a hierarchical relationship if we observe the highest μ value for all operations. However, an ontology administrator will probably prefer to assign a different μ value for each operator. Thus, for a $\mu_{\textit{Disjoint}} \geq 0.9$ and $\mu_{\textit{Sub-Sup}} \geq 0.5$ we receive the above relations. Similarly for classes $Ship$ and $Crude$ the same higher disjoint value and the above relations can be observed.

Figure 4 compares the results received by the ontology evolution method presented in this paper to those of the HGSOT (Khan et al., 2003) method implemented on a similar Reuters data set. We see that our algorithm achieves better qualitative results. To find an appropriate concept for each node in the hierarchy, HGSOT uses an automatic concept selection algorithm from WordNet. HGSOT uses the label Node for the nodes it cannot define and the label Object for the nodes WordNet cannot name specifically. As in our results, the connections between the $Sugar$, $Earn$, and $Crude$ nodes cannot be seen in the HGSOT results. However, the HGSOT algorithm does not supply any verification value to the hierarchy selected. Furthermore, the large number of hierarchies defined as $Node$ or $Object$ do not contribute to the ontology administrators' understanding of the ontology structure.

6 Conclusion

Continuous changes in organizations require frequent updates of the infrastructure of their information systems. Ontologies, which are used to model organization infrastructure, therefore need to be frequently updated and changed as well.

This work presents a model and a set of algorithms to semi-automatically support ontology relationship evolution using contexts. Given an ontology operator and operands, the model provides the quantification of the extent to which the relationship is valid. The model is validated by empirical analysis, using initial experiences with real-world RSS and Reuters news traces. These experiences show how relationships between the classes can be created and modified. Preliminary empirical results show that our model can assist ontology administrators in evolving ontologies.

To recap, the main contribution of this work is both conceptual and algorithmic. We present an ontology verification model, a quantified model for automatically assessing the validity of relationships in an ontology, and we provide a mapping of several ontology operators for determining relationships among classes.

The results of this work will be embedded as part of a European Commission eGovernment project. Future research will examine the model performance on eGovernment data and other large data sets. In addition, we plan on extending the model to include additional operators.

References

J. McCarthy. 1993. Notes on formalizing context. *In Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*.

T. R. Gruber. 1993. A translation approach to portable ontologies. *Knowledge Acquisition*, 5(2).

L. Khan and F. Luo and I-L. Yen. 2003. Automatic ontology derivation from documents. In *Conf. on Advanced Information Systems Engineering (CAISE)*.

F. Ciravegna and A. Dingli and D. Guthrie and Y. Wilks. 2003. Integrating information to bootstrap information extraction from web sites. In *Proceedings of IJCAI Workshop on Information Integration on the Web*.

N. F. Noy and M. Klein. 2004. Ontology evolution: Not the same as schema evolution. *Knowledge and Information Systems*, 6(4):428–440.

G. Tsatsaronis and R. Pitkanen and M. Vazirgiannis. 2005. Clustering for ontology evolution. In *Proceedings of the 29th Annual Conference of the German Classification Society (GfKl 2005)*.

A. Segev. 2005. Identifying the multiple contexts of a situation. In *Proceedings of IJCAI-Workshop Modeling and Retrieval of Context (MRC2005)*.

A. Segev and A. Gal. 2005. Putting things in context: A topological approach to mapping contexts and ontologies. In *Proceedings of AAAI-Workshop Workshop on Contexts and Ontologies: Theory, Practice and Applications*.