Towards Theory Translation [⋆]

Dejing Dou[1] and Drew McDermott[2]

[1] Computer and Information Science
University of Oregon
Eugene, OR 97403, USA
`dou@cs.uoregon.edu`

[2] Computer Science Department
Yale University
New Haven, Connecticut 06520, USA
`drew.mcdermott@yale.edu`

Abstract. Ontologies play a key role in agent communication and the emerging Semantic Web to define a vocabulary of concepts and their relationships. Different agents and web services may use vocabularies from different ontologies to describe their data. The current research on ontology mapping and ontology translation mainly focuses on how to map and translate vocabularies and associated data instances from one ontology to another. However, more complicated true statements, such as axioms (rules), are used or being developed to describe the relationships among the concepts. When extending one ontology using complicated true statements (theory) from another, we must confront the problem of *theory translation*, which is difficult because of the *asymmetry of translation.* In this paper, using an inferential approach we call *axiom derivation*, we show how to translate complex axioms between different time ontologies. We also prove the validity of our algorithm.

1 Introduction

Ontologies, which can be defined as the formal specification of a vocabulary of concepts and the relationships among them, play a key role in agent communication and the emerging Semantic Web [3]. Multiple agents and Semantic Web services often need to share data, in spite of the fact that they describe similar domains using different vocabularies, or use the same symbols to mean different things. This *semantic heterogeneity* problem has received significant attention. (See [21] for a survey.) We can distinguish several different problems:

1. *Ontology matching and mapping:* Finding correspondences (matchings) and mappings between the concepts of two ontologies. The mappings are often equivalences, subclass-superclass (or subproperty-superproperty) relationships and other more complicated relationships.

⋆ This work was partly supported by the DARPA DAML project, under contract F30602-00-2-0600. This is an invited paper that is extended from our short paper "Deriving Axioms across Ontologies" in AAMAS2006 [12].

2. *Ontology translation:* Translating a dataset (assertions) or a query expressed using one ontology (or set of ontologies) into a form that uses a different ontology (or set).
3. *Theory translation:* Translating more complicated true statements (theory), such as axioms, of one ontology into the vocabulary of another while preserving their validity.

Significant works [10, 17, 20, 5] cover the first category while others [15, 7, 9, 14, 8] concern the second. In this paper we focus on the third.

Ontologies constrain the meanings of vocabularies by expressing relationships among their symbols, such as subset-superset links, role declarations, and the like. Complex relationships can be expressed only with logical theory, such as axioms (rules). For example, to overcome the expressivity limitation of OWL (Web Ontology Language [2]), the research on OWL Rule languages [16] has proposed a Horn clause rules extension to represent the relationships among properties. On the other hand, the semantic mappings between the concepts of different ontologies also can be represented as logical axioms. For example, the research on MAFRA [17] and C-OWL [4] can represent the mappings between the contents of different ontologies by extending DAML+OIL [1] and OWL syntax and semantics. Our previous work on OntoMerge [13, 14] has used first order bridging axioms to represent semantic mappings. It is well known that complex mapping rules can not be generated fully automatically; human experts may need to help the process based on the mapping suggestions from automatic or semi-automatic ontology mapping tools such as those described in [10, 17, 20, 5].

Normally, the data instances and queries can be automatically translated from the source ontology to the target ontology once the semantic mappings have been discovered and presented to translators [7, 9, 14, 8]. However, little research has been done on translating the axioms or other complicated true statements (theory) automatically, which raises the possibility that important semantic constraints will be lost in translation. It is important to represent theories in different vocabularies and preserve their validity.

In this paper, we will first use one inferential framework to unify the ontology translation and theory translation for facts, queries, axioms and other complicated true statements. One assumption of our framework is that the mappings between the concepts from different ontologies are represented as Horn-like mapping rules, which we call *bridging axioms*. We point out that it is the *asymmetry of translation* that makes the translation of axioms and other complicated theories difficult. We then describe an algorithm called *axiom derivation* for translating theories, such as those axioms in time ontologies, from one ontology to another by both forward and backward chaining. We also prove the validity of our algorithm.

2 Framework and Previous Work

2.1 Merged Ontologies & Bridging Axioms

In this section we briefly review our previous work on ontology translation. We assume that in order to translate facts from one ontology to another there must be a *merged ontology* in which all the relevant symbols are allowed to interact. For example, consider a pair of genealogy ontologies $Gonto_1$ and $Gonto_2$. The former uses the symbols husband and married while the latter uses male_partner, spouse, and in_marriage. The matchings (correspondences) between them can be represented as:

husband $\longrightarrow$ male_partner, spouse
married $\longrightarrow$ in_marriage

but these correspondences only hint at the actual semantic relationships, which can be expressed exactly using these axioms (rules):

$$(\forall x, y)\mathsf{husband}(x, y) \rightarrow \mathsf{male\_partner}(x, y) \tag{1}$$

$$(\forall x, y)\mathsf{husband}(x, y) \leftrightarrow \mathsf{spouse}(x, y) \tag{2}$$

$$(\forall x, y)\mathsf{married}(x, y) \leftrightarrow \mathsf{in\_marriage}(x, y) \tag{3}$$

where x and y are universal variables to represent male and female respectively. We call these axiomatic mapping rules *bridging axioms*. We have developed *Web-PDDL* as a strongly typed first-order logic language with Lisp-like syntax, to express Horn-like bridging axioms. These axioms are embedded in a *merged ontology* complete with namespace declarations and type-equivalence rules. For example, to represent axiom 2, we use the following Web-PDDL expression:

```
(forall (x - @Go1:Male y - @Go1:Female)
    (iff  (@Go1:husband x y)
          (@Go2:spouse x y)))
```

where Go1 and Go2 are prefixes of $Gonto_1$ and $Gonto_2$. These correspond to XML namespaces, and when Web-PDDL is translated to RDF [19], that is exactly what they become. The hyphen notation is used to declare a constant or variable to be of a type T, by writing "`x - `T".

However, unlike typical Horn clauses, bridging axioms can have conjunction of predicates on the conclusion side (see more examples later in Section 5). The predicates in bridging axioms can have built-in (but non-recursive) functions as arguments. Based on our previous experience with mapping different pairs of ontologies, this form is expressive enough for most mappings found by mapping tools or human experts.

2.2 Inferential Ontology Translation

We will use the symbol $\rightsquigarrow$ to indicate translation: $\alpha \rightsquigarrow \beta$ means that β is the translation of α. We call the ontology O_s that α uses the *source* ontology and O_t, the one β uses, the *target*. In the case of sets of assertions ("datasets"), we stipulate that the translation of α_d is simply the strongest set of assertions, β_d, in O_t entailed by α_d. A consequence of this stipulation is that

$$(KB; \alpha_d) \rightsquigarrow \beta_d \textit{ only if } (KB; \alpha_d) \vDash \beta_d$$

where we add to the left-hand sides the symbol KB to refer to the merged ontology which includes bridging axioms (mapping rules). It means we use entailment ($\vDash$) to define dataset translation: if all the bridging axioms in KB and all the assertions in α_d are true, then all the assertions in β_d should be true. Alternatively, we say that β_d is a logical (or semantic) consequence of KB and α_d. The only way to guarantee this entailment is to use sound inference ($\vdash$) in first order theory. In other words, "$\rightsquigarrow$" entails soundness, so we actually can use $\vdash$ to implement dataset translation:

$$(KB; \alpha_d) \rightsquigarrow \beta_d \;\Leftrightarrow\; (KB; \alpha_d) \vdash \beta_d \;\Rightarrow\; (KB; \alpha_d) \vDash \beta_d$$

This definition means that β_d is the largest set of assertions that can be derived from KB and α_d by inference.

Similarly, if α_q is a query in O_s, its translation is a query β_q in O_t such that any answer (set of bindings) to β_q is also an answer to α_q. In other words:

$$(KB; \alpha_q) \rightsquigarrow \beta_q \textit{ only if } (KB; \theta(\beta_q)) \vDash \theta(\alpha_q)$$

for any substitution θ, which is from the facts in the target database. It means we still use entailment ($\vDash$) to define the query translation. It is easy to get, for any substitution θ,

$$\begin{aligned}(KB; \alpha_q) \rightsquigarrow \beta_q \;&\Leftrightarrow\; (KB; \theta(\beta_q)) \vdash \theta(\alpha_q)\\ &\Rightarrow\; (KB; \theta(\beta_q)) \vDash \theta(\alpha_q)\end{aligned}$$

where a sound inference ($\vdash$) can actually implement and guarantee the entailment. The point is that β_q need not be (and seldom is) *equivalent* to α_q, in the sense that any answer to one is an answer to the other. All we need is that any answer to β_q be an answer to α_q. If we take O_s to be $Gonto_2$ and O_t to be $Gonto_1$, the query $\mathsf{male\_partner}(?x, ?y)$ in $Gonto_2$ will be translated into the query $\mathsf{husband}(?x, ?y)$ in $Gonto_1$. But the set of all husbands is not equivalent to the set of all male partners, since husbands are only one kind of male partners.

In order to use bridging axioms for inferential ontology translation, we built a special purpose first-order theorem prover called OntoEngine. OntoEngine has both forward chaining and backward chaining reasoners using generalized modus ponens [22]. The dataset translation can be implemented by a forward chaining reasoner and the query translation can be implemented by a backward chaining reasoner. Our framework has been evaluated by using OntoEngine on several real ontology translation tasks. Some of them need OntoEngine to process large sets of data. The results of experiments show that the translation for both data and queries works efficiently [14, 11].

3 Asymmetry and Composition of Theory Translation

3.1 Asymmetry of Translation

In the previous section, we have shown that a genealogy ontology $Gonto_1$ has two concepts (properties): husband and married. There may be a first-order logic axiom to describe their relationship:

$$(\forall x, y)\mathsf{husband}(x, y) \rightarrow \mathsf{married}(x, y) \tag{4}$$

We also know that another genealogy ontology $Gonto_2$ has male_partner, spouse and in_marriage as mapped properties. Some facts (assertions) expressed in the language of $Gonto_1$, can be translated into the language of $Gonto_2$ by simply replacing corresponding properties:

$$\mathsf{husband}(\mathsf{John}, \mathsf{Mary}) \rightsquigarrow \mathsf{male\_partner}(\mathsf{John}, \mathsf{Mary})$$

$$\mathsf{husband}(\mathsf{John}, \mathsf{Mary}) \rightsquigarrow \mathsf{spouse}(\mathsf{John}, \mathsf{Mary})$$

$$\mathsf{married}(\mathsf{John}, \mathsf{Mary}) \rightsquigarrow \mathsf{in\_marriage}(\mathsf{John}, \mathsf{Mary})$$

The translations are correct in terms of the semantics of $Gonto_1$ and $Gonto_2$, where husband can be thought of as spouse and a special kind of male_partner. However, if we use the same technique to translate the axiom (4) of $Gonto_1$ to $Gonto_2$, we get:

$$(\forall x, y)\mathsf{male\_partner}(x, y) \rightarrow \mathsf{in\_marriage}(x, y) \tag{5}$$

$$(\forall x, y)\mathsf{spouse}(x, y) \rightarrow \mathsf{in\_marriage}(x, y) \tag{6}$$

It is obvious that (5) is not always true since a man is a partner of a woman doesn't mean that he must be in a marriage with her; (5) is not a valid axiom in $Gonto_2$. But (6) is true as an axiom in $Gonto_2$. Why is it that the translation from (4) to (6) is correct, but the translation from (4) to (5) is not correct?

Given our inferential ontology translation framework, translation exhibits certain asymmetries that one must be wary of. Query translation is different from assertion translation. We will subscript the symbol $\rightsquigarrow$ with a "Q" to indicate the query case ($\rightsquigarrow_Q$), and with a "D" (for "data") to indicate the assertion or dataset case($\rightsquigarrow_D$). (We leave the subscript off in those cases where the context allows either reading.) In addition, if β_t is the translation of α_s:

$$(KB; \alpha_s) \rightsquigarrow \beta_t$$

that doesn't mean α_s is the translation of β_t:

$$(KB; \beta_t) \rightsquigarrow \alpha_s$$

Slightly less obviously, if $(KB; P) \rightsquigarrow Q$ we can't conclude $(KB; \neg P) \rightsquigarrow \neg Q$.

3.2 Composition of Theory Translation

Instead (not surprisingly), negation ends up involving the same duality as query translation. Assume that R is an expression which can be derived from KB and $\neg P_s$ by inference. Using the deduction theorem in first-order logic and considering that $\neg P_s \to R$ is equivalent to $\neg R \to P_s$, we know that

$$\begin{aligned}(KB; \neg P_s) \vdash R \Leftrightarrow\ & KB \vdash (\neg P_s \to R)\\ \Leftrightarrow\ & KB \vdash (\neg R \to P_s)\\ \Leftrightarrow\ & (KB; \neg R) \vdash P_s\end{aligned}$$

This gives us a way to translate negations. We can think of P_s as a "ground query" ($\theta(P_s) = P_s$): Given P_s, try to find a $Q_t^{'}$, which satisfies $(KB; Q_t^{'}) \vdash P_s$. But this is just the problem of translating the query P_s: $(KB; P_s) \rightsquigarrow_Q Q_t^{'}$.

Therefore, if the query translation of P_s is $Q_t^{'}$, $\neg Q_t^{'}$ can be derived from KB and $\neg P_s$ by the data translation and vice versa:

$$\begin{aligned}(KB; P_s) \rightsquigarrow_Q Q_t^{'} \Rightarrow\ & (KB; \neg P_s) \rightsquigarrow_D \neg Q_t^{'}\\ (KB; P_s) \rightsquigarrow_D Q_t^{'} \Rightarrow\ & (KB; \neg P_s) \rightsquigarrow_Q \neg Q_t^{'}\end{aligned}$$

Theory, such as axioms, are usually more complex than a typical dataset element, and it would be useful if we could attack this complexity by translating the pieces of a complex formula and composing the results. The presence of asymmetry means that care is required in doing the composition. For conjunctions and disjunctions, composition of translation is straightforward. It is easy to show that if we know that

$$(KB; P_{s1}) \rightsquigarrow Q_{t1}\ ;\ (KB; P_{s2}) \rightsquigarrow Q_{t2}$$

then

$$\begin{aligned}(KB; P_{s1} \wedge P_{s2}) \rightsquigarrow Q_{t1} \wedge Q_{t2}\\ (KB; P_{s1} \vee P_{s2}) \rightsquigarrow Q_{t1} \vee Q_{t2}\end{aligned}$$

But when we encounter a negation we must flip from "D" mode to "Q" mode or vice versa.

Since every complicated true statement (theory) in first-order can be put into CNF [3], we just need to consider the translation of negations and disjunctions. The composition of the translation of disjunctions is straightforward. In the following section, we will describe an algorithm for translating implications (e.g., axiom (4)), which includes translating negations. It actually shows that we can translate any true statement (theory) from one ontology to another ontology, after we transform the theory to CNF form.

[3] Conjunctive normal form (CNF): a conjunction of clauses, where each clause is a disjunction of literals. Literals can have negations and variables. For example, the implication $P_{s1} \to P_{s2}$ can be transformed to its equivalent $\neg P_{s1} \vee P_{s2}$.

4 Axiom Derivation

4.1 Conditional Facts and ICF Axioms

To explain our approach to theory (axiom) translation, we first show how to translate *conditional facts* using OntoEngine. A conditional fact is a formula of the form:

$$P_1 \wedge \cdots \wedge P_i \cdots \wedge P_n \rightarrow Q_1 \wedge \cdots \wedge Q_j \wedge \cdots \wedge Q_m$$

where all $P_i (1 \leq i \leq n)$ and $Q_j (1 \leq j \leq m)$ are ground atomic formulas (facts). The axioms, such as axioms 4 and horn-like bridging axioms, can be put in this form which we call *ICF (Implicative Conjunction Form)*, but of course axioms have quantified variables:

$$\forall v_1 \ldots \exists v_k \ldots v_l,\ P_1 \wedge \cdots \wedge P_i \cdots \wedge P_n \rightarrow Q_1 \wedge \cdots \wedge Q_j \wedge \cdots \wedge Q_m$$

where the v are quantified and typed variables, some universal (e.g., v_1) and some existential.

It is unusual but not unheard of for people to need to express that some facts are true only if some other facts are also true:

$$\begin{aligned} &precedes(deathof(Roosevelt),\ endof(WW2)) \rightarrow \\ &\qquad president(Truman,\ endof(WW2)) \end{aligned}$$

"If Roosevelt died before the end of World War 2, then Truman was president at the end of World War 2."

4.2 Conditional Fact Translation

Conditional fact translation is the translation of a conditional fact from the source ontology to the target ontology. This is a typical example for which we need to consider asymmetry of translation since the translation of implications actually includes the translation of negations and disjunctions. For example, suppose we have a simple conditional fact in $Gonto_1$:

$$@\mathsf{Go1} : \mathsf{husband}(A, B) \rightarrow @\mathsf{Go1} : \mathsf{married}(A, B)$$

where Go1 is the prefix of $Gonto_1$ (adopting some syntax from Web-PDDL) and A and B are a male and a female. We want to translate this conditional fact to $Gonto_2$ which has prefix Go2.

Considering the asymmetry of translation, the antecedent @Go1:husband(A,B) can be translated to @Go2:spouse(A,B) by the query translation with backward chaining, and the conclusion @Go1:married(A,B) can be translated to @Go2: in_marriage(A,B) by data translation with forward chaining. We know the result

$$@\mathsf{Go2} : \mathsf{spouse}(A, B) \rightarrow @\mathsf{Go2} : \mathsf{in\_marriage}(A, B)$$

is a true statement in $Gonto_2$.

In summary, the algorithm to do conditional fact translation is:

```
Procedure CFT($C_s$, $M$)
input: conditional fact $C_s$ in the source ontology $O_s$, bridging axioms $M$
    between $O_s$ and the target ontology, $O_t$
output: translated conditional fact $C_t$ in $O_t$
steps:
 1. Let $Ant_s$ be the antecedent of $C_s$ and $Con_s$ be the conclusion of $C_s$ ($C_s$: $Ant_s \rightarrow$
    $Con_s$.)
 2. Get the antecedent of $C_t$, $Ant_t$, by backward chaining of $Ant_s$ with $M$ from $O_s$
    to $O_t$.
 3. Get the conclusion of $C_t$, $Con_t$, by forward chaining of $Con_s$ with $M$ from $O_s$ to
    $O_t$.
 4. Return $C_t$ as $Ant_t \rightarrow Con_t$.
```

It should be obvious that this process yields a valid result, in the sense that the translated fact follows from the original fact and the axioms. If backward chaining from the antecedent fails to find any goals in the target ontology, then the antecedent of the translated conditional fact will be empty, or false, making the translation itself equivalent to true — and hence useless.

4.3 Extending Conditional Facts Translation to Axiom Derivation

The translation of ICF axioms can still be thought of as an inference process called *axiom derivation*, if we can transform the axioms to conditional facts and transform the conditional facts back to axioms. The idea is to substitute Skolem constants for the variables temporarily. (A similar technique was used in [18].) In general, axiom derivation can be broken into three steps:

From ICF axioms to conditional facts: we can use Universal Elimination and Existential Elimination [22] to transform ICF axioms to conditional facts. Suppose that we have an axiom in the source ontology O_s:

$$(\forall x, y)@\mathsf{O\_s} : \mathsf{P}(x, y) \rightarrow$$
$$(\exists z)@\mathsf{O\_s} : \mathsf{Q}(x, z) \wedge @\mathsf{O\_s} : \mathsf{R}(z, y)$$

We can substitute the universal quantified variables with constants (e.g., Atx and Bty) and substitute the existential quantified variables with uniquified Skolem terms (e.g., Skz01):

$$@\mathsf{O\_s} : \mathsf{P}(\mathsf{Atx}, \mathsf{Bty}) \rightarrow$$
$$@\mathsf{O\_s} : \mathsf{Q}(\mathsf{Atx}, \mathsf{Skz01}) \wedge @\mathsf{O\_s} : \mathsf{R}(\mathsf{Skz01}, \mathsf{Bty})$$

Conditional facts translation: suppose that the target ontology is O_t and we already have the merged ontology of O_s and O_t. The conditional fact in O_s can be translated to O_t. By backward chaining from the antecedent and forward chaining from the conclusion, we finally get a conditional fact in O_t:

$$@\mathsf{O\_t} : \mathsf{S}'(\mathsf{Atx}, \mathsf{Ctc}) \wedge @\mathsf{O\_t} : \mathsf{T}'(\mathsf{Ctc}, \mathsf{Bty}) \rightarrow$$
$$@\mathsf{O\_t} : \mathsf{U}(\mathsf{Atx}, \mathsf{Skz01}) \wedge @\mathsf{O\_t} : \mathsf{V}(\mathsf{Skz01}, \mathsf{Skd02})$$
$$\wedge @\mathsf{O\_t} : \mathsf{W}(\mathsf{Bty}, \mathsf{Skz01})$$

where Ctc is a constant and Skd02 is a new generated Skolem term by forward chaining.

From conditional facts to ICF axioms: we can use Universal Generalization [18] and Existential Introduction [22] to transform conditional facts back to ICF axioms.

We can use Universal Generalization to replace all constants which have substituted universal variables with universal variables. For example, Atx, Bty and Ctc can be replaced by x, y and c. We also can use Existential Introduction to replace all Skolem terms with existential variables. Skz01 and Skd02 can be replaced by z and d. Therefore, the generated ICF axiom looks thus:

$$(\forall x, y, c)@\mathsf{O\_t} : \mathsf{S}'(x, c) \wedge @\mathsf{O\_t} : \mathsf{T}'(c, y) \rightarrow$$
$$(\exists z, d)@\mathsf{O\_t} : \mathsf{U}(x, z) \wedge @\mathsf{O\_t} : \mathsf{V}(z, d) \wedge @\mathsf{O\_t} : \mathsf{W}(y, z)$$

4.4 Proof of Axiom Derivation

It's not so obvious that this procedure works, but we can prove that it does.

Theorem: Any axiom developed by the above procedure is a logical consequence of the axioms of the merged ontologies.

Proof: It suffices to show that the negation of the axiom is inconsistent with the merged ontology. As usual, we assume the axiom is in ICF form:

$$Y_1 v_1 \ldots Y_k v_k (R_1 \wedge \cdots \wedge R_i \wedge \cdots \wedge R_n \rightarrow T_1 \wedge \cdots \wedge T_m)$$

where the v_j are quantified variables and Y_j are the quantifiers, some universal and some existential. The axiom this is derived from is

$$X_1 u_1 \ldots X_k u_k (P_1 \wedge P_2 \wedge \ldots \wedge P_n \rightarrow Q_1 \wedge \ldots \wedge Q_m)$$

where the u_i are quantified variables and the X_i are also the quantifiers, some universal and some existential.

What we will actually show is that a weakened version of the negation of the axiom is inconsistent; from which it follows that a strong version is inconsistent as well. Negating the axiom flips the quantifiers, so the existentials become universals and vice versa. We weaken the negation of the axiom by moving all the existentials inward. If $(\exists x \forall y)(\ldots)$ is true, then so is $(\forall y \exists x)(\ldots)$. We can then use the resolution procedure to derive a contradiction. Skolemizing the weakened version turns the (originally) universally quantified variables into Skolem constants and the (originally) existentially quantified variables into free variables. In addition, the conclusion becomes disjunctive, so that we have a list of clauses:

$R'_1 \quad R'_2 \quad \ldots \quad R'_n$
$\neg T'_1 \vee \neg T'_2 \vee \ldots \vee \neg T'_m$

Now we mimic the deductions performed during our axiom derivation procedure, running them in reverse. That is, if we inferred $\theta(R_1 \wedge \ldots \wedge R_k)$ from R_i using a backward-chaining rule $R_1 \wedge \ldots \wedge R_k \rightarrow P_i$, where P_i and R_i unify with substitution θ, we now infer $\theta'(R'_i)$ from $\theta'(R'_1 \wedge \ldots)$. The unification then is possible now, because R'_i is derived from R_i by replacing some Skolem constants with different Skolem constants (possibly losing some of their arguments). The resulting θ' is less restrictive than the original θ, so the process can be repeated until variants of the original P_i are derived. Similarly, we can run the forward chaining from T_1 backward to Q_1, resulting in a smaller clause (with some free variables substituted) $\neg T''_2 \vee \ldots \neg T''_m$. We now repeat the procedure for T''_2, and so forth, until the empty clause is derived. *Q.E.D.*

5 Axiom Derivation for different Time Ontologies

We want to evaluate our axiom derivation algorithm in some real application scenarios. We are especially interested in the translation between complex ontologies which have large sets of axioms. For example, several time ontologies, such as Cyc time [4], SUMO [5] time, and OWL-Time (formerly DAML-Time) [6], describe temporal concepts and their relationships which are represented using large sets of logic axioms (e.g., the OWL-Time ontology has around 180 first order axioms.)

Researchers have manually built some mappings among the concepts of some time ontologies, but have not talked about how to represent the axioms in different time ontologies. In this paper, we use an example to illustrate the automatic translation of axioms from the Cyc time ontology to the OWL-Time ontology.

For example, one of the axioms in the Cyc time ontology can be represented in Web-PDDL as following:

```
 (forall (te1 - TemporalThing da2 - Date)
  (if (dateOfEvent te1 da2)
      (and (startingDate te1 da2)
           (endingDate te1 da2))))
```

It is an axiom to describe the relationship between three properties: dateOfEvent, startingDate and endingDate. This axiom means if some event happens on a specific date, it must begin and end on the same date. The task for OntoEngine is to represent this axiom in the OWL-Time ontology.

We have manually generated the bridging axioms between the Cyc time and OWL-Time ontologies. Here are some examples (cyc and ot are the prefixes for the Cyc time and OWL-Time ontologies.):

```
(forall (e1 - @cyc:Eventuality d2 - @cyc:Date)
        (iff (@cyc:dateofEvent e1 d2)
             (exists (ti - @ot:Interval)
                     (and (@ot:during e1 ti)
                          (@ot:int-during ti d2)))))
```

[4] http://www.cyc.com/cycdoc/vocab/time-vocab.html

[5] http://ontology.teknowledge.com/

[6] http://www.isi.edu/~pan/OWL-Time.html

```
(forall (t1 - @cyc:TimeInterval d2 - @cyc:Date)
        (iff (@cyc:startingDate t1 d2)
             (exists (ti - @ot:Instant)
                     (and (@ot:begins ti t1) (@ot:inside ti d2)))))

(forall (t1 - @cyc:TimeInterval d2 - @cyc:Date)
        (iff (@cyc:endingDate t1 d2)
             (exists (ti - @ot:Instant)
                     (and (@ot:ends ti t1) (@ot:inside ti d2)))))
 ...
```

Where the Cyc time ontology's Date type is treated as a specialization (subtype) of TimeInterval. With those bridging axioms, OntoEngine can do axiom derivation from the Cyc time ontology to the OWL-Time ontology.

First, that axiom in the Cyc time ontology will be transformed to a conditional fact:

```
 (:objects T1 - @cyc:TemporalThing D2 - @cyc:Date)

 (if (@cyc:dateOfEvent T1 D2)
     (and (@cyc:startingDate T1 D2)
          (@cyc:endingDate T1 D2)))
```

Then OntoEngine can do backward chaining from the antecedent, (@cyc:dateOfEvent T1 D2), and forward chaining from both (@cyc:startingDate T1 D2) and (@cyc:ending Date T1 D2). Finally the translated conditional fact in OWL-Time ontology can be transformed back to an axiom in the OWL-Time ontology:

```
 (forall (e - Eventuality d - Interval)
   (if (exists (t - Interval)
               (and (during e t) (int-during t d)))
       (exists (ti1 ti2 - Instant)
               (and (begins ti1 t) (inside ti1 d)
                    (ends ti2 t) (inside ti2 d)))))
```

It is interesting that this generated axiom does not belong to those 180 existing axioms in the OWL-Time ontology. It is a new axiom. However, it does make sense to describe the relationships between an event and a time interval: during the interval the event happens but it may not be through the whole interval. The total 44 axioms in the Cyc time ontology can be automatically translated to 18 axioms in the OWL-Time ontology in 2 seconds using OntoEngine. Not all axioms in the Cyc time ontology can be translated to the OWL-Time ontology because those two ontologies do not have exactly the same concepts.

This real example shows that some time ontologies may have different axioms from other time ontologies, although they have very similar concepts (i.e., types and properties). If we fail to port the axioms from one to another, we lose important aspects of the semantics of the terms involved.

6 Related Work

A lot of other ontology translation work [7, 9] focuses on term rewriting between different ontologies or different ontology languages, but does not use inference. To the best of our knowledge, the only other work on axiom (theory) translation is [6]. This work presents a formalism for knowledge translation based on the theory of contexts [18]. The authors define knowledge translation in terms of truth, and like us they propose using a theorem prover to perform translations. However, the paper doesn't say exactly *how* the theorem-proving process would work. We have shown that a special-purpose inference engine using backward and forward chaining can be used in an efficient mechanism for translating axioms.

7 Conclusion

Complex ontologies require complicated true statements (theory), such as logic axioms (rules). Many relationships among their symbols simply can't be expressed any other way. Based on our formal framework and inference engine for inferential ontology translation, this paper has described and proved the correctness of an *axiom derivation* algorithm for theory translation from one ontology to another.

We have shown that theory translation is necessary in some real application scenarios in which ontologies have large sets of axioms, such as different time ontologies. Although our algorithm is provably correct, practical application requires further work on problems of incompleteness and redundancy. Our algorithm does not by itself guarantee that the axioms we produce are complete, nor does it avoid producing axioms that are already present in the target ontology. Those are problems for future research for theory translation.

Acknowledgements

We would like to thank Haizheng Zhang and Prof. Victor Lesser from the University of Massachusetts, Amherst, for their help with presenting the poster of our short paper [12] in AAMAS2006. We also would like to thank Paea LePendu for his help with the design of the poster.

References

1. DAML+OIL web ontology language.
`http://www.w3.org/TR/daml+oil-reference`.
2. OWL Web Ontology Language.
`http://www.w3.org/TR/owl-ref/`.
3. T. Berners-Lee, J. Hendler, and O. Lassila. The Semantic Web. *Scientific American*, 284(5), May 2001.
4. P. Bouquet, F. Giunchiglia, F. van Harmelen, L. Serafini, and H. Stuckenschmidt. C-OWL: Contextualizing Ontologies. In *International Semantic Web Conference*, pages 164–179, 2003.

5. P. Bouquet, B. Magnini, L. Serafini, and S. Zanobini. A SAT-based algorithm for context matching. In *CONTEXT*, pages 66–79, 2003.
6. S. Buvac and R. Fikes. A Declarative Formalization of Knowledge Translation. In *Proceedings of the ACM CIKM conference*, 1995.
7. H. Chalupsky. Ontomorph: A Translation System for Symbolic Logic. In *Proceedings of the KR conference 2000*, pages 471–482, 2000.
8. Ó. Corcho and A. Gómez-Pérez. A Layered Model for Building Ontology Translation Systems. *Int. J. Semantic Web Inf. Syst.*, 1(2):22–48, 2005.
9. M. Dell'Erba, O. Fodor, F. Ricci, and H. Werthner. Harmonise: A solution for data interoperability. In *I3E 2002*, 2002.
10. A. Doan, J. Madhavan, P. Domingos, and A. Y. Halevy. Learning to Map Between Ontologies on the Semantic Web. In *International World Wide Web Conferences (WWW)*, pages 662–673, 2002.
11. D. Dou and P. LePendu. Ontology-based Integration for Relational Databases. In *SAC '06: Proceedings of the 2006 ACM symposium on Applied computing*, pages 461–466, 2006.
12. D. Dou and D. McDermott. Deriving Axioms across Ontologies. In *Proceedings of International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'06)*, pages 952–954, 2006.
13. D. Dou, D. V. McDermott, and P. Qi. Ontology Translation on the Semantic Web. In *Proceedings of the International Conference on Ontologies, Databases and Application of Semantics (ODBASE)*, pages 952–969, 2003.
14. D. Dou, D. V. McDermott, and P. Qi. Ontology Translation on the Semantic Web. *Journal of Data Semantics*, 2:35–57, 2005.
15. T. Gruber. Ontolingua: A Translation Approach to Providing Portable Ontology Specifications. *Knowledge Acquisition*, 5(2):199–220, 1993.
16. I. Horrocks and P. F. Patel-Schneider. A proposal for an OWL rules language. In *WWW*, pages 723–731, 2004.
17. A. Maedche, B. Motik, N. Silva, and R. Volz. MAFRA - A MApping FRAmework for Distributed Ontologies. pages 235–250, 2002.
18. J. McCarthy and S. Buvac. Formalizing context (expanded notes). In A. Aliseda, R. van Glabbeek, and D. Westerstahl, editors, *Computing Natural Language*. University of Chicago Press, 1997.
19. D. V. McDermott and D. Dou. Representing Disjunction and Quantifiers in RDF. In *International Semantic Web Conference*, pages 250–263, 2002.
20. P. Mitra, G. Wiederhold, and M. Kersten. A graph-oriented model for articulation of ontology interdependencies. In *Proceedings of Conference on Extending Database Technology (EDBT 2000)*, 2000.
21. N. F. Noy. Semantic Integration: A Survey Of Ontology-Based Approaches. *SIGMOD Record*, 33(4):65–70, 2004.
22. S. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice-Hall, Inc, 1995.