Wheat and Chaff – Practically Feasible Interactive Ontology Revision

Nadeschda Nikitina[1], Birte Glimm[2], and Sebastian Rudolph[1]

[1] Institute AIFB, Karlsruhe Institute of Technology, DE
[2] Ulm University, Institute of Artificial Intelligence, DE

Abstract. When ontological knowledge is acquired automatically, quality control is essential. We consider the tightest possible approach – an exhaustive manual inspection of the acquired data. By using automated reasoning, we partially automate the process: after each expert decision, axioms that are entailed by the already approved statements are automatically approved, whereas axioms that would lead to an inconsistency are declined. Adequate axiom ranking strategies are essential in this setting to minimize the amount of expert decisions.
In this paper, we present a generalization of the previously proposed ranking techniques which works well for arbitrary validity ratios – the proportion of valid statements within a dataset – whereas the previously described ranking functions were either tailored towards validity ratios of exactly 100% and 0% or were optimizing the worst case. The validity ratio – generally not known *a priori* – is continuously estimated over the course of the inspection process. We further employ partitioning techniques to significantly reduce the computational effort. We provide an implementation supporting all these optimizations as well as featuring a user front-end for successive axiom evaluation, thereby making our proposed strategy applicable to practical scenarios. This is witnessed by our evaluation showing that the novel parameterized ranking function almost achieves the maximum possible automation and that the computation time needed for each reasoning-based, automatic decision is reduced to less than one second on average for our test dataset of over 25,000 statements.

1 Introduction

Many real-world applications in the Semantic Web make use of ontologies in order to enrich the semantics of the data on which the application is based. As a popular example, consider DBpedia, which consists of structured information from Wikipedia. DBpedia uses a background ontology, which defines the meaning of and relationships between terms. For example, if two terms are related via the property *river*, the first one can be inferred to be an instance of the class *Place* and the latter one of the class *River*.

In order to guarantee very high quality standards, the DBpedia background ontology has been created manually. For many applications, however, the time requirements of a completely manual knowledge acquisition process are too high. An additional application of (semi-) automatic knowledge acquisition methods such as ontology learning or matching is, therefore, often considered to be a reasonable way to reduce the expenses of ontology development. The results produced by such automatic methods usually need to be manually inspected either partially, to estimate the overall quality of the resulting data, or to the full extent, to keep the quality of the developed ontology under control.

So far, the knowledge representation community has been focusing on restoring the consistency of ontologies enriched with new axioms as done in various belief revision and repair approaches, see, e.g., [1, 10]. Thereby, new axioms not causing inconsistency are treated as valid facts, which do not require further inspection. Our goal is to support a more restrictive quality control process in which a domain expert inspects a set of candidate axioms and decides for each of them whether it is a desired logical consequence. Based on this decision, we automatically discard or include yet unevaluated axioms depending on their logical relationships with the already evaluated axioms. In the following, we call this interactive process *ontology revision*.

Throughout the paper, we use the following running example, which we write in OWL's functional-style syntax using an imaginary prefix ex to abbreviate IRIs:

Example 1. Let us assume that we have already confirmed that the axioms, which state subclass relations between classes, belong to the desired consequences:

SubClassOf(ex:AluminiumNitrideNanotube ex:AluminiumNitride)
SubClassOf(ex:AluminiumNitride ex:NonOxideCeramics)
SubClassOf(ex:NonOxideCeramics ex:Ceramics)
SubClassOf(ex:Ceramics ex:MaterialByMaterialClass)
SubClassOf(ex:MaterialByMaterialClass ex:Material)
SubClassOf(ex:Material ex:PortionOfMaterial)
SubClassOf(ex:Material ex:TangibleObject)

We further assume that the following axioms, which define several different types for the individual ex:nanotube1, are still to be evaluated:

ClassAssertion(ex:AluminiumNitrideNanotube ex:nanotube1) (1)
ClassAssertion(ex:AluminiumNitride ex:nanotube1) (2)
ClassAssertion(ex:NonOxideCeramics ex:nanotube1) (3)
ClassAssertion(ex:Ceramics ex:nanotube1) (4)
ClassAssertion(ex:MaterialByMaterialClass ex:nanotube1) (5)
ClassAssertion(ex:Material ex:nanotube1) (6)
ClassAssertion(ex:PortionOfMaterial ex:nanotube1) (7)
ClassAssertion(ex:TangibleObject ex:nanotube1) (8)

If Axiom (8) is declined, we can immediately also decline Axioms (1) to (6) assuming OWL or RDFS reasoning since accepting the axioms would implicitly lead to the undesired consequence (8). Note that no automatic decision is possible for Axiom (7) since it is not a consequence of Axiom (8) and the already approved subsumption axioms. Similarly, if Axiom (1) is approved, Axioms (2) to (8) are implicit consequences, which can be approved automatically. If we start, however, with declining Axiom (1), no automatic evaluation can be performed. It can, therefore, be observed that

- a high grade of automation requires a good evaluation order, and
- approval and decline of an axiom has a different impact.

Which axioms have the highest impact on decline or approval and which axioms can be automatically evaluated once a particular decision has been made can be determined with the help of algorithms for automated reasoning, e.g., for RDFS or OWL reasoning. One of the difficulties is, however, that it is not known in advance, which of the two decisions the user makes. In our previous work [8], we tackle this problem by showing that, if the quality of the acquired axioms is known, a prediction about the decision of the user can be made: if the quality is high, the user is likely to approve an axiom. Hence, axioms that have a high impact on approval should be evaluated with higher priority. For low quality data, the situation is reversed. We measure the quality by means of the *validity ratio*, i.e., the percentage of accepted axioms, and show in [8] that, depending on the validity ratio of a dataset, different impact measures used for axiom ranking are beneficial. In this paper, we extend the previous results in several directions:

- First, we generalize the ranking functions proposed in [8], which are tailored towards validity ratios of 100% and 0% by parametrizing the ranking function by an estimated validity ratio. In our evaluation, we show that the revision based on the novel ranking function almost achieves the maximum possible automation. The gain is particularly important for datasets with a validity ratio close to 50%, since the currently existing ranking function for those datasets only optimizes the worst case and does not fully exploit the potential of automation.
- Second, since the expected validity ratio is not necessarily known in advance, we suggest a ranking function where the validity ratio is learned on-the-fly during the revision. We show that, even for small datasets (50-100 axioms), it is worthwhile to rank axioms based on this learned validity ratio instead of evaluating them in a random order. Furthermore, we show that, in case of larger datasets (e.g., 5,000 axioms and more) with an unknown validity ratio, learning the validity ratio is particularly effective due to the law of large numbers, thereby making the assumption of a known or expected validity ratio unnecessary. For such datasets, our experiments show that the proportion of automatically evaluated axioms when learning the validity ratio is nearly the same (difference of 0.3%) as in case where the validity ratio is known in advance.

Even for not very expressive knowledge representation formalisms, reasoning is an expensive task and, in an interactive setting as described above, a crucial challenge is to minimize the number of expensive reasoning tasks while maximizing the number of automated decisions. In our previous work [8], we have developed *decision spaces* – data structures, which exploit the characteristics of the logical entailment relation between axioms to maximize the amount of information gained by reasoning. Decision spaces further allow for reading off the impact that an axiom will have in case of an approval or decline. In this paper, we extend the latter work by combining decision spaces with a partitioning technique in order to further improve the efficiency of the revision process. It is interesting to observe that partitioning intensifies the effectiveness of decision spaces, since it increases the relative density of dependencies between axioms considered together during the revision.

We further present *revision helper*: an interactive application supporting ontology revision. We evaluate the proposed techniques and demonstrate that even for expressive OWL reasoning, an interactive revision process is feasible with on average 0.84 seconds

(7.4 reasoning calls) per expert decision, where the automatic evaluation significantly reduces the number of expert decisions.[3]

The remainder of this paper is organized as follows. Next, we describe relevant preliminaries. Section 3 describes the proposed new ranking function and how the validity ratio can be learned during the revision. Section 4 introduces partitioning as a way of optimizing the efficiency of the revision process. We then evaluate the approach in Section 5 and present the user front-end of *revision helper* in Section 6. In Section 7, we discuss the existing related approaches and then conclude in Section 8.

2 Preliminaries

In this section, we introduce the basic notions that are relevant to the revision of an ontology. The ontologies that are to be revised can be written in standard semantic web languages such as RDFS or OWL. We focus, however, on OWL 2 DL ontologies.

The revision of an ontology $\mathcal{O}$ aims at a separation of its axioms (i.e., logical statements) into two disjoint sets: the set of intended consequences $\mathcal{O}^{\models}$ and the set of unintended consequences $\mathcal{O}^{\not\models}$. This motivates the following definitions.

Definition 1 (Revision State). *A* revision state *is defined as a tuple* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ *of ontologies with* $\mathcal{O}^{\models} \subseteq \mathcal{O}, \mathcal{O}^{\not\models} \subseteq \mathcal{O}$*, and* $\mathcal{O}^{\models} \cap \mathcal{O}^{\not\models} = \emptyset$*. Given two revision states* $(\mathcal{O}, \mathcal{O}_1^{\models}, \mathcal{O}_1^{\not\models})$ *and* $(\mathcal{O}, \mathcal{O}_2^{\models}, \mathcal{O}_2^{\not\models})$*, we call* $(\mathcal{O}, \mathcal{O}_2^{\models}, \mathcal{O}_2^{\not\models})$ *a refinement of* $(\mathcal{O}, \mathcal{O}_1^{\models}, \mathcal{O}_1^{\not\models})$*, if* $\mathcal{O}_1^{\models} \subseteq \mathcal{O}_2^{\models}$ *and* $\mathcal{O}_1^{\not\models} \subseteq \mathcal{O}_2^{\not\models}$*. A revision state is* complete*, if* $\mathcal{O} = \mathcal{O}^{\models} \cup \mathcal{O}^{\not\models}$*, and* incomplete *otherwise. An incomplete revision state* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ *can be refined by evaluating a further axiom* $\alpha \in \mathcal{O} \setminus (\mathcal{O}^{\models} \cup \mathcal{O}^{\not\models})$*, obtaining* $(\mathcal{O}, \mathcal{O}^{\models} \cup \{\alpha\}, \mathcal{O}^{\not\models})$ *or* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models} \cup \{\alpha\})$*. We call the resulting revision state an* elementary refinement *of* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$.

Since we expect that the deductive closure of the intended consequences in $\mathcal{O}^{\models}$ must not contain unintended consequences, we introduce the notion of *consistency* for revision states. If we want to maintain consistency, a single evaluation decision can predetermine the decision for several yet unevaluated axioms. These implicit consequences of a refinement are captured in the *revision closure*.

Definition 2 (Revision State Consistency and Closure). *A (complete or incomplete) revision state* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ *is* consistent *if there is no* $\alpha \in \mathcal{O}^{\not\models}$ *such that* $\mathcal{O}^{\models} \models \alpha$*. The* revision closure $\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ *of* $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ *is* $(\mathcal{O}, \mathcal{O}_c^{\models}, \mathcal{O}_c^{\not\models})$ *with* $\mathcal{O}_c^{\models} := \{\alpha \in \mathcal{O} \mid \mathcal{O}^{\models} \models \alpha\}$ *and* $\mathcal{O}_c^{\not\models} := \{\alpha \in \mathcal{O} \mid \mathcal{O}^{\models} \cup \{\alpha\} \models \beta$ *for some* $\beta \in \mathcal{O}^{\not\models}\}$.

We observe that, for a consistent revision state $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$, the closure $\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ is again consistent and that every further elementary refinement of $\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ is also consistent; furthermore, any consistent and complete refinement of $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ is a refinement of $\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ [8, Lemma 1]. Algorithm 1 employs these properties to implement a general methodology for interactive ontology revision. Instead of initializing $\mathcal{O}_0^{\models}$ and $\mathcal{O}_0^{\not\models}$ with the empty set, one can initialize $\mathcal{O}_0^{\models}$ with already approved axioms, e.g., from a previous revision, and $\mathcal{O}_0^{\not\models}$ with declined axioms from a previous revision and with axioms that express inconsistency and unsatisfiability of classes (or properties), which we assume to be unintended consequences.

[3] Anonymized versions of the used ontologies and the *revision helper* tool can be downloaded from http://people.aifb.kit.edu/nni/or2010/Interactive_Ontology_Revision/.

Algorithm 1: Interactive Ontology Revision

```
   Data: $(\mathcal{O}, \mathcal{O}_0^{\models}, \mathcal{O}_0^{\not\models})$ a consistent revision state
   Result: $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ a complete and consistent revision state
1  $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) \leftarrow \mathrm{clos}(\mathcal{O}, \mathcal{O}_0^{\models}, \mathcal{O}_0^{\not\models})$;
2  while $\mathcal{O}^{\models} \cup \mathcal{O}^{\not\models} \neq \mathcal{O}$ do
3      choose $\alpha \in \mathcal{O} \setminus (\mathcal{O}^{\models} \cup \mathcal{O}^{\not\models})$;
4      if expert confirms $\alpha$ then
5          $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) \leftarrow \mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models} \cup \{\alpha\}, \mathcal{O}^{\not\models})$;
6      else
7          $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) \leftarrow \mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models} \cup \{\alpha\})$ ;
```

In line 3, an axiom is chosen that is evaluated next. As motivated in the introduction, a random decision can have a detrimental effect on the amount of manual decisions. Ideally, we want to rank the axioms and choose one that allows for a high number of consequential automatic decisions. The notion of *axiom impact* captures how many axioms can be automatically evaluated when the user approves or declines an axiom. Note that after an approval, the closure might extend both $\mathcal{O}^{\models}$ and $\mathcal{O}^{\not\models}$, whereas after a decline only $\mathcal{O}^{\not\models}$ can be extended. We further define $?(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ as the number of yet unevaluated axioms and write $|S|$ to denote the cardinality of a set S:

Definition 3 (Impact). *Let $(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models})$ be a consistent revision state with $\alpha \in \mathcal{O}$ and let $?(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) := |\mathcal{O} \setminus (\mathcal{O}^{\models} \cup \mathcal{O}^{\not\models})|$. For an axiom α, we define*

$$\begin{aligned}
\textit{the}\text{ approval impact } \mathit{impact}^{+}(\alpha) &= ?(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) - ?(\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models} \cup \{\alpha\}, \mathcal{O}^{\not\models})),\\
\textit{the}\text{ decline impact } \mathit{impact}^{-}(\alpha) &= ?(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models}) - ?(\mathrm{clos}(\mathcal{O}, \mathcal{O}^{\models}, \mathcal{O}^{\not\models} \cup \{\alpha\})),\\
\textit{the}\text{ guaranteed impact } \mathit{guaranteed}(\alpha) &= \min(\mathit{impact}^{+}(\alpha), \mathit{impact}^{-}(\alpha)).
\end{aligned}$$

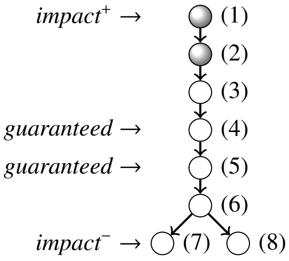
We further separate $\mathit{impact}^{+}(\alpha)$ into the number of automatic approvals, $\mathit{impact}^{+a}(\alpha)$, and the number of automatic declines, $\mathit{impact}^{+d}(\alpha)$:

$$\begin{aligned}
\mathit{impact}^{+a}(\alpha) &= |\{\beta \in \mathcal{O} \mid \mathcal{O}^{\models} \cup \{\alpha\} \models \beta\}|,\\
\mathit{impact}^{+d}(\alpha) &= |\{\beta \in \mathcal{O} \mid \mathcal{O}^{\models} \cup \{\alpha, \beta\} \models \gamma, \gamma \in \mathcal{O}^{\not\models}\}|.
\end{aligned}$$

Note that $\mathrm{impact}^{+}(\alpha) = \mathrm{impact}^{+a}(\alpha) + \mathrm{impact}^{+d}(\alpha)$. The function *impact*$^{+}$ privileges axioms, for which the number of automatically evaluated axioms in case of an accept is high. Going back to our running example, Axiom (1), which yields 7 automatically accepted axioms in case it is accepted, will be ranked highest. The situation is the opposite for *impact*$^{-}$. It privileges axioms, for which the number of automatically evaluated axioms in case of a decline is high (Axioms (7) and (8)). The function *guaranteed* privileges axioms with the highest guaranteed impact, i.e., axioms with the highest number of automatically evaluated axioms in the worst-case (Axioms (4) and (5)). Table 1 lists the values for all ranking functions for the axioms from Example 1.

Which ranking function should be chosen for an ontology revision in order to maximize the amount of automatic decisions depends on the expected validity ratio within the axiom set under revision. For a validity ratio of 100% the function impact^{+} is the most effective, whereas for a validity ratio of 0%, impact^{-} clearly performs best. In

Table 1. Example axiom dependency graph and the corresponding ranking values



Axiom	*impact*$^{+a}$	*impact*$^{+d}$	*impact*$^-$	*guaranteed*
(1)	7	0	0	0
(2)	6	0	1	1
(3)	5	0	2	2
(4)	4	0	3	3
(5)	3	0	4	3
(6)	2	0	5	2
(7)	0	0	6	0
(8)	0	0	6	0

cases when the expected validity ratio is close to 50%, the guaranteed impact can be used to get a reasonable compromise between impact$^+$ and impact$^-$. The ranking functions do, however, not adapt to validity ratios that divert from these extremes. We address this in the next section, by introducing a parametrized ranking function.

Since computing such an impact as well as computing the closure after each evaluation (lines 1, 5, and 7) can be considered very expensive due to the high worst-case complexity of reasoning, we developed *decision spaces* [8] as auxiliary data structures which significantly reduce the cost of computing the closure upon elementary revisions and provide an elegant way of determining high impact axioms. Intuitively, a decision space keeps track of the dependencies between the axioms, i.e., if an axiom β is entailed by the approved axioms together with an unevaluated axiom α, then an "entails" relationship is added linking α to β. Similarly, if adding β to the approved axioms together with an unevaluated axiom α would yield an inconsistency, then a "conflicts" relationship is established between α to β. We show a simplified graph capturing the entails relation for our running example on the left-hand side of Table 1 (the conflicts relation for the example is empty). Note that the entails relation is transitive and reflexive, but for a clearer presentation, we show a transitively reduced version of the graph. From this graph we can see, for example, that if we approve Axiom (5), then we can automatically approve Axioms (6) to (8) as indicated by the (entails) edges in the graph. Thus, decision spaces allow for simply reading-off the consequences of revision state refinements upon an approval or a decline of an axiom, thereby reducing the required reasoning operations. Furthermore, updating a decision space after an approval or a decline can be performed more efficiently compared to a recomputation of all dependencies.

3 Parametrized Ranking

In order to motivate the introduction of the parametrized ranking and to clarify its difference to the three previously proposed ranking functions, we now consider so-called *key axioms* for a path in the *entails*-graph of the decision space: an axiom α on a path p is a key axiom for p if

1. any axiom β on p such that α entails β is correct and
2. any axiom γ on p such that γ entails α is incorrect.

It can be observed that each path is such that there is at least one and there are at most two key axioms (one axiom in case all axioms on the path are correct or incorrect). Intuitively, by making a decision about the key axioms of a path first, we can automatically make a decision for all remaining axioms on the path. While these decisions are made, we might also automatically find conflicts and perform further automatic declines. Conflicts allow, however, for fewer automatic decisions. Hence we focus on the entails paths in the decision space. From this perspective, the behavior of *impact*$^+$ in a tree-shaped structure corresponds to the search for such a key axiom starting from the source axioms with only outgoing entails edges, while the behavior of *impact*$^-$ corresponds to the search from the sink axioms with only incoming entails edges. On the other hand, the behavior of *guaranteed* corresponds to binary search. For instance, if we assume that in our example, Axioms (1) and (2) are incorrect and we choose Axiom (4) among the two highest ranked axioms under the *guaranteed* ranking function, then Axioms (5) to (8) will be automatically evaluated, leaving us with Axioms (1) to (3). This time, Axiom (2) will receive the highest ranking value (1 in contrast to 0 for Axioms (1) and (3)). After another expert decision, Axiom (1) will remain, which is ranked with 0. Therefore, after each expert decision, the remaining axioms are again divided into more or less equally large sets until the set of unevaluated axioms is empty. The improvement achieved by *guaranteed* in comparison to a revision in random order becomes more and more visible with the growing size of the set of unevaluated axioms forming a connected decision space graph, since, in this scenario, the probability of incidentally choosing an axiom with the above specified property becomes lower.

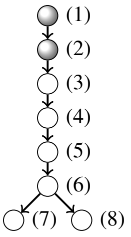
Under the assumption that the dataset in the example has a validity ratio of 75%, the ranking technique *guaranteed* will (theoretically) require 2.8 expert decisions. This is an average for the different possible choices among the highest ranked axioms assuming that these have the same probability of being chosen. In contrast to that, *impact*$^+$ will require 3 expert decisions, while *impact*$^-$ will require even 7 decisions. It is obvious that if the expected validity ratio would have been taken into account, the corresponding ranking strategy would choose Axioms (2) and (3) and require only two expert decisions. In the following, we generalize the ranking techniques *impact*$^+$ and *impact*$^-$, which assume the expected validity ratio to be 100% and 0%, respectively, to a ranking technique, which is parametrized by the actual expected validity ratio. The new ranking technique then chooses axioms based on the expected validity ratio for the dataset.

The goal of the parametrized ranking is to privilege axioms that are most probably key axioms under the assumption that the validity ratio is R. While in Example 1, Axioms (2) and (3) would be the clear choice, in an arbitrary graph, more than two axioms can have such a property. Interestingly, the examination of decision space structures computed within our experiments indicates that the number of possible axioms with such a property is close to two within the connected components of such graphs.

3.1 The Ranking Function *norm*

We now define the ranking function $norm_R$ according to the above set goals. We first normalize the number of automatic approvals and declines to values between 0 and 1. Since in the case of an approval we can possibly accept and decline axioms, we split the approval impact accordingly. We can then normalize the obtained values with respect

Table 2. The values for $\text{norm}_{0.75}$ and the intermediate functions (shown in percentage)



Axiom	$impact_N^{+a}$	$impact_N^{+d}$	$impact_N^{-}$	$norm_{0.75}^{+a}$	$norm_{0.75}^{+d}$	$norm_{0.75}^{-}$	$norm_{0.75}$
(1)	100.0%	0.0%	12.5%	-25.0%	-25.0%	-12.5%	-12.5%
(2)	87.5%	0.0%	25.0%	-12.5%	-25.0%	0.0%	0.0%
(3)	75.0%	0.0%	37.5%	0.0%	-25.0%	-12.5%	0.0%
(4)	62.5%	0.0%	50.0%	-12.5%	-25.0%	-25.0%	-12.5%
(5)	50.0%	0.0%	62.5%	-25.0%	-25.0%	-37.5%	-25.0%
(6)	37.5%	0.0%	75.0%	-37.5%	-25.0%	-50.0%	-25.0%
(7)	12.5%	0.0%	87.5%	-62.5%	-25.0%	-62.5%	-25.0%
(8)	12.5%	0.0%	87.5%	-62.5%	-25.0%	-62.5%	-25.0%

to the expected validity ratio which allows for choosing an axiom that behaves best according to our expectation.

Definition 4. *Let $O^?$ be a connected component of the decision space and R the expected validity ratio. The* normalized impact functions *are:*

$$impact_N^{+a} = \frac{1 + impact^{+a}}{|O^?|}, \quad impact_N^{+d} = \frac{impact^{+d}}{|O^?|}, \quad impact_N^{-} = \frac{1 + impact^{-}}{|O^?|}.$$

The ranking functions $norm_R^{+a}$, $norm_R^{+d}$ and $norm_R^{-}$ are then defined by

$norm_R^{+a} = -|R - impact_N^{+a}|, \quad norm_R^{+d} = -|1 - R - impact_N^{+d}|, \quad norm_R^{-} = -|1 - R - impact_N^{-}|.$

Finally, the ranking function $norm_R$ is:

$$norm_R = max(norm_R^{+a}, norm_R^{+d}, norm_R^{-}).$$

Note that we do not add 1 for $impact_N^{+d}$ since the axiom itself is not declined, i.e., we capture just the “side-effect” of accepting another axiom. Table 2 shows the computation of $norm_{0.75}$ for Example 1. The function $norm_R^{+a}$ captures how the fraction of automatically accepted axioms deviates from the expected overall ratio of wanted consequences, e.g., accepting Axiom (2) or (4) deviates by 12.5%: for the former axiom we have automatically accepted too many axioms, while for the latter we do not yet have accepted enough under the premise that the validity ratio is indeed 75%. Since Example 1 does not allow for automatic declines after an approval, the function $norm_R^{+d}$ shows that for each accept, we still deviate 25% from the expected ratio of *invalid* axioms, which is $1 - R$. The function $norm_R^{-}$ works analogously for declines. Hence, $norm_R$ is defined in a way that it takes the greatest value if the chance that all wanted (unwanted) axioms are accepted (declined) at once becomes maximal.

Note that the expected validity ratio needs to be adjusted after each expert decision, to reflect the expected validity ratio of the remaining axioms. For instance, after Axiom (2) has been declined, $norm_{1.00}$ needs to be applied to rank the remaining axioms. If, however, Axiom (3) has been accepted, $norm_{0.00}$ is required. Note that employing $norm_{0.00}$ for ranking yields the same behavior as $impact^{-}$. On the other hand, $norm_{1.00}$ corresponds to $impact^{+}$ in case no conflicting axioms are involved.

3.2 Learning the Validity Ratio

Users might only have a rough idea of the validity ratio of a dataset in advance of the revision or the validity ratio might not be known at all. Hence, it might be difficult or impossible to decide upfront which R should be used for $norm_R$. To address this problem, we investigate how efficient we can "learn" the validity ratio on-the-fly. In this setting, the user gives an estimate for R (or we use 50% as default) and with each revision step, R is adjusted based on the number of accepted and declined axioms. Thus, the algorithm tunes itself towards an optimal ranking function, which relieves the user from choosing a validity ratio. We call the according ranking function *dynnorm* as it dynamically adapts the estimated validity ratio over the course of the revision.

In our experiments, we show that, already for small datasets, *dynnorm* outperforms random ordering and, in case of sufficiently large datasets, the estimate converges towards the actual validity ratio, thereby making the assumption of a known validity ratio unnecessary.

4 Partitioning

Since reasoning operations are very expensive (the reasoner methods take 99.2% of the computation time in our experiments according to our profiling measurements), we combine the optimization using decision spaces with a straight-forward partitioning approach for ABox axioms (i.e., class and property assertions):

Definition 5. *Let $\mathcal{A}$ be a set of ABox axioms, ind($\mathcal{A}$) the set of individual names used in $\mathcal{A}$, then $\mathcal{A}$ is* connected *if, for all pairs of individuals $a, a' \in ind(\mathcal{A})$, there exists a sequence $a_1, \ldots, a_n$ such that $a = a_1$, $a' = a_n$, and, for all $1 \leq i < n$, there exists a property assertion in A containing a_i and a_{i+1}. A collection of ABoxes $\mathcal{A}_1, \ldots, \mathcal{A}_k$ is a* partitioning of *$\mathcal{A}$ if $\mathcal{A} = \mathcal{A}_1 \cup \ldots \cup \mathcal{A}_k$, $ind(\mathcal{A}_i) \cap ind(\mathcal{A}_j) = \emptyset$ for $1 \leq i < j \leq k$, and each $\mathcal{A}_i$ is connected.*

In the absence of nominals (OWL's oneOf constructor), the above described partitions or clusters of an ABox are indeed independent. Thus, we take each partition separately, join the partition with the TBox/schema axioms and perform the revision. In order to also partition TBox axioms or to properly take axioms with nominals into account, the signature decomposition approach by Konev et al.[6] could be applied. This approach partitions the signature of an ontology (i.e., the set of occurring class, property, and individual names) into subsets that are independent regarding their meaning. The resulting independent subsets of the ontology can then be reviewed independently from each other analogously to the clusters of ABox axioms used in our evaluation. We show in our experiments that:

- In particular in case of large datasets containing several partitions, the additional optimization based on partitioning significantly reduces the computational effort.
- Partitioning intensifies the effectiveness of decision spaces, since the density of entailment and contradiction relations are significantly higher within each partition than the density within a set of independent partitions.

5 Experimental Results

We evaluate our revision support methodology within the project *NanOn*[4] aiming at ontology-supported literature search. During this project, a hand-crafted ontology modeling the scientific domain of nano technology has been developed, including substances, structures, and procedures used in that domain. The ontology, denoted here with $\mathcal{O}$, is specified in the Web Ontology Language OWL 2 DL [11] and comprises 2,289 logical axioms. This ontology is used as the core resource to automatically analyze scientific documents for the occurrence of NanOn classes and properties by the means of lexical patterns. When such classes and properties are found, the document is automatically annotated with those classes and properties to facilitate topic-specific information retrieval on a fine-grained level. In this way, one of the project outputs is a large amount of class and property assertions associated with the *NanOn* ontology. In order to estimate the accuracy of such automatically added annotations, they need to be inspected by human experts, which provides a natural application scenario for our approach. The manual inspection of annotations provided us with sets of valid and invalid annotation assertions (denoted by $\mathcal{A}^+$ and $\mathcal{A}^-$, respectively). To investigate how the quality and the size of each axiom set influences the results, we created several distinct annotation sets with different *validity ratios* $|\mathcal{A}^+|/(|\mathcal{A}^+| + |\mathcal{A}^-|)$. As the annotation tools provided rather reliable data, we manually created additional frequently occurring wrong patterns and applied them for annotating texts to obtain datasets with a lower validity ratio.

For each set, we applied our methodology starting from the revision state $(\mathcal{O} \cup \mathcal{O}^- \cup \mathcal{A}^+ \cup \mathcal{A}^-, \mathcal{O}, \mathcal{O}^-)$ with $\mathcal{O}$ containing the axioms of the NanOn ontology and with $\mathcal{O}^-$ containing axioms expressing inconsistency and class unsatisfiability. We obtained a complete revision state $(\mathcal{O} \cup \mathcal{O}^- \cup \mathcal{A}^+ \cup \mathcal{A}^-, \mathcal{O} \cup \mathcal{A}^+, \mathcal{O}^- \cup \mathcal{A}^-)$ where on-the-fly expert decisions about approval or decline were simulated according to the membership in $\mathcal{A}^+$ or $\mathcal{A}^-$. For computing the entailments, we used the OWL reasoner HermiT.[5]

For each set, our baseline is the reduction of expert decisions when axioms are evaluated in random order, i.e., no ranking is applied and only the revision closure is used to automatically evaluate axioms. For this purpose, we repeat the experiments 10 times and compute the average values of effort reduction. The upper bound for the in principle possible reduction of expert decisions is obtained by applying the *optimal* ranking as suggested by the“impact oracle” for each axiom α that is to be evaluated:

$$\text{KnownImpact}(\alpha) = \begin{cases} \mathit{impact}^+(\alpha) & \text{if } \alpha \in \mathcal{A}^+, \\ \mathit{impact}^-(\alpha) & \text{if } \alpha \in \mathcal{A}^-. \end{cases}$$

5.1 Evaluation of *norm*

To compare the effectiveness of the three previously proposed impact measures and the new impact measure, we created five sets of annotations L_1 to L_5, each comprising 5,000 axioms and validity ratios varying from 10% to 90%.

[4] http://www.aifb.kit.edu/web/NanOn

[5] http://www.hermit-reasoner.com

Table 3. Revision results of *norm* in comparison with other ranking functions for the sets L_1-L_5

	validity ratio	*optimal*	*norm*	*best previous*	*random*
L_1	90%	65.6%	65.4%	(*impact*$^+$) 65.4%	41.7%
L_2	76%	59.8%	55.8%	(*impact*$^+$) 59.9%	35.8%
L_3	50%	47.8%	47.6%	(*guaranteed*) 36.5%	24.4%
L_4	25%	59.9%	59.8%	(*impact*$^-$) 54.9%	37.6%
L_5	10%	63.9%	63.9%	(*impact*$^-$) 63.9%	40.3%

Table 3 shows the results for the different ranking techniques: the column *optimal* shows the upper bound achieved by using the impact oracle, *norm* shows the reduction for our novel ranking parametrized with the actual validity ratio, *best previous* shows the best possible value achievable with the previously introduced ranking functions *impact*$^+$, *guaranteed* and *impact*$^-$, and, finally, the column *random* states the effort reduction already achieved by presenting the axioms in random order.

The results show that *norm* consistently achieves almost the maximum effort reduction with an average difference of 0.1%. The previously introduced ranking functions only work well for the high and low quality datasets, as expected. For the dataset with the validity ratio of 50%, *norm* achieves an additional 11.1% of automation by using the parametrized ranking.

In general, the actual difference in performance achieved by the more precise parametrized ranking increases with the increasing average maximum path length within connected decision space graphs. To see this, consider again the decision space shown in Table 1 and 2. It is clear that the distance between the highest ranked axioms for different ranking functions increases with the increasing height of the presented tree.

5.2 Evaluation of *dynnorm*

In order to evaluate our solution for situations where the validity ratio is unknown or only very rough estimates can be given upfront, we now analyze the effectiveness of the dynamically learning ranking function *dynnorm*. For this, we created the following annotation sets in addition to the datasets $L_1 - L_5$:

- small datasets S_1 to S_5 with the size constantly growing from 29 to 102 axioms and validity ratios varying from 10% to 90%,
- medium-sized datasets M_1 to M_5 with 500 axioms each and validity ratios varying from 10% to 91%.

Table 4 shows the results of the revision: the columns *optimal* and *random* are as described above, the column *norm* shows the results that we would obtain if we were to assume that the validity ratio is known and given as parameter to the *norm* ranking function, the columns *dynnorm*$_{0.50}$, *dynnorm*$_{1.00}$ and *dynnorm*$_{0.00}$ show the results for starting the revision with a validity ratio of 50%, 100%, and 0%, respectively, where over the course of the revision, we update the validity ratio estimate.

We observe that, in case of small datasets (S_i), the deviation from *norm* (on average 5%) as well as the dependency of the results on the initial value of the validity ratio are

Table 4. Revision results for datasets S_1 to S_5, M_1 to M_5, and L_1 to L_5

	validity ratio	*optimal*	*norm*	*dynnorm*$_{0.50}$	*dynnorm*$_{1.00}$	*dynnorm*$_{0.00}$	*random*
S_1	90%	72.4%	72.4%	58.6%	72.4%	65.5%	40.8%
S_2	77%	68.6%	65.7%	57.1%	62.9%	48.6%	38.2%
S_3	48%	65.1%	65.1%	65.1%	60.3%	61.9%	22.0%
S_4	25%	68.3%	68.3%	64.6%	63.4%	67.1%	37.6%
S_5	10%	72.5%	72.5%	71.6%	67.6%	72.5%	29.2%
M_1	91%	66.4%	66.0%	66.2%	66.4%	65.6%	40.8%
M_2	77%	60.0%	60.0%	59.6%	59.8%	59.2%	38.2%
M_3	44%	40.8%	40.6%	40.4%	40.6%	40.4%	22.0%
M_4	25%	60.0%	60.0%	59.6%	59.2%	59.8%	37.6%
M_5	10%	53.2%	53.0%	52.8%	52.8%	53.2%	29.2%
L_1	90%	65.6%	65.4%	65.4%	65.4%	65.3%	41.7%
L_2	76%	59.8%	59.8%	59.8%	59.8%	59.9%	35.8%
L_3	50%	47.8%	47.6%	47.4%	47.2%	47.3%	24.4%
L_4	25%	59.9%	59.8%	59.8%	59.8%	59.8%	37.6%
L_5	10%	63.9%	63.9%	63.9%	63.8%	63.9%	40.3%

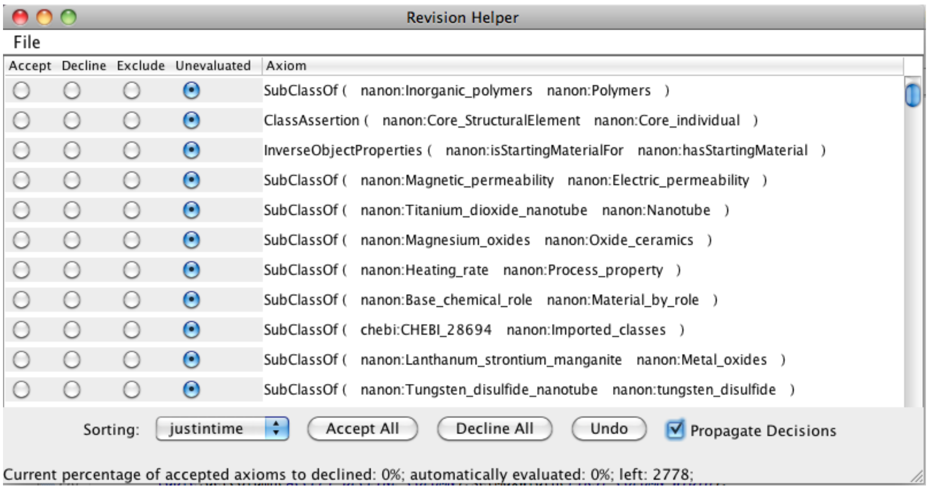
clearly visible. However, the results of *dynnorm* are significantly better (45.0%) than those of a revision in random order. It is also interesting to observe that the average deviation from *norm* decreases with the size of a dataset (6.9%, 10.5%, 2.7%, 3.3%, 1.9% for S_1 to S_5, respectively) and that the probability of a strong deviation is lower for datasets with an extreme validity ratio (close to 100% or 0%).

For medium-sized and large datasets (M_i and L_i), the deviation from *norm* (on average 0.3% for both) as well as the dependency on the initial value of the validity ratio are significantly lower. We conclude that

- ranking based on learning validity ratio is already useful for small datasets (30-100 axioms), and improves significantly with the growing size of the dataset under revision;
- in case of large datasets, the performance difference between the results with a validity ratio known in advance and a learned validity ratio almost disappears, thereby making the assumption of a known average validity ratio not necessary for axiom ranking.

5.3 Computational Effort

During our experiments, we measured the average number of seconds after each expert decision required for the automatic evaluation and ranking as well as the average number of reasoning calls. If we compute the average values for the revision based on *dynnorm* ranking for all 15 datasets, the revision takes on average 0.84 seconds (7.4 reasoning calls) after each expert decision. In the case of small datasets, partitioning yields additionally an improvement by an order of magnitude in terms of reasoning calls. For medium-sized datasets, the first step out of on average 153 evaluation steps took already 101,101 reasoning calls (ca. 3 hours) even when using decision spaces. Without the decision spaces, the required number of reasoning calls would be more than 500,000



Accept	Decline	Exclude	Unevaluated	Axiom
○	○	○	●	SubClassOf (nanon:Inorganic_polymers nanon:Polymers)
○	○	○	●	ClassAssertion (nanon:Core_StructuralElement nanon:Core_individual)
○	○	○	●	InverseObjectProperties (nanon:isStartingMaterialFor nanon:hasStartingMaterial)
○	○	○	●	SubClassOf (nanon:Magnetic_permeability nanon:Electric_permeability)
○	○	○	●	SubClassOf (nanon:Titanium_dioxide_nanotube nanon:Nanotube)
○	○	○	●	SubClassOf (nanon:Magnesium_oxides nanon:Oxide_ceramics)
○	○	○	●	SubClassOf (nanon:Heating_rate nanon:Process_property)
○	○	○	●	SubClassOf (nanon:Base_chemical_role nanon:Material_by_role)
○	○	○	●	SubClassOf (chebi:CHEBI_28694 nanon:Imported_classes)
○	○	○	●	SubClassOf (nanon:Lanthanum_strontium_manganite nanon:Metal_oxides)
○	○	○	●	SubClassOf (nanon:Tungsten_disulfide_nanotube nanon:tungsten_disulfide)

Sorting: justintime

☑ Propagate Decisions

Current percentage of accepted axioms to declined: 0%; automatically evaluated: 0%; left: 2778;

Fig. 1. Revision Helper GUI

judging from the required reasoning calls to build the corresponding decision space in the worst case. For this reason, we did not try to run the experiment for large datasets, which would require more than 50 million reasoning calls without decision spaces. In contrast to that, the average number of required reasoning calls for a complete revision of the sets M_1 to M_5 amounts to 3,380. The revision of datasets L_1 to L_5 required overall on average 16,175 reasoning calls, which corresponds to between 6 and 7 reasoning calls per evaluation decision. We can summarize the evaluation results as follows:

- The proposed reasoning-based support performs well in an interactive revision process with on average 0.84 seconds per expert decision.
- In particular in case of large datasets containing several partitions, the additional optimization based on partitioning significantly reduces the computational effort.
- Decision spaces save in our experiments on average 75% of reasoner calls. As measured in case of small datasets, partitioning further intensifies the effect of decision spaces and we save even 80% of reasoner calls.

6 User Front-End

Figure 1 shows the user front-end of the *revision helper* tool. It allows the user to load the set $\mathcal{O}$ of axioms under revision and save or load an evaluation state for the currently loaded set $\mathcal{O}$. Thereby, the user can interrupt the revision at any time and proceed later on. If partitioning is activated, revision helper shows the partitions one after another and the revision of each partition is independent from the revision of all other partitions.

By default, revision helper initializes the set $\mathcal{O}^{\not\models}$ of undesired statements with the minimal set of statements expressing the inconsistency of the ontology or unsatisfiability of its classes. The set of desired statements $\mathcal{O}^{\models}$ can be initialized by loading an arbitrary ontology. A statement can be evaluated by choosing one of the values *Accept* or *Decline*, and it can be excluded from the revision process by choosing *Exclude*. The latter option should be used if the meaning of a statement is not clear and the user cannot decide whether to accept or to decline it. After the statement has been evaluated, it disappears from the revision list as well as all statements that could be evaluated automatically, unless the checkbox *Propagate Decisions* is deactivated. The ranking strategy used for sorting the statements can be selected or deactivated at any time and is taken into account after the next evaluation decision. At any stage of the revision, it is possible to export the current set $O^{\models}$ of accepted statements as an ontology. For the export, we exclude, however, axioms with which $O^{\models}$ has been initialized at the beginning of the revision.

7 Related Work

We are aware of two approaches for supporting the revision of ontological data based on logical appropriateness: an approach by Meilicke et al.[7] and another one called *ContentMap* by Jiménez-Ruiz et al.[5]. Both approaches are applied in the context of mapping revision. An extension of *ContentMap* called *ContentCVS* [4] supports an integration of changes into an evolving ontology.

In all of these approaches, dependencies between evaluation decisions are determined based on a set of logical criteria each of which is a subset of the criteria that can be derived from the notion of revision state consistency introduced in Def. 1.

In contrast to our approach, the focus of ContentMap and ContentCVS lies within the visualization of consequences and user guidance in case of difficult evaluation decisions. These approaches selectively materialize and visualize the logical consequences caused by the axioms under investigation and support the revision of those consequences. Subsequently, the approved and declined axioms are determined in correspondence with the revision of the consequences. The minimization of the manual and computational effort required for the revision is out of scope. In contrast to our approach, which requires at most a polynomial number of entailment checks, ContentMap and ContentCVS require an exponential number of reasoning operations compared to the size of the ontology under revision. The reason for this is that ContentMap is based on the computation of *justifications*, i.e., sets of axioms causing an entailment, and, in the worst-case, there can be exponentially many justifications for a particular statement.

Similarly to our approach, Meilicke et al. aim at reducing the manual effort of mapping revision. However, their results are difficult to generalize to the revision of ontologies, since the notion of impact is defined based on specific properties of mapping axioms. For every mapping axiom possible between the entities of the two mapped ontologies O_1 and O_2, they define the impact as the corresponding number of possible entailed and contradicting mapping axioms. The assumption is that the set of possible mapping axioms and the set of possible axioms in O_1 and O_2 are mostly disjoint, since axioms in O_1 and O_2 usually refer only to entities from the same ontology, while mapping axioms are assumed to map only entities from different ontologies. In case of ontology revision in general, no such natural distinction criteria for axioms under revision can be defined. Moreover, in contrast to our approach, Meilicke et al. abstract from the interactions between more than one mapping axiom.

Another strand of work is related to the overall motivation of enriching ontologies with additional expert-curated knowledge in a way that minimizes the workload of the human expert: based on the *attribute exploration* algorithm from formal concept analysis (FCA) [3], several works have proposed structured interactive enumeration strategies of inclusion dependencies or axioms of certain fragments of description logics which then are to be evaluated by the expert [9, 2]. While similar in terms of the workflow, the major difference of these approaches to ours is that the axioms are not pre-specified but created on the fly and therefore, the exploration may require (in the worst case exponentially) many human decisions.

8 Summary

In our previous work [8], we established the theoretical ground for partially automated interactive revision of ontologies. In this paper, we present the implementation of the approach including an optimization based on partitioning, which significantly reduces the required computational effort. We further define a generalization of the previously proposed ranking techniques, called *norm*, which is parametrized by the expected validity ratio. The ranking function *norm* works well for any validity ratio, whereas the previous functions were tailored towards validity ratios of 100% or 0%. We define a variant of *norm*, called *dynnorm*, which can be used without knowing the validity ratio beforehand: starting with an initial estimate, e.g., 50%, the estimate is more and more refined over the course of the revision. We evaluate our implementation in a revision of ontology-based annotations of scientific publications comprising over 25,000 statements and show that

- All claims made in [8] hold also in case of large datasets under revision; on average, we were able to reduce the number of required evaluation decisions by 36% when the statements were reviewed in an arbitrary order, and by 55.4% when the ranking techniques suggested in [8] were used.
- The proposed reasoning-based support is feasible for an interactive revision process requiring on average less than one second after each expert decision in our evaluation.
- The parametrized ranking technique proposed in this paper almost achieved the maximum possible automation (59.4% of evaluation decisions) thereby reducing the manual effort of revision by 59.3%. The gain is particularly important for datasets with a validity ratio close to 50%, since for those datasets the potential of automation was not fully exploited by the other ranking techniques. In our experiments, we managed to achieve an additional 11.1% of automation for the dataset with the validity ratio of 50% by using the parametrized ranking.
- In case of large datasets with an unknown validity ratio, learning the validity ratio is particularly effective due to the law of large numbers. In our experiments, the proportion of automatically evaluated statements is nearly the same as in case where the validity ratio is known *a priori* and is used as a fixed parameter of *norm*, thereby making the assumption of known average validity ratio not necessary for axiom ranking.

As part of our future work, we intend to study more general partitioning methods, e.g., [6], to increase the applicability of the partitioning optimization. Another interesting approach in this direction would also be to study the effects of separating the ontology into parts that are not logically independent. In such a case, we might miss automatic decisions, but the potential performance gain, due to the reasoning with smaller subsets of the ontology, might compensate for this drawback.

Acknowledgments

This work is supported by the German Federal Ministry of Education and Research (BMBF) under the SAW-project NanOn.

References

1. Alchourrón, C.E., Gärdenfors, P., Makinson, D.: On the logic of theory change: Partial meet contraction and revision functions. Journal of Symbolic Logic 50, 510–530 (1985)
2. Baader, F., Ganter, B., Sertkaya, B., Sattler, U.: Completing description logic knowledge bases using formal concept analysis. In: Proceedings of the 20th International Joint Conference on Artificial Intelligence (IJCAI 2007). pp. 230–235 (2007)
3. Ganter, B., Wille, R.: Formal Concept Analysis: Mathematical Foundations. Springer-Verlag (1997)
4. Jiménez-Ruiz, E., Grau, B.C., Horrocks, I., Llavori, R.B.: Building ontologies collaboratively using ContentCVS. In: Proceedings of the 22nd International Workshop on Description Logics (DL 2009). CEUR Workshop Proceedings, vol. 477. CEUR-WS.org (2009)
5. Jiménez-Ruiz, E., Grau, B.C., Horrocks, I., Llavori, R.B.: Ontology integration using mappings: Towards getting the right logical consequences. In: Proceedings of the 6th European Semantic Web Conference (ESWC 2009). LNCS, vol. 5554, pp. 173–187. Springer-Verlag (2009)
6. Konev, B., Lutz, C., Ponomaryov, D., Wolter, F.: Decomposing description logic ontologies. In: Proceedings of the 12th International Confonference on Principles of Knowledge Representation and Reasoning (KR 2010) (2010)
7. Meilicke, C., Stuckenschmidt, H., Tamilin, A.: Supporting manual mapping revision using logical reasoning. In: Proceedings of the 23rd Conference on Artificial Intelligence (AAAI 2008). pp. 1213–1218. AAAI Press (2008)
8. Nikitina, N., Rudolph, S., Glimm, B.: Reasoning-supported interactive revision of knowledge bases. In: Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011) (2011)
9. Rudolph, S.: Exploring relational structures via FLE. In: Conceptual Structures at Work: 12th International Conference on Conceptual Structures. LNCS, vol. 3127, pp. 196–212. Springer-Verlag (2004)
10. Schlobach, S., Cornet, R.: Non-standard reasoning services for the debugging of description logic terminologies. In: Proceedings of the 18th International Joint Conference on Artificial Intelligence (IJCAI 2003). pp. 355–362. Morgan Kaufman (2003)
11. W3C OWL Working Group: OWL 2 Web Ontology Language: Document Overview. W3C Recommendation (October 2009), available at `http://www.w3.org/TR/owl2-overview/`