Semi-Automatic Verification of Ontology Compatibility Supported by Reasoning
TR 3004

Nadejda Nikitina [1]

Abstract. As the amount of available ontologies and their size grow, ontology reuse gains in importance. However, the online available formalized knowledge in many cases need a revision which can lead to a high manual effort. In this paper, we propose an approach to support the revision of ontologies. We show that our method reduces the manual effort measured in number of decisions that have to be made by an ontology engineer by up to 83%.

1 INTRODUCTION

Constructing ontologies for real-world knowledge-intensive applications is a highly time-consuming task. The revision of ontologies is a typical part of it, since semi-automatic ontology construction, but also an extensive modification of ontologies by a human expert are error-prone tasks. One of the most practical usage scenarios for ontology revision however is ontology reuse which has become an attractive means of reducing the ontology development effort since the introduction of OWL.

There are several reasons for the required manual inspection of the semantic data potentially relevant for the reuse in a new application context:

1. Ontologies often contain fragments irrelevant to the particular application scenario. Therefore, the relevant fragment has to be identified and extracted if necessary.
2. The logical compatibility of the selected relevant fragment and the target ontology has to be verified, since ontologies often model the same domain from different points of view and lexically similar entities can have different logical characteristics.
3. The available relevant knowledge bases tend to overlap. For instance, 25% of the available ontologies within the biomedical and chemical domain have an ontology mapping for more than the half of their concepts [3].

The existing approaches help reducing the manual effort during the ontology reuse with respect to a particular distinguished criterion. A range of ontology partitioning and modularization techniques aim at a semi-automatic extraction of relevant ontology fragments. For instance, Grau et al. [4] and Lutz et al. [6] propose two approaches to a safe, meaning-preserving extraction of ontology fragments based on the notion of conservative extensions. Modular ontology languages aim at reducing the logical incompatibility of formalized knowledge from different sources by restricting the interpretation of external statements within a knowledge base. For instance, $\mathcal{SHOIQP}$ supports context-specific reuse of knowledge from multiple ontologies by the means of contextualized negation and bottom-concept.

[1] KIT, Germany, email: nadejda.nikitina@kit.edu

To the best of our knowledge, we propose the first approach to support the revision of formalized knowledge with respect to any logically verifyable, non-syntactic criteria such as relevance, logical appropriateness or the level of DL expressivity. In our approach, we use reasoning to propagate manual evaluation decisions in order to automatically evaluate axioms and in this way to reduce the number of axioms that have to be evaluated by the human expert. To further minimize the number of required expert decisions, we apply axiom ranking and present the axioms to the evaluator in a particular order. Therefore, we evaluate three different ranking techniques and show that the obvious, local-optimum-oriented solution is not feasible due to its highly problematic execution time and that very good results can also be obtained using an alternative heuristic solution but with an acceptable execution time.

We also propose an ontology reuse method based on ontology revision which allows an ontology engineer to select and reuse any part of the externally specified knowledge in contrast to the approaches [4] and [6]. It includes an integration of the knowledge from different, potentially overlapping sources that allows the ontology engineer to obtain a less complicated input for the revision on the one hand and to safely extend the reused data on the other hand. The output of our approach is standard OWL and allows the ontology engineer to apply any OWL-capable tool to the resulting data.

As we show in our experiments, the proposed approach can reduce the effort spent on the revision of semantic resources, in the special case of ontology reuse by up to 83%. The ranking of axioms yields an average improvement of 25%.

The remainder of this paper is organized as follows: In the next section, we describe some basics that are relevant within this paper. In Section 3, we describe our approach for the support of the ontology revision including different ranking techniques and their evaluation. Section 5 describes the ontology reuse process based on the proposed revision method as well as its evaluation. Finally, Section 6 summarizes the contribution of this paper.

2 PRELIMENARIES

The formalisms of this work are based on the description logic $\mathcal{SHOIQ}$. Please note that our implementation works with the ontology language OWL DL which is based on the description logic $\mathcal{SHOIN}(\mathcal{D})$[5], but for the ease of presentation, we do not consider datatypes in this paper. However, the result of this paper can easily be extended to $\mathcal{SHOIN}(\mathcal{D})$.

The syntax of $\mathcal{SHOIQ}$ is given by a signature S and a set of constructors. S is the disjoint union of a set $\mathbf{C}$ of atomic concepts $(A, B, \ldots)$ representing sets of elements, a set $\mathbf{R}$ of atomic roles

$(r, s \ldots)$ representing binary relations between elements, and a set $\mathbf{I}$ of individuals $(a, b, \ldots)$ representing elements.[2] In the following, we will use the term *ontology entities* or simply *entities* to refer to the elements of the set $\mathbf{C} \cup \mathbf{R}$ of concepts and roles of an ontology.

Constructors provide the means for defining the set $\mathrm{Con}(S)$ of general concepts $(C, D, \ldots)$, the set $\mathrm{Rol}(S)$ of general roles $(R, S, \ldots)$, and the set $\mathrm{Ax}(S)$ of axioms $(\alpha, \beta, \ldots)$. $\mathrm{Rol}(S)$ is built using the inverse roles (R^-) construct. General concepts can be constructed using the following grammar:

$$C \leftarrow A|\neg C|C_1 \sqcap C_2|\exists R.C| \geq nS.C$$

In the latter expression, we use an atomic concept A, an atomic role S, a concept C, a role R, and a positive integer n. We introduce some additional shortcuts: the bottom concept $\perp$ stands for $A \sqcap \neg A$ and the top concept $\top$ is a shortcut for $\neg \perp$, the disjunction of concepts $C_1 \sqcup C_2$ stands for $\neg(\neg C_1 \sqcap \neg C_2)$, and the value restriction $\forall R.C$ stands for $\neg(\exists R.\neg C)$. Another shortcut is $\leq nS.C$ which stands for $\neg(\geq n + 1S.C)$.

The set of terminological axioms (TBox) for a particular signature S consists of concept inclusion $(A \sqsubseteq B)$ and equivalence $(A \equiv B)$ axioms as well as role inclusion $(R \sqsubseteq S)$, role equivalence $(R \equiv S)$, and transitivity $(R^+ \sqsubseteq R)$ axioms.

The logical entailment $\models$ is defined using the set-theoretic semantics for description logics. Given a signature S, an interpretation $\mathcal{I}$ is a pair $\mathcal{I} = (\Delta^{\mathcal{I}}, \cdot^{\mathcal{I}})$, where $\Delta^{\mathcal{I}}$ is a non-empty set, called the domain of the interpretation, and $\cdot^{\mathcal{I}}$ is the interpretation function that assigns to every $A \in \mathbf{C}$ a subset $A^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}}$, to every $R \in \mathbf{R}$ a binary relation $r^{\mathcal{I}} \subseteq \Delta^{\mathcal{I}} \times \Delta^{\mathcal{I}}$, and to every $a \in \mathbf{I}$ an element $a^{\mathcal{I}} \in \Delta^{\mathcal{I}}$. The extension of the interpretation function to complex roles and concepts as well as the satisfaction relation between an interpretation $\mathcal{I}$ and an axiom α can be found in [1]. An interpretation $\mathcal{I}$ is a model of an ontology $\mathcal{O}$, if it satisfies all axioms in $\mathcal{O}$. An ontology $\mathcal{O}$ implies an axiom α, if every model $\mathcal{I}$ of $\mathcal{O}$ satisfies α. An ontology $\mathcal{O}$ implies a set of axioms $\mathcal{T}$, denoted as $\mathcal{O} \models \mathcal{T}$, if it implies each axiom $\alpha \in \mathcal{T}$.

3 REVISION OF KNOWLEDGE BASES

When considering the reuse of knowledge from foreign ontologies, a decision needs to be taken for each of its axioms whether it complies with the requirements underlying $\mathcal{T}$ such as requirements concerning the logical expressiveness within the ontology or the exact meaning of ontology entities. In our approach of ontology revision, the expert reviews the axioms one by one while after each decision taken by the expert some axioms are evaluated automatically and disappear from the revision list. In order to motivate the subsequent formal presentation of the automatic evaluation, we consider the following example. Assume that we have to review the following set of axioms:

(α) Person $\sqsubseteq \neg$Event
(β) Employee $\sqsubseteq \neg$Lecture
(γ) Ordinary $\equiv$ Ordinary_Lecture

given a target ontology with the two following concept hierarchies:

1. Ordinary $\sqsubseteq$ Employee $\sqsubseteq$ Person and
2. Ordinary_Lecture $\sqsubseteq$ Lecture $\sqsubseteq$ Event

In this example, if we approve the axiom α, then, on the one hand, the axiom γ has to be declined, since it would otherwise lead to an inconsistent knowledge base. On the other hand, the axiom β has to be approved, since it already is indirectly contained in the knowledge base due to the subsumption hierarchies and α.

This example illustrates the main idea behind the proposed reasoning-based support for ontology revision which is inspired by the work on debugging ontology mappings by Meilicke et al. [7], in particular the following two assumptions underlying the latter work:

1. If an unevaluated mapping relationship contradicts with the already approved ones, the expert would decline it.
2. If an unevaluated mapping relationship is entailed by the already approved ones, the expert would approve it.

In this way, the the revision decisions of the expert are propagateted to yet unevaluated mappings and the number of mappings to be evaluated by a human expert is reduced. We transfer these two assumptions to the revision of $\mathcal{SHOIQ}$ knowledge bases and define the following succession operator which is based on several concrete notions of contradiction and entailment.

Definition 1 *(Succession operator) Let $\mathcal{O}$ be a set of axioms which has to be reviewed to verify the compatibility with a target ontology $\mathcal{T}$, and let $\mathcal{V}$ be the set {approved, declined, unevaluated} of evaluation values. Let Ω denote the set of all possible axiom sets satisfying the $\mathcal{SHOIQ}$ syntax restrictions. The evaluation state function $f_{\mathcal{T}} : \mathcal{O} \cup \mathcal{T} \rightarrow \mathcal{V}$ with $\forall_{\gamma \in \mathcal{T}} (f_{\mathcal{T}}(\gamma) =$ approved) can be transformed into a more advanced evaluation state function using the succession operator $\Phi : \mathcal{V}^{\mathcal{O}} \rightarrow \mathcal{V}^{\mathcal{O}}$ as follows:*

$$\Phi(f_{\mathcal{T}})(\alpha) = \begin{cases} \textit{approved} & \textit{if } f_{\mathcal{T}}^{-1}(\textit{approved}) \models \alpha \\ \textit{declined} & \textit{if } f_{\mathcal{T}}^{-1}(\textit{approved}) \cup \{\alpha\} \models \beta, \\ & \beta \in f_{\mathcal{T}}^{-1}(\textit{declined}) \cup \{\perp\} \\ & \textit{or } f_{\mathcal{T}}^{-1}(\textit{approved}) \cup \{\alpha\} \notin \Omega \\ \textit{unevaluated} & \textit{otherwise} \end{cases}$$

Notice that due to the monotonicity of reasoning in $\mathcal{SHOIQ}$, Φ preserves the values {*approved, declined*} assigned by $f_{\mathcal{T}}$ to the axioms and only influences the evaluation values of axioms with $f_{\mathcal{T}}(\alpha) =$ *unevaluated*.

The proposed approach is also based on the assumptions that the requirements for the resulting ontology are known to the reviewing ontology engineer and that evaluation decisions cannot be changed during the evaluation. These assumptions are not explicitly modelled in the definition stated above, however they must hold in order for the methods to work correctly.

4 RANKING

The impact of reasoning-based support depends on the order in which axioms are evaluated. For instance, if the ontology engineer has to evaluate the axioms concerning the entity *Decision*

1. Decision $\sqsubseteq$ Distinct-Entity
2. Decision $\sqsubseteq$ Mental-Entity
3. Decision $\sqsubseteq$ Mental-Object

and the evaluated ontology $\mathcal{T}$ already contains the subclass axioms

(α) Mental-Entity $\sqsubseteq$ Distinct-Entity
(β) Mental-Object $\sqsubseteq$ Mental-Entity

[2] In the literature, also the term 'class' is used instead of 'concept' and the terms 'property' and 'relation' are used instead of 'role'.

then, declining axiom 1 or approving axiom 3 would result in an automatic evaluation of axioms 1-3. Approving the axioms in the presented order or declining in the inverse order would however require two additional expert decisions. Obviously, ranking and ordering the axioms can increase the effect of the reasoning-based support. The results of our latter experiments support the intuitive assumption that the calculation of an optimal order by comparing the results of all possible axiom arrangements would not be feasible for an interactive application. In the following, we consider one available ranking technique and develop two new ranking techniques which aim at finding an advantageous order of the axioms.

4.1 MEILICKERANK

Meilicke et al. [7] propose a ranking function for mapping relationships of types $\{$*subconcept, superconcept, disjoint concepts*$\}$ resulting from ontology mapping. This ranking is based on combinatorics and is independent from the current evaluation state. We adapt the this technique for ranking of $\mathcal{SHOIQ}$ axioms in order to evaluate how well the ranking technique is suited for ontology revision. The ranking value for an axiom α containing concepts $C \in \mathbf{C}(\mathcal{O}_1)$ and $D \in \mathbf{C}(\mathcal{O}_2)$ is calculated as follows:

- $\text{sub}(\mathcal{O}_1, C) \cdot (\text{super}(\mathcal{O}_2, D) + \text{dis}(\mathcal{O}_2, D))$, if $\alpha = (C \sqsubseteq D)$
- $\text{super}(\mathcal{O}_1, C) \cdot (\text{sub}(\mathcal{O}_2, D) + \text{dis}(\mathcal{O}_2, D))$, if $\alpha = (C \sqsupseteq D)$
- $\text{sub}(\mathcal{O}_1, C) \cdot (\text{super}(\mathcal{O}_2, D) + \text{dis}(\mathcal{O}_2, D))+$
  $\text{super}(\mathcal{O}_1, C) \cdot (\text{sub}(\mathcal{O}_2, D) + \text{dis}(\mathcal{O}_2, D))$, if $\alpha = (C \equiv D)$

where $\text{sub}(\mathcal{O}, C)$ returns the number of all subconcepts of concept C in $\mathcal{O}$, super $(\mathcal{O}, C)$ returns the number of all superconcepts of C in $\mathcal{O}$, and dis $(\mathcal{O}, C)$ returns the number of all concepts that are disjoint with C . Since the ranking values have to be calculated only once, the ranking method promises to be efficient in terms of time.

4.2 NEXTMAXRANK

As we can see from the example, the number of axioms which can be entailed about the currently considered entity E_u depends on the set of already approved axioms about the entity E_u. Therefore, another possible way to rank the axioms would be to calculate for each unevaluated axiom the number of unevaluated axioms that can be entailed, if the considered axiom is approved. This is exactly how the ranking function NEXTMAXRANK works. It applies reasoning at each evaluation step and presents the axiom with the maximal number of subsequent entailments to the expert. This maximal number of subsequent entailments represents the local maximum of the corresponding evaluation step. Since the ranking values depend on the approved axioms, the result is to a certain extent dependent on the original order of the axioms.Therefore, the overall number of entailments is not neserrarily the global optimum. However, this ranking technique takes possible axiom declines into account by calculating the ranking values after each expert evaluation step.

4.3 MINSETRANK

Another possibility to rank the axioms is to determine a minimal, logically nonredundant subset within the total set of unevaluated axioms which can be used to deduce the remaining unevaluated axioms. If no axioms are declined, un evaluation of the minimal set would suffice. Therefore, the minimal set should be ranked higher than the remainder in order to insure that it will be evaluated first. The ranking technique MINSETRANK is an approximation of this idea. By the means of reduction rules shown in Table 1, a set of axioms can be reduced to a much smaller set with the same amount of information. We rank an axiom with 1, if there are no reduction rules defined for it or the defined rules are satisfied by the considered knowledge base. Otherwise we rank it with 0.

Axiom type	Reduction Rules
$C_1 \sqsubseteq C_2$	$\mathcal{O} \not\models \{C_1 \sqsubseteq C_3, C_3 \sqsubseteq C_2\}$ for any $C_3 \in \mathbf{C}/\{C \vert \mathcal{O} \models \{C \equiv C_1\} \vee$ $\mathcal{O} \models \{C \equiv C_2\}\}$
$C_1 \sqsubseteq \neg C_2$	$\mathcal{O} \not\models \{C_1 \sqsubseteq C_3, C_2 \sqsubseteq \neg C_3\}$ for any $C_3 \in \mathbf{C}/\{C \vert \mathcal{O} \models \{C \equiv C_1\}$ $\mathcal{O} \not\models \{C_2 \sqsubseteq C_4, C_1 \sqsubseteq \neg C_4\}$ for any $C_4 \in \mathbf{C}/\{C \vert \mathcal{O} \models \{C \equiv C_2\}$
$\exists R_1.\top \sqsubseteq C_1$	$\mathcal{O} \not\models \{C_1 \sqsupseteq C_2, \exists R_1.\top \sqsubseteq C_2)$ for any $C_2 \in \mathbf{C}/\{C \vert \mathcal{O} \models \{C \equiv C_1\}$ $\mathcal{O} \not\models \{R_2 \equiv R_1^-, R_2 = R^*, \top \sqsubseteq \forall R_2.C_1\}$ for any $R_2 \in \mathbf{R}/\{R \vert \mathcal{O} \models \{R \equiv R^*\}$
$\top \sqsubseteq \forall R_1.C_1$	$\mathcal{O} \not\models \{C_1 \sqsupseteq C_2, \top \sqsubseteq \forall R_1.C_2)$ for any $C_2 \in \mathbf{C}/\{C \vert \mathcal{O} \models \{C \equiv C_1\}$ $\mathcal{O} \not\models \{R_2 \equiv R_1^-, R_2 = R^*, \exists R_2.\top \sqsubseteq C_1\}$ for any $R_2 \in \mathbf{R}/\{R \vert \mathcal{O} \models \{R \equiv R^*\}$
$R_1 \sqsubseteq R_2$	$\mathcal{O} \not\models \{R_1 \sqsubseteq R_3, R_3 \sqsubseteq R_2\}$ for any $R_3 \in \mathbf{R}/\{R \vert \mathcal{O} \models \{R \equiv R_1\} \vee$ $\mathcal{O} \models \{R \equiv R_2\}\}$

Table 1. MINSETRANK reduction rules for different axiom types

In some cases, there are several alternative ways to express a piece of information and both alternatives would result in an equal number of axioms in the minimal set. In this case, we choose one of the alternatives to be an element of the minimal set based on the original order of the axioms. For instance, one of the two roles inverse to each other is chosen as a target for axioms (denoted as R^* within the reduction rules) if it was the first one to be mentioned in an axiom. The redundant axioms of the second role will then be ranked with 0.

In order to verify the satisfaction of the reduction rules in an inconsistent axiom set, we introduce a simple paraconsistent reasoning method...

4.4 Evaluation

We compare the described ranking methods on several axiom sets in order to estimate their strengths and weaknesses. On the one hand, we investigate, to what extent the manual effort of ontology revision can be reduced in terms of required expert decisions. On the other hand, we are interested in execution time required for the evaluation of the different axiom sets. The relative effort reduction shown in Table 3 is calculated as

$$\varrho(\mathcal{O}) = 100\% \cdot \frac{\#(\mathcal{O}) - \varepsilon(\mathcal{O})}{\#(\mathcal{O})}$$

where $\varepsilon(\mathcal{O})$ is the number of axioms that have to be evaluated by a human expert and $\#(\mathcal{O})$ the number of axioms considered in the revision. In order to measure $\varepsilon(\mathcal{O})$, we run a simulation of the evaluation where a virtual expert evaluates axioms according to a particular simulation setting. Since the discussed ranking techniques are to different extents optimized for the approval of the reviewed data, they can react differently on a decreasing level of data quality. Each simulation setting has a particular defined proportion of declined axioms which reflects a situation with a particular niveau of data quality. In

Ranking	Simulation Mode	Test Ontology A	Test Ontology B	Pcs	MyReview	Cmt	OpenConf	Sofsem
Axiom set size		43	18	187	1126	694	452	424
RANDOM	approve all	22/**48%**	12/**33%**	129/**31%**	820/**27%**	399/**43%**	352/**22%**	270/**36%**
	decline every 10th	22/48%	12/33%	140/25%	873/22%	445/36%	367/19%	293/31%
	decline every 2nd	31/27%	16/11%	167/10%	1025/9%	584/16%	421/7%	382/10%
	best time	00:00:01	00:00:01	00:00:03	00:03:21	00:00:21	00:00:31	00:00:12
NEXTMAX	approve all	9/**79%**	6/**66%**	118/**36%**	361/**68%**	192/**72%**	228/**49%**	218/**48%**
	decline every 10th	9/79%	6/66%	127/32%	422/62%	222/68%	247/45%	242/42%
	decline every 2nd	24/44%	14/22%	171/8%	842/25%	441/36%	360/20%	349/17%
	best time	00:00:06	00:00:01	00:03:24	09:29:23	01:07:19	00:57:34	01:00:44
MEILICKE	approve all	18/**58%**	11/**38%**	128/**31%**	789/**30%**	350/**50%**	336/**26%**	263/**38%**
	decline every 10th	18/58%	11/38%	134/28%	839/25%	398/43%	351/22%	278/34%
	decline every 2nd	26/39%	14/22%	165/11%	996/12%	527/24%	425/6%	370/13%
	best time	00:00:00	00:00:01	00:00:03	00:02:10	00:00:45	00:02:37	00:00:37
MINSET	approve all	9/**79%**	6/**66%**	125/**33%**	405/**64%**	231/**67%**	252/**44%**	238/**44%**
	decline every 10th	9/79%	6/66%	136/27%	447/60%	269/61%	259/43%	254/40%
	decline every 2nd	24/44%	14/22%	175/6%	822/27%	517/26%	382/15%	337/21%
	best time	00:00:00	00:00:00	00:00:03	00:01:53	00:00:57	00:01:32	00:00:35

Table 2. Simulation results for different ranking techniques containing $\varrho(\mathcal{T})$ and $\varepsilon(\mathcal{T})$ values for each axiom set $\mathcal{T}$ and each experimental setting.

this evaluation, we have used three different simulation setting: *accept all*, *decline every 10th* and *decline every 2nd*.

We run the simulation for different axiom sets obtained from the ontologies of the OntoFarm dataset [8]. In order to better understand the ranking effectiveness, we additionally use a test ontology with an explicitly known structure, which is small enough to be read at once by a human. The test ontology A is shown in Fig. 1. The total number of logical axioms extracted from the test ontology amounts to 43 axioms. To demonstrate how the redundancy depends on the hierarchy depth, we also construct a test ontology B that does not contain the concepts A_6, A_5 and A_4. In the domain axiom,A_6 is replaced by A_3. The number of extracted logical axioms decreased to 18 which is less than a half of the previous amount. Indeed, alone the number of possible subclass axioms grows factorially with the hierarchy depth. We conclude that the hierarchy depth is a very important factor for the redundancy within a knowledge base.

$$A_1 \sqsupseteq A_2 \sqsupseteq A_3 \sqsupseteq A_4 \sqsupseteq A_5 \sqsupseteq A_6$$
$$A_7 \sqsupseteq A_8 \sqsupseteq A_9$$
$$A_1 \sqsubseteq \neg A_7$$
$$\exists R_1.\top \sqsubseteq A_6$$

Figure 1. The structure of the test ontology A

The results of the simulation on a MacBook Pro with 2.53 GHz Intel Core 2 Duo Processor and 4 GB memory are presented in Table 2. Before discussion the ranking methods, we point out two interesting findings that concern all ranking methods:

1. Even though the experimental setting RANDOM does not require the additional execution time for ranking and sorting of axioms since they are evaluated in a random order, MINSETRANK and MEILICKERANK outperform it with three axiom sets out of seven. This can be explained by the significant reduction of the number of required evaluation steps which in turns leads to an execution time reduction that is higher than the time required for axiom ranking. In case of an evaluation by a human expert, we expect the improvement of evaluation time caused by ranking to be much more prominent, since each expert evaluation step will take more time than the simulated expert decision and therefore the number of expert evaluation steps will be leveraged significantly higher than the time required for ranking.
2. While the results in the simulation mode *accept all* show more or less stable performance differences between the different ranking methods, the other two modes, and especially *decline every 2nd*, yield less stable experiment results. This can be explaned by the fact that the decline of different axioms also has differently strong consequences for the effort reduction and since the ranking methods do not explicitly take the potential of a decline into account, a higher number of declines leads to more random results.

As we can see, NEXTMAXRANK achieves the best average performance. However, the time results of NEXTMAXRANK are unacceptable with the execution time of 9.5 hours on the axiom set MyReview. The time results of MEILICKERANK are significantly better with 2.18 hours for the same axiom set, but its effectiveness is on average 20% lower than that of NEXTMAXRANK. MINSETRANK achieves almost as high effectiveness as NEXTMAXRANK with on average only 3% lower performance and at the same time very good results in terms of execution time. In fact, the execution time of MINSETRANK is only slightly higher than that of the RANDOM experimental setting, but for the reason discussed above, in a real revision setting MINSETRANK is very likely to yield the best time results. Because of its good performance and execution time, we use MINSETRANK in the latter experiments within the context of ontology reuse.

5 REVISION-BASED ONTOLOGY REUSE

In principle, ontology revision is an important task within most ontology enginnering processes such as manual ontology construction or ontology learning. In this section, we demonstrate how the proposed ontology revision support can be embeded in an ontology reuse scenario, where the knowledge from several potentially overlapping sources needs to be reused in a particular application context that is partially specified by a target ontology $\mathcal{T}$.

The reuse process is depicted in Algorithm1. It consists of an automatic information integration phase and an interactive revision phase. The first phase includes an Ontology Matching task in order to establish the relationships between the entities $E_\mathcal{T}$ of the target ontology and the entities $E_{\mathcal{F}_i}$ of the external knowledge bases $\mathcal{F}_i$. Note that when the externaly formalized knowledge comes from different sources, several correspondence relationships for the same entity as well as correspondence relationships between the foreign ontologies are possible. In this section, we denote the set of logical equivalence axioms $E_\mathcal{T} \equiv E_{\mathcal{F}_i}$ derived from these correspondence relationships as $\mathcal{C}_{\mathcal{F}_i\mathcal{T}}$.

Algorithm 1 Ontology reuse supported by reasoning-based revision

Require: target ontology $\mathcal{T}$, sets of axioms $\mathcal{F}_i$
$\mathcal{C}_{\mathcal{F}_i\mathcal{T}}$ =MATCH($\mathcal{T}, \mathcal{F}_i$) {$\mathcal{C}_{\mathcal{F}_i\mathcal{T}}$: equivalence axioms derived from Ontology Matching}
$U \leftarrow$ TRANSLATE$^\mathcal{T}(\mathcal{F}_i, \mathcal{C}_{\mathcal{F}_i\mathcal{T}})$ {U: set of unevaluated axioms}
$f \leftarrow$ INIT($\mathcal{T}, U$) {f: function reflecting the state of the revision}
$f \leftarrow \Phi(f)$
$U \leftarrow f^{-1}(unevaluated)$
$U \leftarrow$ MINSETRANKSORT(U, f)
while $U \neq \emptyset$ **do**
  $\alpha \leftarrow$ GETFIRST(U)
  $f \leftarrow$ EXPERTEVALUATE(α)
  $U \leftarrow U/\{\alpha\}$
  $f \leftarrow \Phi(f)$
  $U \leftarrow f^{-1}(unevaluated)$
end while
return f

Using $\mathcal{C}_{\mathcal{F}_i\mathcal{T}}$, the axioms of the knowledge bases are "translated" into the vocabulary of the target ontology in order to obtain a less complicated input for the revision and allow the ontology engineer a safe partial reuse and modification of the reused knowledge. This processing step will be discussed in more detail in the next subsection.

Since $\mathcal{T}$ is assumed to already comply with the application requirements, its axioms can be seen as approved and are assigned the corresponding value when the evaluation state function is initialized. The initialized function is used to propagate the evaluation values based on the definition of Φ before the axioms are ranked and sorted using MINSETRANK. Already at this stage and before the expert evaluates any axioms, many of the unevaluated axioms entailed by and contradicting $\mathcal{T}$ can be evaluated automatically.

During the second phase, the expert reviews the sorted axioms in U one by one while after each evaluation decision some axioms are evaluated automatically by Φ and disappear from the revision list.

5.1 Knowledge Base Translation

As already mentioned in the introduction, externally specified knowledge bases often contain statements that are not compatible with the requirements unterlying the new application context or do not contain some statements that are required within this context. If, for instance, the role *partOf* is specified as irreflexive, but it has to be reflexive in the new appication scenario due to a slightly different meaning, there is a need for a safe way to adapt the specification of this role. Since the IRIs of ontology entities are supposed to be globally unique, it would be safe to create a new entity with a new IRI or to use the IRI of an entity declared within the target ontology. The latter of the two alternatives would also allow us to reduce the sets of equivalent entities declared in $\mathcal{C}_{\mathcal{F}_i\mathcal{T}}$ to a single entity per set with a single IRI that is probably already familiar to the ontology engineer responsible for the revision.

To transfer the original specification into the target entity, we partially "translate" the language of a foreign ontology into the language of $\mathcal{T}$ by the means of the following translation operator:

Definition 2 *(Translation operator) A translation of an axiom $\alpha \in \mathcal{F}$ into the language of the ontology $\mathcal{T}$ w.r.t. the set of equivalence relationships $\mathcal{C}_{\mathcal{F}\mathcal{T}}$ is defined as*

$$T^\mathcal{T}(\alpha, \mathcal{C}_{\mathcal{F}\mathcal{T}}) = \alpha\{E_\mathcal{F} \mapsto E_\mathcal{T} | (E_\mathcal{F} \equiv E_\mathcal{T}) \in \mathcal{C}_{\mathcal{F}\mathcal{T}}\}$$

A translation of a set of axioms $\mathcal{F}$ is defined as

$$T^\mathcal{T}(\mathcal{F}, \mathcal{C}_{\mathcal{F}\mathcal{T}}) = \{T^\mathcal{T}(\alpha, \mathcal{C}_{\mathcal{F}\mathcal{T}}) | \alpha \in \mathcal{F}\}$$

Consequently, a translation of $\mathcal{F}$ is a set of axioms resulting from the replacement of the foreign ontology symbol $E_\mathcal{F}$ within $\mathcal{F}$ by the symbol $E_\mathcal{T}$ of the target ontology, if $\mathcal{C}_{\mathcal{F}\mathcal{T}}$ contains $E_\mathcal{F} \equiv E_\mathcal{T}$. Note that axioms obtained from a translation may reference entities from both, the foreign and the target ontologies, since entities $E_\mathcal{F}$ not connected by any equivalence axioms to the entities of $\mathcal{T}$ were not replaced.

We claim that the following statements expressing three different requirements hold for the translation operator:

Theorem 1 *(Requirements fulfillment by the translation operator) Let Ω denote the set of all possible axiom sets satisfying the $\mathcal{SHOIQ}$ syntax restrictions. Let $\mathcal{T}$, $\mathcal{Y}$ and $\mathcal{F}$ be sets of $\mathcal{SHOIQ}$ axioms.*

1. *$(\mathcal{T} \cup T^\mathcal{T}(\mathcal{F}, \mathcal{C}_{\mathcal{F}\mathcal{T}}) \subset \Omega$, if $(\mathcal{T} \cup \mathcal{F} \cup \mathcal{C}_{\mathcal{F}\mathcal{T}}) \subset \Omega$ (Preservation of $\mathcal{SHOIQ}$ syntax restrictions)*
2. *$(\mathcal{T} \cup \mathcal{F} \cup \mathcal{C}_{\mathcal{F}\mathcal{T}}) \models \mathcal{Y}$, iff $\mathcal{T} \cup T^\mathcal{T}(\mathcal{F}, \mathcal{C}_{\mathcal{F}\mathcal{T}}) \models T^\mathcal{T}(\mathcal{Y}, \mathcal{C}_{\mathcal{F}\mathcal{T}})$ (Entailment)*
3. *$\mathcal{T} \cup \mathcal{F} \cup \mathcal{C}_{\mathcal{F}\mathcal{T}}$ is consistent, iff $\mathcal{T} \cup T^\mathcal{T}(\mathcal{F}, \mathcal{C}_{\mathcal{F}\mathcal{T}})$ is consistent (Consistency)*

The stated requirements on the one hand prevent the generation of facts which were not a part of the original knowledge base, and, on the other hand, preserve all facts of the original knowledge base. They also preserve the consistency and prevent a violation of the $\mathcal{SHOIQ}$ syntax restrictions within the knowledge base. Since the translation operator only merges equivalent entities, the proof for these statements is trivial.

5.2 Ontology Reuse Evaluation

We evaluate the proposed methodology in the scenario where search engine results are considered for reuse in a particular context represented by the target ontologies. As target ontologies, we consider the ontologies from the OntoFarm dataset [8], shown in the column headers of Table 3. We perform the search and revision for each of them separately. We use the ontology search engine Watson [2] to obtain potential axioms for the reuse. Watson matches entity names with keywords provided as input based on simple string-similarity and retrieves axioms referencing these entities. We use the name of each entity referenced in the target ontologies as a keyword. On average, we obtained 3.060 axioms for a target ontology.

The obtained axiom sets were processed as described in Algorithm 1 using the simulation setting *approve all* introduced in Section 4.4.

Results		Cmt	MyReview	OpenConf	Pcs	Sofsem
Considered ontologies		3303	355	1596	3251	2674
Considered axioms		3426	3168	2376	3538	2791
RANDOM	Total manual decisions	1622	1658	873	1653	1399
	Total automated decisions	1804	1510	1503	1885	1392
	Implications	1205	1115	926	1376	1162
	Declines	599	395	577	509	230
	Reduction of effort	**53%**	**48%**	**63%**	**53%**	**50%**
MINSETRANK	Total manual decisions	711	773	612	704	467
	Total automated decisions	2715	2395	1764	2834	2324
	Implications	2116	2000	1187	2325	2094
	Declines	599	395	577	509	230
	Reduction of effort	**79%**	**76%**	**74%**	**80%**	**83%**

Table 3. Experimental results for supporting the revision of search results.

We also explicitly measure the effect of ranking within this scenario and therefore repeat the same procedure but without ranking and sorting.

Table 3 shows the results of each target ontology. We were able to reduce the effort of the evaluation by up to 83%. As we see, the ranking of axioms results in an additional average improvement of 25%. We also see that the number of automatically declined axioms is surprisingly high (15%). We investigated the source of this high level of incompatibility and found out that the majority of the conflicts were caused by a difference in the type of entities with similar names. For instance, if in the one axiom, *title* is modeled as a concept, and in another axiom, it is modeled as a role, combining these two axioms would cause problems. Since the number of entity type conflicts usually increases with the number of different sources, the experienced high number of incompatible axioms seems plausible.

6 SUMMARY

In this paper, we presented a method to support the revision of formalized knowledge in order to minimize the manual effort of ontology reuse. In particular, the following support is provided:

1. Integration of the semantic information in a way that allows a safe customization of the reused data
2. Propagation of expert decisions in order to automatically eliminate redundant as well as logically or syntactically incompatible axioms
3. Optimization of the order, in which axioms are evaluated in order to further minimize the number of required expert decisions

Our experiments demonstrate the potential of our method. We were able to reduce the effort of the revision by up to 83%. The ranking of axioms using the MINSETRANK helped to achieve an additional average improvement of 25%.

ACKNOWLEDGEMENTS

This work is supported by the EU FP6 NeOn project http://www.neon-project.org.

REFERENCES

[1] Franz Baader, Diego Calvanese, Deborah McGuinness, Daniele Nardi, and Peter Patel-Schneider, *The Description Logic Handbook: Theory, Implementation and Applications*, Cambridge University Press, 2003.

[2] Claudio Baldassarre, Laurian Gridinoc, Sofia Angeletou, Marta Sabou, and Enrico Motta, 'Watson: A gateway for next generation semantic web applications', (2008).

[3] Amir Ghazvinian, Natasha Noy, Clement Jonquet, Nigam Shah, and Mark Musen, 'What four million mappings can tell you about two hundred ontologies', in *8th International Semantic Web Conference (ISWC2009)*, (October 2009).

[4] Bernardo Cuenca Grau, Ian Horrocks, Yevgeny Kazakov, and Ulrike Sattler, 'A logical framework for modularity of ontologies', in *In Proc. IJCAI-2007*, pp. 298–304. AAAI, (2007).

[5] I. Horrocks, P.F. Patel-Schneider, and F. van Harmelen, 'From shiq and rdf to owl: The making of a web ontology language', *Journal of Web Semantics*, **1**(1), 7–26, (2003).

[6] Carsten Lutz, Dirk Walther, and Frank Wolter, 'Conservative extensions in expressive description logics', in *IJCAI'07: Proceedings of the 20th international joint conference on Artifical intelligence*, pp. 453–458, San Francisco, CA, USA, (2007). Morgan Kaufmann Publishers Inc.

[7] Christian Meilicke, Heiner Stuckenschmidt, and Andrei Tamilin, 'Supporting manual mapping revision using logical reasoning', in *AAAI*, pp. 1213–1218, (2008).

[8] O. Svab, V. Svatek, P. Berka, D. Rak, and P. Tomasek, 'Ontofarm: Towards an experimental collection of parallel ontologies', in *Proceedings of the 5th International Semantic Web Conference ISWC-05*, (2005). Poster Track.