Representing and Reasoning with Modular Ontologies

Jie Bao and **Vasant Honavar**
Artificial Intelligence Research Laboratory
Computer Science Department
Iowa State University, Ames, IA USA 50011
Email: {baojie, honavar}@cs.iastate.edu

Abstract

[1] Many real world applications call for ontology language support for modular ontologies, distributed reasoning, and selective knowledge sharing. This paper summarizes a novel approach based on package-based description logics to address this need.

Introduction

Ontologies that explicitly identify objects and relationships of objects in specific domains of inquiry are essential for collaborations that involve sharing of data, knowledge, or resources among autonomous individuals or groups in open environments, such as the Semantic Web (Berners-Lee, Hendler, & Lassila 2001). Consequently, there has been a significant body of recent work on languages for specifying ontologies, e.g., OWL. With the growing number and scale of available ontologies, there is also an urgent need for modular ontology languages for the development, management and reasoning with large-scale ontologies. Representative applications of modular ontologies include the following:

- **Collaborative Ontology Building**. The building process of an ontology is usually a collaborative process that involves cooperation among multiple domain experts. Therefore, the ontology needs to capture the knowledge from multiple contributors. The building process of such an ontology involves the generation, management and integration of multiple components of the ontology in a principled fashion.
- **Partial Ontology Reuse**. Knowledge bases, and in particular ontologies, are very likely to be reused. For example, a wine ontology may reuse knowledge about grape in a food ontology. However, the lack of modularity in current ontology languages forces an ontology to be totally reused or not be reused at all. An ontology has to be completely reused even if only a small fragment of it is actually needed. Modular ontologies will facilitate more flexible and efficient reuse of existing ontologies (Bao, Caragea, & Honavar 2006d).
- **Selective Knowledge Hiding**. In many applications, the provider of an ontology may not wish, because of copyright considerations, privacy or security concerns, to make the entire ontology directly visible to all users, while willing to expose certain parts of the ontology to certain subsets of users. In other cases, an ontology component may only provide a limited query interface, while the details of the component are not important to users and other ontology components. Both the two cases call for knowledge scope control in modular ontologies (Bao, Caragea, & Honavar 2006d).
- **Distributed Data Management**. Applications involving distributed data sources require explicit description of the data semantics for the data source. For example, a data integration problem may need ontologies to describe the schema and data content semantics of tables in multiple relational databases. Instead of a single global ontology, such applications require distributed, connected, multiple ontologies, each capturing a subset of the domain of discourse.

In contrast, "The current state of the art in ontology engineering is reminiscent of the state of programming languages nearly four decades ago: Ontology languages are largely unstructured, with no support for restricting the scope of knowledge, and limited support for ontology modules, and hence little support for reuse of knowledge" (Bao, Caragea, & Honavar 2006d). The characteristics of web ontologies demand that next generation ontology languages need to support collaborative construction, selective sharing and use of ontologies.

Against this background, this paper introduces the framework of package-based ontologies to meet this need. We present a novel family of modular ontology languages, Package-based Description Logics (P-DL), that support modularity and selective knowledge sharing in ontologies as well as associated reasoning algorithms.

Representing and Reasoning with Modular Ontologies

P-DL language features are aimed at providing fine-grained modular organization of an ontology and selective knowledge disclosure, to meet the requirements of applications outlined in the previous sections.

[1]This research was supported in part by the NSF award 0639230

In P-DL, an ontology is composed of a set of ontology modules or *packages*. Terms and axioms are defined in specific home packages.

Definition 1 (Package) *Let $O = (S, A)$ be an ontology, where S is the set of terms and A is the set of axioms over terms in S. A package $P = (\Delta_S, \Delta_A)$ of the ontology O is a fragment ofO, such that $\Delta_S \subseteq S$, $\Delta_A \subseteq A$. A term $t \in \Delta_S$ or an axiom $t \in \Delta_A$ is called a* member *of P, denoted as $t \in P$. P is called the* home package *of t, denoted as $\mathcal{HP}(t) = P$.*

Terms can be names of concepts, roles, or individuals. A package can *import* terms defined in another package.

Definition 2 (Foreign Term and Importing) *A term t that appears in a package P, but has a different home package Q is called a* foreign term *in P. We say that P* imports *$Q : t$ and denote thisas $Q \xrightarrow{t} P$. If any term de ned in Q is imported into P, we say P imports Q, and denote it by $Q \mapsto P$.*

Foreign terms can be used to construct local concepts. All concepts except for atomic foreign concepts in package $P_i$are i-concepts. Therefore, although P_i can import concepts in P_j $(i \neq j)$, i-concepts are still disjoint with j-concepts. For example, an ontology O has two packages:

$$\mathbf{P_{Animal}}$$

(1a) $1 : Dog \sqsubseteq 1 : Carnivore$
(1b) $1 : Carnivore \sqsubseteq \forall 1 : eats.(1 : Animal)$

$$\mathbf{P_{Pet}}$$

(2a) $2 : PetDog \sqsubseteq 1 : Dog \sqcap 2 : Pet$
(2b) $2 : PetDog \sqsubseteq \exists 1 : eats.(2 : DogFood)$

We will omit the prefix when there is no confusion. Both $1 : Dog \sqcap 2 : Pet$ and $\exists 1 : eats.(2 : DogFood)$ are2-concepts constructed using some foreign terms. We denote the package extension to DL as $\mathcal{P}$. For example, $\mathcal{ALCP}$ is the package-based version of DL$\mathcal{ALC}$. $\mathcal{P}_{\mathcal{C}}$ denotes a restricted type of package extension which only allows acyclic import of concept names (Bao, Caragea, & Honavar 2006d).

P-DL also allows nesting of a package within another package. Consequently, a P-DL ontology may have a fine-grained organizational structure, i.e. a package hierarchy. Such an organizational structure, in addition to the semantic structure of the ontology, provides the possibility for flexible partial reuse of an ontology and the management of an ontology in collaborative environments.

Due to the distributed nature of modular ontologies, integrating the ontology modules into a centralized ontology is expensive or infeasible. In such a setting, there is a need for *distributed* reasoning wherein the inference is not centralized, but divided into several local reasoning processes, one for each ontology module. Distributed reasoning, in addition to improving the scalability of the reasoning process, also helps ensure the autonomy of each ontology module.

The reasoning algorithm for P-DL is an extension of existing DL tableau algorithms and is motivated by the localized semantics of P-DL (Bao, Caragea, & Honavar 2006d). For each package in a P-DL, we have the model-theoretic local interpretation of the package:

Definition 3 (Local Interpretation) *A local interpretation of a package P is a pair $\mathcal{I}_P =< \Delta^{\mathcal{I}_P}, (.)^{\mathcal{I}_P} >$, where $\triangle^{\mathcal{I}_P}$ the local domain (set of all objects) and $(.)^{\mathcal{I}}_P$ is a function that maps each concept name C to $C^{\mathcal{I}_P} \subseteq \Delta^{\mathcal{I}_P}$; each role name R to $R^{\mathcal{I}_P} \subseteq \Delta^{\mathcal{I}_P} \times \triangle^{\mathcal{I}_P}$, and each individual name i to $i^{\mathcal{I}_P} \in \Delta^{\mathcal{I}_P}$.*

Local domains of packages in a P-DL ontology may be partially overlapping. We may have a conceptual global interpretation for the (virtually) integrated ontology from all modules, with each local interpretation as a*projection* of it:

Definition 4 (Global Interpretation) *A global interpretation of a set of packages $\{P_i\}$ with local interpretations $\mathcal{I}_i = \langle \Delta^{\mathcal{I}_i}, (.)^{\mathcal{I}_i} \rangle$, $i = 1, \cdots m$ is$\mathcal{I}_g = \langle \Delta^{\mathcal{I}_g}, (.)^{\mathcal{I}_g} \rangle$, where $\Delta^{\mathcal{I}_g} = \cup_{i=1}^{m} \Delta^{\mathcal{I}_i}$ and $(.)^{\mathcal{I}_d}$ maps each concept name C to $C^{\mathcal{I}_g} \subseteq \Delta^{\mathcal{I}_g}$; each role name R to $R^{\mathcal{I}_g} \subseteq \Delta^{\mathcal{I}_g} \times \Delta^{\mathcal{I}_g}$, and each individual name i to $i^{\mathcal{I}_g} \in \Delta^{\mathcal{I}_g}$.*

Each $\mathcal{I}_i$ is called a projection *of $\mathcal{I}_g$. We have: (1)$\Delta^{\mathcal{I}_i} \subseteq \Delta^{\mathcal{I}_g}$; and (2) for each concept or role name t, $t^{\mathcal{I}_i} \subseteq t^{\mathcal{I}_g}$ and for each individual name t, $t^{\mathcal{I}_i} = t^{\mathcal{I}_g}$. Such a relation is denoted as $(.)^{\mathcal{I}_i} \subseteq (.)^{\mathcal{I}_g}$.*

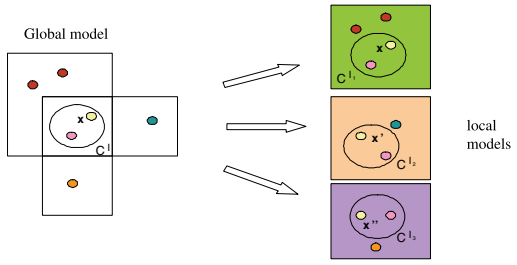

Figure 1: Model Projection

Such a projection relation is illustrated in Figure 1: local domains of each package are not necessarily disjoint, i.e., $\Delta^{\mathcal{I}_i} \cap \Delta^{\mathcal{I}_j} \neq \emptyset$; shared individuals in local domains can be merged to obtain the global interpretation. On the other hand, each local interpretation can be seen as a "projection" of the global interpretation w.r.t. the terminology in the corresponding package.

Classical tableau algorithms preform reasoning with a DL ontology by constructing an interpretation (i.e., a model) for that ontology. The reasoning algorithm for P-DL (Bao, Caragea, & Honavar 2006c) is driven by the above fact that a conceptual global model for the whole ontology can be constructed from multiple local models. Thus, a global tableau, which is the presentation of a global model, can also be constructed from multiple projections of it (Figure 2).

P-DL reasoning algorithm allows each local projection of the global tableau, called a local tableau, to be created and maintained by a local reasoner. Thus, reasoning is carried out by a federation of reasoners instead of a single reasoner. Local tableaux may share individuals (since local domains may be partially overlapping) and correspondence between packages that share such individuals can be established by a set of messages. The advantage of this approach is that the tableau expansion rules for P-DL can be easily obtained from the existing tableau expansion rules for classical DLs.

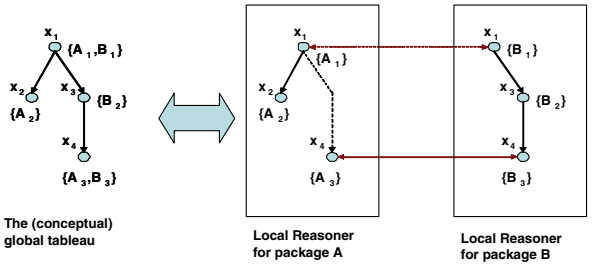

Figure 2: Tableau Projection

For example,for the P-DL $\mathcal{ALCP}_{\mathcal{C}}$, we may change the $\mathcal{ALC}$ expansion rules as the following (Bao, Caragea, & Honavar 2006c) (in what follows, x is an individual (node) in the tableau and C is a concept):

- For each class label test query $C \in L(x)$ in $\mathcal{ALC}$ rules, we may replace it with a *membership query message* $m(x, C)$ that being sent to the reasoner of $\mathcal{HP}(C)$.
- For each class label appending operation $L(x) = L(x) \cup \{C\}$, we may replace it with a *reporting message*: if there is a copy of x (denoted as x') in the local tableau of $P_i = \mathcal{HP}(C)$, then $L_i(x') = L_i(x') \cup \{C\}$, otherwise create a x' with $L_i(x') = \{C\}$.

P-DL semantics ensures that distributed reasoning with a modular ontology will yield the same conclusion as that obtained by a classical reasoning process applied to an integration of the respective ontology modules. Additional details of the reasoning algorithm are available in (Bao, Caragea, & Honavar 2006c).

Representing and Reasoning with Hidden Knowledge

Traditional ontology languages allow all terms and axioms in an ontology to be globally visible. However, an open and collaborative environment also requires support for selective limiting of term and axiom *scopes* to protect the autonomy of each ontology module and the knowledge contained in it.

For example, personal information, such as calendars, may be shared on the web. However,people may also need to protect sensitive personal information from unintended disclosure. For example, suppose Bob uses an online calendar to coordinate his daily activities with others. He may wish to show the schedule information at different levels of granularity to different individuals. For example, if Bob has a date at a certain time, he may not wish to share the details of the date with the public. However, he might be willing to share information that he is busy at that time to the public.

Another example is offered in (Farkas, Brodsky, & Jajodia 2005): Jane may need to take certain medicine for preventing breast cancer. However, she does not want the pharmacy to provide her health insurance company complete information about the prescription, since the medicine she has been prescribed is only used by people who are have a high risk of developing breast cancer. Jane might be worried that the insurance company may infer that she has a high risk of suffering from cancer in the future, and increase her insurance premium.

Considerations such as those illustrated by the above examples lead us to associating *scope limitation modi ers* (SLM) with terms and axioms defined in a package (Bao, Caragea, & Honavar 2006d). A SLM controls the visibility of the corresponding term or axiom to entities on the web,in particular, to other packages.

Definition 5 (SLM) *The scope limitation modi er of a term or an axiomt_K in package K is a boolean function $f(p, t_K)$, where p is a URI of an entity, the entity identi ed by p can access t_K iff $f(p, t) = true$. We denote this by $t_K \in_f K$.*

An entity on the web can be a user (e.g. emailto://baojie@cs.iastate.edu), a web program (e.g.http://www.foo.com/query.jsp), or another package (e.g. http://boole.cs.iastate.edu/animal/pet.powl). For example, we can define SLMs as follows:

- $\forall p, public(p, t) := true$, means t is accessible everywhere.
- $\forall p, private(p, t) := (t \in p)$, means t is visible only to its home package.

Other SLMs can also be specified as needed. The set of all SLMs in a package is called the *scope policy* of the package.

Scope limitation in P-DL is different from access control (e.g., XACML (Godik & Moses 2002)) or encryption of ontology (e.g.,partial encryption of RDF (Giereth 2005)) in several ways. Those efforts are aimed at safe access on*syntactic* level. On the other hand, SLM in P-DL aims at knowledge hiding on *semantic* level, where the hiding is not *total*, but *partial*, because not only a subset of the axioms is invisible to other parties, but also the semantics of such axioms is partially hidden from disclosure but is available for answering queries that are *safe* in the sense that they do not compromise the hidden knowledge.

The difference between *total hiding* and *partial hiding*, can be illustrated by considering an alternative approach in which the private axioms are totally hidden from other ontology modules. Hence, such axioms cannot be used in the reasoning process. In the online calendar example, if Bob makes information about his activity at some time t completely hidden, a reasoner cannot arrive at the conclusion that he is busy at that time. On the other hand, a P-DL version of the same ontology can model the same knowledge as follows:

public: $Dating \sqsubseteq Activity$
private: $t \sqsubseteq \exists hasActivity.Dating$

Then a query $t \sqsubseteq \exists hasActivity.Activity$ (there is an activity at time t) can still be safely answered as YES, but the hidden knowledge $t \sqsubseteq \exists hasActivity.Dating$ may be answered as unknown. That is, knowledge is *concealable* but not forbidden from *safe* use. We refer to such reasoning in P-DL with SLMs as *concealable reasoning*.

For a web entity k, a P-DL ontology K can be divided into a visible part K_v and a hidden part K_h, such that for any axiom $t \in K_h$, SLM $f(k, t) = false$ and for any

$t' \in K_v$, SLM $f(k, t') = true$. Figure 3 shows the general concealable reasoning process. For a given query α to an ontology$K$, if any hidden knowledge γ can be inferred from α and the visible part K_v, the reasoner should deny answering the query. Otherwise, the reasoner should answer a safe query in the same way as a classical reasoner.

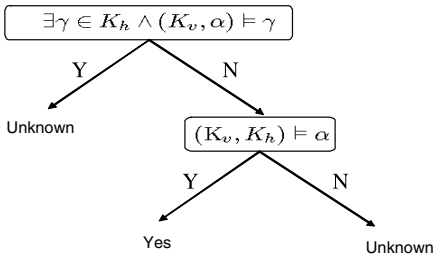

Figure 3: Concealable Reasoner

When the reasoner answers "unknown" to a query α, it appears as if the ontology has incomplete knowledge aboutα. Formally, assume a query to a DL ontology is an entailment query $C \sqsubseteq D$, we have the definition:

Definition 6 (Complete Knowledge) *A P-DL ontology O has complete knowledge about concept subsumption $\alpha : (i : C \sqsubseteq j : D)$, if for every model $\{\mathcal{I}_i\}$ of O, $C^{\mathcal{I}_i} \subseteq D^{\mathcal{I}_j}$; otherwise, it has incomplete knowledge about α.*

For example, for the animal ontology given in the previous section, it has complete knowledge about $2 : PetDog \sqsubseteq 1 : Carnivore$, but incomplete knowledge about $2 : PetDog \sqsubseteq 1 : Animal$. We adopt the "Open World Assumption" (OWA). Hence, failing to prove the negation of an assertion does not imply the validity of the assertion. Hence, a user cannot tell from the answers of a concealable reasoner whether the ontology contains hidden knowledge or incomplete knowledge with regard to the answer for the (entailment) query.

To ensure the integrity of concealable reasoning with a P-DL ontology, we require that the ontology must have *safe scope policy*, such that any hidden knowledge can not be inferred from the visible part of the ontology. We assume each ontology has a set of concept entailment axioms and an assertion is an entailment query to the ontology. We have the definition:

Definition 7 (Safe Scope Policy) *An ontology $K = \{K_v, K_h\}$ is said to have a safe scope policy if for any γ, $\gamma \in K_h \rightarrow K_v \not\models \gamma$.*

Hence, an ontology has safe scope policy (w.r.t. certain web entity) if its visible part does not entail the hidden part of the ontology, i.e., $K_v \not\models K_h$. Thus, knowledge can be partially hidden while being available for use in safe inference.

Conclusions

In this paper, we have presented P-DL, a modular ontology language that can model modular ontologies with and selective knowledge hiding. We have introduced a distributed reasoning algorithm for P-DL that can reason with modular ontologies without integrating them into a single ontology. We have introduced the notion of concealable reasoning in P-DL that allows ontology modules to use hidden knowledge to answer queries without inadevertantly exposing the hidden knowledge.

Related work of interest includes Distributed Description Logics (DDL) (Borgida & Serafini 2002) and E-Connections (Grau, Parsia, & Sirin 2004). P-DL is more expressive than the both formalism (Bao, Caragea, & Honavar 2006a). P-DL avoids some of the inference difficulties that DDL and E-connections (Bao, Caragea, & Honavar 2006b). P-DL is unique in its support for selective knowledge hiding and safe reasoning with hidden knowledge.

Work in progress includes the implementation of the reasoning algorithms presented in this paper, as well as an OWL-compatible syntax for P-DL.

References

[Bao, Caragea, & Honavar 2006a] Bao, J.; Caragea, D.; and Honavar, V. 2006a. Modular ontologies - a formal investigation of semantics and expressivity. In *R. Mizoguchi, Z. Shi, and F. Giunchiglia (Eds.): Asian Semantic Web Conference 2006, LNCS 4185*, 616–631.

[Bao, Caragea, & Honavar 2006b] Bao, J.; Caragea, D.; and Honavar, V. 2006b. On the semantics of linking and importing in modular ontologies. In *accepted by ISWC 2006 (In Press)*.

[Bao, Caragea, & Honavar 2006c] Bao, J.; Caragea, D.; and Honavar, V. 2006c. A tableau-based federated reasoning algorithm for modular ontologies. In *Accepted by 2006 IEEE/WIC/ACM International Conference on Web Intelligence (In Press)*.

[Bao, Caragea, & Honavar 2006d] Bao, J.; Caragea, D.; and Honavar, V. 2006d. Towards collaborative environments for ontology construction and sharing. In *International Symposium on Collaborative Technologies and Systems (CTS 2006)*. IEEE Press. 99–108.

[Berners-Lee, Hendler, & Lassila 2001] Berners-Lee, T.; Hendler, J.; and Lassila, O. 2001. The semantic web. *Scientific American* 284(5):34–43.

[Borgida & Serafini 2002] Borgida, A., and Serafini, L. 2002. Distributed description logics: Directed domain correspondences in federated information sources. In *CoopIS*, 36–53.

[Farkas, Brodsky, & Jajodia 2005] Farkas, C.; Brodsky, A.; and Jajodia, S. 2005. Unauthorized inferences in semi-structured databases. In *Submitted to Information Sciences*.

[Giereth 2005] Giereth, M. 2005. On partial encryption of rdf-graphs. In Gil, Y.; Motta, E.; Benjamins, V. R.; and Musen, M. A., eds., *International Semantic Web Conference*, volume 3729 of *Lecture Notes in Computer Science*, 308–322. Springer.

[Godik & Moses 2002] Godik, S., and Moses, T. 2002. Oasis extensible access control markup language (xacml). OASIS Committee Secification cs-xacml-specification-1.0, November 2002, http://www.oasis-open.org/committees/xacml/.

[Grau, Parsia, & Sirin 2004] Grau, B. C.; Parsia, B.; and Sirin, E. 2004. Working with multiple ontologies on the semantic web. In *International Semantic Web Conference*, 620–634.