An Investigation of Definability in Ontology Alignment

David Geleta, Terry R. Payne, and Valentina Tamma

Department of Computer Science, University of Liverpool, UK
{D.Geleta|T.R.Payne|V.Tamma}@liverpool.ac.uk

Abstract. The ability to rewrite defined ontological entities into syntactically different, but semantically equivalent forms is an important property of *Definability*. While rewriting has been extensively studied, the practical applicability of currently existing methods is limited, as they are bounded to particular Description Logics (DLs), and they often present only theoretical results. Moreover, these efforts focus on computing single definitions, whereas the ability to find the complete set of alternatives, or even just their signature, can support ontology alignment, and semantic interoperability in general. As the number of possible rewritings is potentially exponential in the size of the ontology, we present a novel approach that provides a comprehensive and efficient way to compute in practice all definition signatures of the feasible (given pre-defined complexity bounds) defined entities described using a DL language for which a particular definability property holds (*Beth definability*). This paper assesses the prevalence, extent and merits of definability over large and diverse corpora, and lays the basis for its use in ontology alignment.

1 Introduction

The ability to rewrite defined ontological entities into syntactically different, but semantically equivalent forms is an important feature of the notion of *Definability*. In particular, *Beth definability* [2, 11] is a well-known property from classical logic, that relates the notion of *implicit definability* to the one of *explicit definability*, by stating that every implicitly defined concept is also explicitly definable, in any definitorially complete DL language [21]. For example, given an ontology $\mathcal{O} = \{\mathsf{C} \equiv \mathsf{A} \sqcup \mathsf{B}, \mathsf{A} \sqsubseteq \neg\mathsf{B}, \mathsf{D} \sqsubseteq \exists \mathsf{r}.\top\}$, where the concept C is defined explicitly, i.e. $\mathsf{C} \equiv \mathsf{A} \sqcup \mathsf{B}$, the concept A is defined implicitly under $\mathcal{O}$ by the set of general concept inclusions $\{\mathsf{C} \equiv \mathsf{A} \sqcup \mathsf{B}, \mathsf{A} \sqsubseteq \neg\mathsf{B}\}$. Thus, A can be explicitly defined by the axiom $\mathsf{A} \equiv \mathsf{C} \sqcap \neg\mathsf{B}$.

Definability in general (and Beth definability in particular) have been utilised within DLs to generate syntactically different, albeit semantically equivalent definitions. Known as *rewriting*, this process is primarily used for: 1) extracting equivalent terminology from a general TBox [1]; and 2) finding equivalent query rewritings in ontology-based data access scenarios [18]. These approaches exploit the fact that any defined concept has one or more possible alternative definitions; however they usually focus on finding a single alternate definition;

whereas several ontology engineering tasks would benefit from the ability to identify a complete set. In particular this paper focusses on *ontology alignment* [7], where several approaches have been proposed that successfully align ontologies [3]. However, Stuckenschmidt et. al. have argued that existing approaches often fail to compute complex correspondences: typically, systems are only able to identify simple equivalence statements between class or relation names, but often fail to identify richer semantic relation between elements of different ontologies [20]. Thus, the ability to rewrite concept definitions can widen the search space for possible correspondences. This is illustrated by the fact that some alignment mechanisms may not find a simple correspondence for some concept C, but may be able to find a complex correspondence, given the its definition $\mathsf{C} \equiv \mathsf{A} \sqcup \mathsf{B}$.

Determining the complete set of possible definitions of defined concepts is a challenging task, as the number of different definitions is potentially exponential in the size of the ontology. This is problematic for large scale ontologies, such as SNOMED CT [6] or FMA [15]. Existing rewriting algorithms are *language dependent*, and thus different approaches to construct rewritings are used for different types of DL expressivity. Furthermore, even if there was an existing approach for a given language, many rewriting systems provide only a theoretical characterization of the rewriting mechanism, therefore making them less usable in practice. Finally, rewriting *requires a seed signature* to be specified in input, i.e. a restricted vocabulary to be used in defining a given concept. The process of identifying all valid seed signatures is inherently complex, as it requires examining each member of the powerset of the ontology signature and verifying whether it actually implicitly defines a particular entity. Therefore, reducing the search space for these problems is highly desirable.

In this paper we present a pragmatic approach to computing the complete set of rewriting signatures for a given ontology. Our approach exploits Beth definability to identify all possible alternative definitions of defined entities. We present the notion of Beth definability in Section 2, and then introduce our novel approach (Section 3) that, *in practice*, can efficiently compute the complete set of definition signatures (DS) of defined entities, for any DL language where the Beth definability property holds. Section 4 presents *definition patterns* (DP), that aid comprehension of definability, and also serve as input for a *heuristic-based rewriting approach*, which produces definition axioms without using reasoning services. Section 5 presents an empirical analysis over a wide range of OWL ontologies, and assesses the prevalence of definability and the behaviour of the proposed algorithms for computing definability. Finally, the paper discusses definability for ontology alignment and presents concluding remarks.

2 Beth Definability in Description Logics

The *vocabulary* of a DL ontology[1] consists of the (disjoint) union of the countably infinite sets of concept names, role names and individual names, where an *entity*

[1] In this paper, we assume familiarity with basic notions of Description Logics [1] and the Web Ontology Language [10] (OWL).

e is either a named concept, role, or individual. A *signature* is an arbitrary set of entities; by $\mathsf{Sig}(\mathsf{C})$ we denote the signature of the a complex concept C, while $\mathsf{Sig}(\mathcal{T})$ denotes the signature of a TBox. In this paper, Σ refers to a *definition signature* (DS), i.e. the set of entities that implicitly define a given concept. A DS is used to characterise *implicitly definable* concepts in terms of their explicit definability, by exploiting the *Beth definability theorem*. The theorem, initially studied for first-order logic [2], states that a logical term is *implicitly definable* with respect to a theory if and only if it is also *explicitly definable*. Given that explicit definability implies implicit definability, the Beth definability property holds for some logic language $\mathcal{L}$ if the converse also holds, i.e. if implicit definability implies explicit definability. Consequently, if a term is implicitly defined then it is always possible to define it explicitly. As there are several variants of Beth definability [21], we focus on *Projective Beth definability* which is a stronger formulation [11] with the ability to specify a set of terms, thus permitting us to restrict the vocabulary that can be used in definitions. Beth definability has also been studied in the context of DLs [21], where it has been used to compute explicit definitions based on implicit definitions. We thus assume a general DL language $\mathcal{L}$ for which the Beth definability property holds. We define an explicit definability concept as:

Definition 1 (Explicitly defined concept). *Let* C *be a concept name, and* $\mathcal{T}$ *a TBox, where* $\mathsf{C} \in \mathsf{Sig}(\mathcal{T})$. C *is explicitly defined under* $\mathcal{T}$*, if and only if there is an axiom* $\alpha : \mathsf{C} \equiv \mathsf{D}$*, such that* $\alpha \in \mathcal{T}$*, where* D *is either a concept name in* $\mathcal{T}$*, or a complex concept such that* $\mathsf{Sig}(\mathsf{D}) \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$.

For example, let us consider $\mathcal{T}^{Family}$, a small $\mathcal{ALC}$-TBox describing the family domain, shown in Fig. 1 (upper). The concept Parent, defined by the axioms α_1 and α_2, is the only explicitly defined concept in the ontology. Similarly, we can define implicitly definable concepts:

Definition 2 (Implicitly definable concept). *Let* C *be a concept name,* $\mathcal{T}$ *a TBox, and* Σ *a signature, where* $\mathsf{C} \in \mathsf{Sig}(\mathcal{T})$*, and* $\Sigma \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$. C *is implicitly definable from* Σ *under* $\mathcal{T}$*, if and only if for any two models* $\mathcal{I}$ *and* $\mathcal{K}$ *of* $\mathcal{T}$*,* $\Delta^{\mathcal{I}} = \Delta^{\mathcal{K}}$*, and for all entity* $P \in \Sigma$*,* $P^{\mathcal{I}} = P^{\mathcal{K}}$*, then it holds that* $\mathsf{C}^{\mathcal{I}} = \mathsf{C}^{\mathcal{K}}$.

Given the example, it can be seen that both Mother and Father are implicitly defined concepts in $\mathcal{T}^{Family}$, and each has six syntactically different, but semantically equivalent definitions (Fig. 1, lower).

The *number of possible rewritings* of a defined concept, regardless of whether it is explicitly or implicitly defined, is potentially exponential in the size of the ontology. Descriptions of defined concepts (i.e. the right-hand side of a non-primitive concept definition axiom) are built inductively using other, potentially defined concepts. Thus, the number of possible concept rewritings is dependent on the definability of its constituent concepts. As the definability of any defined description member concept is dependent on the definability of its own description, definability is therefore a *recursive notion* [8].

Deciding Definability. A particular concept name C can either be defined *explicitly* or *implicitly* under an ontology, or be *undefined*. Explicit definability

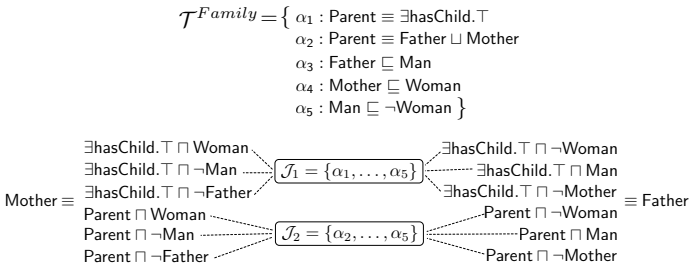

Fig. 1. This small ontology describes a family domain. Concepts Mother and Father are implicitly defined in $\mathcal{T}^{Family}$, hence these are also explicitly definable, as shown by their definition axioms. Each axiom is explained by a justification $(\mathcal{J}_1, \mathcal{J}_2)$, denoted with dashed line.

is a syntactic notion; deciding whether C is explicitly defined under an ontology is the trivial process of searching the TBox for a concept equivalence axiom whose left-hand side is C, and the potentially complex concept on the right-hand side does not include C (e.g. $\mathsf{C} \equiv \mathsf{D}$ where $\mathsf{C} \notin \mathsf{Sig}(\mathsf{D})$). In contrast, implicit definability is a semantic notion whose detection requires reasoning. Ten Cate et.al. have shown that in DLs, testing implicit definability can be reduced to entailment checking [21]; in this article, $\textsc{ImpDef}(\mathsf{C}, \Sigma, \mathcal{T})$ denotes the function that determines whether a concept C is implicitly definable under a TBox $\mathcal{T}$ given a signature Σ). The computational complexity of determining whether a concept is implicitly defined depends on the complexity of the entailment check, which is predicted on the expressivity of the given DL language. Thus it is potentially exponential in the size of the ontology, for expressive DL dialects.

Justifying Definability. It is often difficult for humans to identify the axiom set in a TBox that implies definability. *Justifications* [12] can be used to validate definability and to provide a set of axioms supporting an entailment. A justification $\mathcal{J}$ for an entailment η in an ontology is the ontological fragment in which η holds (i.e. a set of TBox axioms such that $\mathcal{J} \subseteq \mathcal{O}$). A justification is *minimal*, if the entailment in question does not follow from any proper subset of the justification. For example, if we assume $\mathcal{O} = \{\mathsf{A} \sqsubseteq \mathsf{B}, \mathsf{B} \sqsubseteq \mathsf{C}, \mathsf{D} \sqsubseteq \exists \mathsf{r}.\mathsf{C}\}$ and the axiom $\alpha : \mathsf{A} \sqsubseteq \mathsf{C}$, then $\mathcal{O} \models \alpha$ holds as $\{\mathsf{A} \sqsubseteq \mathsf{B}, \mathsf{B} \sqsubseteq \mathsf{C}\} \subseteq \mathcal{O}$; i.e. the entailment is justified[2]. The algorithm checking for implicit definability can be modified to compute not only whether a concept is definable with respect to a given signature, but to also provide its justifications.

Definability of Roles. Determining role definability[3] can also be achieved by using the same method outlined for determining concept definability. However, whilst concepts are defined using other concept and role names, roles are defined only in terms of other role names; therefore the entailment check is re-

[2] Horridge et. al. [12] introduced an efficient approach that computes either a single, or all justifications of an entailment.

[3] In this paper, we only focus on concept definability, and omit the description of the approach for determining role definability. However, a full description of the algorithm for deciding role definability is available in [8].

stricted to the RBox (role axioms), where the definition signature contains role names only.

3 Minimal Definition Signatures (MDSs)

This section describes the approach for finding the *complete set of definition signatures* of a particular defined concept, in any DL language where the Beth definability property holds. A definition signature can be defined as:

Definition 3 (Definition Signature (DS)). *A set of entity names Σ is a definition signature of the concept C under a TBox $\mathcal{T}$, if and only if there is a complex concept D, such that $\mathsf{Sig(D)} \subseteq \Sigma$, and $\mathcal{T} \models \mathsf{C} \equiv \mathsf{D}$, where $\Sigma \subseteq \mathsf{Sig}(\mathcal{T}) \setminus \{\mathsf{C}\}$.*

If a concept C is defined in an ontology, then we can entail that there exists some subset Σ of the ontology signature that implicitly defines C, i.e. members of Σ can be used to construct the right-hand side of a definition axiom for C. We only focus on *acyclic definitions*, as definitions with direct cycles (where the defined concept appears in its corresponding description) have no use in rewriting (as such signatures do not permit the substitution of defined entities), thus are excluded by this definition. As definition signatures may contain *redundant members*, their size could be very large, hence we introduced the notion of *signature minimality*:

Definition 4 (Minimal Definition Signature (MDS)). *A signature Σ is a minimal definition signature of a defined concept C under a TBox $\mathcal{T}$, if there exists no other definition signature Σ' such that $\Sigma' \subset \Sigma$.*

The *minimality* property of an MDS refers to minimising the size of the signatures, by eliminating superfluous entities. However, a defined concept may have multiple *unique* MDSs (where the difference of any two MDSs is not an empty set) under an ontology, with the same cardinality. From the definition, it follows that every MDS is also a DS, and any DS may contain at least one, but potentially many MDSs. For example, in the $\mathcal{T}^{Family}$ example (Fig. 1), the signature $\Sigma = \{\mathsf{hasChild}, \mathsf{Man}, \mathsf{Woman}\}$ is a DS of all three defined concepts in the TBox. However, this signature is not a minimal DS of Parent, because it can be defined by the following two MDSs: $\{\mathsf{hasChild}\}, \{\mathsf{Mother}, \mathsf{Father}\}$; as formalised by axiom α_1 and α_2, respectively.

Finding MDSs can be computationally expensive, as the number of definitions themselves can be exponential in the size of the ontology. Furthermore, the set of candidate signatures is equivalent to the power set of the TBox signature (excluding the defined concept itself, i.e. $2^{\mathsf{Sig}(\mathcal{T})\setminus\{\mathsf{C}\}}$). In order to reduce this complexity, *modularisation* [16] is used as space reduction mechanism. As modules preserve all entailments with respect to a signature consisting of a concept name, any MDS of a defined concept is contained in the module signature [8]. *Syntactic locality based modules* (LBM) have been shown to be sound approximations of *semantic locality based modules* (that preserve entailments over all

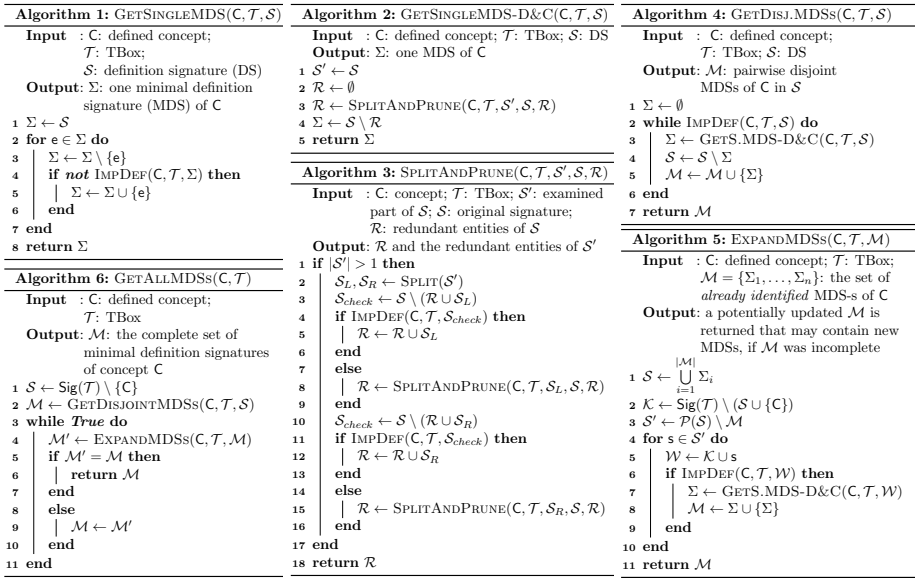

Fig. 2. The algorithms computing Minimal Definition Signatures.

the terms that occur in the module) [5], and there are efficient and widely used polynomial time algorithms for extracting syntactic LBMs[4]. As modules can be considerably smaller compared to the original ontology, modularisation is an effective mechanism for reducing the complexity of computing MDSs.

The basic idea behind *computing a single MDS* (Alg. 1, Fig. 2) is that the input DS (i.e. the signature of the module which describes a given defined concept) is iteratively pruned until it is reduced to a subset that is an MDS. Pruning is achieved by removing a member of the DS and testing the remaining signature to check if it still implicitly defines the given concept. If so, then the entity is redundant as opposed to being required. Algorithm 1 has linear time complexity, as each member of the input is examined exactly once. However, it is worth noting that this *excludes the complexity of the implicit definability check*, which is delegated to an external oracle i.e. a reasoner; thus it always depends on the DL expressivity of the ontology, and that can be exponential in the size of the ontology. Therefore, in order to reduce the overall complexity, we decrease the number of implicit definability checks. By applying a *divide and conquer strategy* (Alg. 2&3) pruning is carried by testing (and removing) entity groups, instead of individual entities. Algorithm 2&3 has logarithmic time complexity w.r.t the size of the module signature in the best case scenario, polynomial in the worst case[5]. Algorithms 1 and 2&3 are complete and correct, as if the input is a valid DS, by

[4] The OWL API provides methods for extracting several types of LBMs.

[5] Both Algorithm 1, and 2&3 are used in practice, where the former is better suited for computing an MDS from a DS that is the RHS signatures of an explicit definition, as such signature typically contain either none, or only a few redundant members (i.e. reaching the worst-case scenario of Alg. 2&3)

only removing redundant entities, they reduce it to a minimal DS. Algorithm 4 computes a *set of pairwise disjoint MDSs*, in polynomial time. The input DS is iteratively reduced to an MDS, and the resulting MDS is subtracted from the input signature. This is repeated until the input signature is no longer a DS of the defined concept, thus the algorithm is complete and correct.

In order to find the *complete set of MDSs* (Alg. 6), the first step requires computing of a set of pairwise disjoint MDSs. After this initial step, any unidentified MDS must overlap with an identified MDS. The rest of the process is delegated to Algorithm 5, which either expands on an existing but incomplete set of MDSs, or confirms its completeness. The expansion process involves iterating through the power set of the union of disjoint MDSs ($\mathcal{S}$, the smallest known set of related signature members of the defined entity), where each subset is combined (into $\mathcal{W}$) with the set of non-MDS entities ($\mathcal{K}$) in the ontology (or module) signature, and tested for validity. Despite its exponential computational complexity, the approach is feasible to use for most real-world ontologies (because $\mathcal{S}$ is typically small), as we show with the empirical evaluation presented in Section 5. A more exhaustive description of the algorithms presented above can be found in [8].

4 Definition Patterns

As part of our empirical investigation, we have computed the DSs for numerous ontologies (Sec. 5), and validated them by obtain the corresponding justifications. An explicit concept definition is always formalised as a single axiom, whereas the definition of an implicitly defined concept is derived from an axiom set (i.e. a justification); thus, implicit definitions are often not straightforward to recognise and interpret. By studying the composition of DSs (their cardinality, and the type and number of their member entities) together with their justifications (their size, and the type of their constituent axioms) we have identified a number of *definition patterns*. The patterns aim to generalise the frequent forms of creating definitions; these were inspired by ontology alignment design patterns [17], and by the predefined axiom types in OWL (where the atomic axioms form complex constructs). In addition to the validation and the interpretation of definability, the identifiable definition patterns permit *rule-based definition axiom generation*, i.e. the generation of an explicit definition of a defined entity, according to an inference rule, by processing a given DS and a justification. In contrast to general rewriting methods, this approach is language independent and does not require ontological reasoning (apart from obtaining a single justification, which is achievable in polynomial time [12]). Furthermore, the computational complexity is polynomial in the size of the input. The set of patterns is not exhaustive, i.e. it is not guaranteed to represent all definitions, however as shown by the empirical analysis, it covers a significant portion of cases[6].

Table 1 presents the list of patterns, showing the composition of the minimal definition signature (Σ), the corresponding justification in terms of its size ($|\mathcal{J}|$)

[6] Further details on patterns, and the axiom generation algorithm are presented in [8].

Table 1. Basic definition patterns

Entity	ID	Pattern	Σ (MDS)	Justification		Defined	
				$\lvert\mathcal{J}\rvert$	$\mathcal{J}$	Expl.	Impl.
Concept	*1*	Explicit definition	$\lvert\Sigma\rvert > 1$	$= 1$	$\{\mathsf{C} \equiv \exists \mathsf{r}.\top \sqcap \mathsf{D}\}$	C	
	2	Explicit synonym	Σ: 1 concept	$= 1$	$\{\mathsf{C} \equiv \mathsf{D}\}$	C, D	
	3	Implicit synonym	Σ: 1 concept	> 1	$\{\mathsf{C} \sqsubseteq \mathsf{D}, \mathsf{D} \sqsubseteq \mathsf{C}\}$		C, D
	4	Disjoint union	$\lvert\Sigma\rvert > 1$	> 1	$\{\mathsf{C} \equiv \mathsf{D}_1 \sqcup \mathsf{D}_2, \mathsf{D}_1 \sqsubseteq \neg \mathsf{D}_2\}$	C	$\mathsf{D}_1, \mathsf{D}_2$
	5	Role domain concept	Σ: 1 role	> 1	$\{\mathsf{C} \sqsubseteq \exists \mathsf{r}.\top, \exists \mathsf{r}.\top \sqsubseteq \mathsf{C}\}$		C
	6	Role range concept	Σ: 1 role	> 1	$\{\top \sqsubseteq \forall \mathsf{r}.\mathsf{C}, \mathsf{C} \sqsubseteq \exists \mathsf{r}^-.\top\}$		C
	7	Synonym role (domain or range)	Σ: 1 role	> 1	$\{\mathsf{C} \equiv \exists \mathsf{r}.\top, \mathsf{r} \equiv \mathsf{s}\}$		C
	8	Inverse role (domain or range)	Σ: 1 role	> 1	$\{\mathsf{C} \equiv \exists \mathsf{r}.\top, \mathsf{r} \equiv \mathsf{s}^-\}$		C
Role	*9*	Explicit definition	$\lvert\Sigma\rvert > 1$	$= 1$	$\{\mathsf{r} \equiv \mathsf{s} \circ \mathsf{q}\}$	r	
	11	Explicit synonym	Σ: 1 role	$= 1$	$\{\mathsf{r} \equiv \mathsf{s}\}$	r, s	
	12	Explicit inverse	Σ: 1 role	$= 1$	$\{\mathsf{r} \equiv \mathsf{s}^-\}$	r	s
	13	Implicit synonym	Σ: 1 role	> 1	$\{\mathsf{r} \sqsubseteq \mathsf{s}, \mathsf{s} \sqsubseteq \mathsf{r}\}$		r, s
	14	Implicit inverse	Σ: 1 role	> 1	$\{\mathsf{r} \sqsubseteq \mathsf{s}^-, \mathsf{s} \sqsubseteq \mathsf{r}^-\}$		r, s

and the set of axioms forming the justification. The right column presents the concepts or roles that are explicitly or implicitly defined in a pattern. In some patterns, more than one entity can be defined, for example, in an explicit, or an implicit synonym concept pattern, both concepts C and D are defined, however the actual members of the particular MDS depend on which defined entity the patterns refers to ($\Sigma^{\mathsf{C}} = \{\mathsf{D}\}, \Sigma^{\mathsf{D}} = \{\mathsf{C}\}$). For example, in $\mathcal{T}^{Family}$ (Fig. 1) the set of axioms $\{\alpha_2, \ldots, \alpha_5\}$ form the justification in a *disjoint union* pattern , where the concept Parent is defined explicitly, and the concepts Mother, Father are both defined implicitly by the same MDS $\Sigma = \{\mathsf{Parent}, \mathsf{Mother}, \mathsf{Father}\}$. The definability cases that do not fall into these basic patterns, are arbitrary *combinations* of the basic ones, meaning that the justification of such combination has one or more subset that is the justification (i.e. a minimal set) of some other defined entity which contribute to the definition. Fig. 3 presents an example of a pattern combination, where the justification entails the definition of concept Invited_speaker; furthermore, it contains two explicit definitions of the concept Conference_contribution, one explicit definition of Regular_contribution, and a disjoint union pattern of Invited_talk. Although our rule-based approach does not produce a definition axiom for Invited_speaker, it generates an axiom for all the other defined entities, that are characterised by processable patterns.

5 Empirical Analysis

In this section we empirically investigate the occurrence of definability in existing ontologies and the impact it has in supporting semantic interoperability.

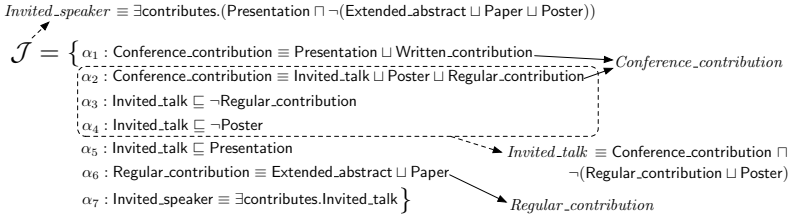

Fig. 3. Example combinations of basic patterns in *Conference.owl*, where respectively, explicit, or implicit definability is denoted with a normal, or a dashed line.

Table 2. Comparing defined and undefined ontology properties

(a) OWL Profiles									
	EL only	*EL*	*QL only*	*QL*	*RL only*	*RL*	*DL only*	*DL*	*Full*
Defined	1.99%	5.89%	0.81%	3.97%	7.69%	11.60%	48.76%	58.62%	36.17%
Undefined	1.76%	9.36%	1.32%	7.29%	7.22%	15.26%	38.88%	56.60%	42.34%

(b) OWL Constructors							
	AL	*C*	*D*	*E*	*F*	*H*	*I*
Defined	82%	28.72%	51.36%	12.33%	15.29%	37.58%	45.40%
Undefined	84%	26.07%	45.02%	9.85%	14.08%	30.13%	37.83%
	N	*O*	*Q*	*R*	*S*	*TRAN*	*U*
Defined	31.04%	29.24%	5.44%	2.84%	18.01%	7.70%	10.83%
Undefined	24.68%	32.62%	4.92%	3.71%	14.02%	6.37%	8.63%

(c) Logical Axioms		
	Defined	Undefined
<10	8.94%	25.42%
10-	33.33%	40.74%
101-	50.21%	26.42%
1001-	6.76%	6.14%
10001-	0.76%	1.29%

(d) Ratio of defined entities in ontologies in the corpus											
Defined Ratio	*0%*	*<0-10%*	*11-20%*	*21-30%*	*31-40%*	*41-50%*	*51-60%*	*61-70%*	*71-80%*	*81-90%*	*91-100%*
Nb of Ont	50.10%	9.01%	7.87%	7.46%	6.37%	6.63%	3.14%	2.88%	1.41%	1.20%	3.93%

We also analyse the behaviour of the proposed algorithms to compute the definability of ontological terms. The underlying assumption we make is that the definability status (undefined, or defined: explicitly and/or implicitly) of ontology signature entities, and the number of MDSs of defined entities provide a measure of the usability of an ontology in semantic interoperability[7]. Thus, in order to gain insight on whether, in practice, the use of definability signatures would contribute to ontology alignment in particular, and ontology engineering in general, we have analyzed the *prevalence and the extent of definability* over a wide range of OWL ontologies. Furthermore, we also study the behaviour of the proposed approximations to compute MDSs in terms of run time taken for each of the stages necessary to compute the MDSs. In the first experiment we distinguish between *defined* and *undefined* ontologies (depending on whether they contain at least one defined concept, or no defined entities), and we examine the definability status and type for each entity in all the ontologies of a large and diverse corpus, and considered several characteristics of the defined and the undefined ontologies. A second aim of this empirical analysis is to *characterise the behaviour and assess the practical applicability* of the proposed definability computation algorithms. This is achieved by measuring the processing time of computing MDSs in a corpus of made of 'semantically rich' ontologies, i.e. that contains a large portion of defined entities and MDSs. We remind the reader (Section 3), that definability computation is a three step process: first the definability status of each entity is established, next the disjoint MDSs are obtained, finally any potentially unidentified MDS is computed (i.e. the complete set of MDSs). While the first two steps are polynomial (excluding the complexity of the implicit definability check, i.e. an entailment check which depends on the ontology language expressivity), the third step has exponential time complexity.

The *experimental framework* used to run this analysis is implemented in Java; the OWL API is used for ontology manipulation and for interacting with the reasoners [13], whilst the OWL Explanation API [12] is used to compute justifications. The framework utilizes both the HermiT [9] and Pellet [19] reasoners[8].

[7] For example, given two versions of an ontology, the one with more defined entities or higher MDS to entity ratio is more valuable, as it may permit the expression of more entities with alignments, that are typically incomplete [7].

[8] HermiT performs faster with most datasets, however Pellet was able to load and process some ontologies that HermiT could not (due to ontologies using datatypes

Table 3. *Cost measured in terms of different characteristics* (time, and number of definability checks $\#Imp$), and *results* ($Def\%$: definable entities in an ontology, $\mathcal{M}$: number of MDSs per defined entity) of the three stages of definability computation.

Ontology ID	Name	$\mathcal{DL}$ Expressivity	Logical Axioms	$(\mathcal{C} \cup \mathcal{R})$	(1) Definability Status Time	#Imp	Def%	(2) Disjoint MDSs Time	#Imp	$\mathcal{M}$	(3) All MDSs Time	#Imp	$\mathcal{M}$
Conference corpus													
1	cmt	$\mathcal{ALCIN}(\mathcal{D})$	226	86	2.88s	86	51.16%	1.85s	99	1.09	0.54s	36	1.09
2	conference	$\mathcal{ALCHIF}(\mathcal{D})$	285	109	3.35s	109	65.14%	6.63s	352	1.13	307.54s	19833	1.54
3	confOf	$\mathcal{SIN}(\mathcal{D})$	196	57	2.92s	57	15.79%	2.67s	195	2.33	8.03s	573	3.33
4	edas	$\mathcal{ALCOIN}(\mathcal{D})$	739	138	20.84s	175	28.99%	28.63s	397	1.18	23.22h	1509274	2.80
5	iasted	$\mathcal{ALCIN}(\mathcal{D})$	358	182	16.14s	182	17.58%	150.89s	212	1.25	1058.74s	1756	1.75
6	sigkdd	$\mathcal{ALEI}(\mathcal{D})$	116	77	3.22s	77	28.57%	2.55s	122	1.50	0.62s	61	1.55
			384.00	**129.80**	**9.87s**	**137.20**	**41.45%**	**38.64s**	**275.40**	**1.70**	**4.72h**	**306306.60**	**2.41**
LargeBio corpus													
7	NCI_fma	$\mathcal{ALC}$	9083	6552	5.97h	6552	29.99%	222.38h	229206	1.31	115.76h	118456	1.31
8	SNOMED_fma	$\mathcal{ALER}$	20243	13431	21.49h	13431	21.47%	234.75h	109980	1.09	756.53h	384930	1.36
9	SNOMED_nci	$\mathcal{ALER}$	71042	51180	392.95h	51180	57.87%	1885.39h	1145057	1.07	3991.85h	2619125	1.08
			33456.00	**23721.00**	**140.14h**	**23721.00**	**36.44%**	**780.84h**	**494747.53**	**1.16**	**1621.38h**	**1040837.00**	**1.25**

All of the data and software, including computed DSs, definition patterns, and other results are available online[9]. The *evaluation corpus* has been assembled from a variety of OWL ontology datasets, including ontologies in the Manchester Ontology Repository [10], datasets used in the OAEI evaluation challenge[11], and a sizable set of ontologies obtained by crawling the Web [14]. In particular, this evaluation corpus consists of 3576 ontologies of different size, DL expressivity, conceptualized domain, and source of origin (professional, academic etc.). The second experiment uses a sample of 9 ontologies, and is carried out on a 16GB RAM, 10 core processor machine (with 8GB memory used by the JVM).

In order to determine which ontology characteristics (if any) are affected by definability, the first experiment examines each concept in all 3576 ontologies with the aim to determine whether they are undefined, or defined either explicitly or implicitly. Therefore, we aim to assess the prevalence of (implicitly) definable concepts in state of the art ontologies. The hypothesis tested in this experiment is that definability occurs in ontology irrespective on the ontology characteristics e.g. size, expressivity etc. The analysis of the entire corpus classifies 1703 ontologies (49.89%) as defined, i.e. that contain *at least one* defined concept. Out of all concepts, 75.82% are undefined, 20.74% are explicitly and 3.44% are implicitly (only) defined. Table (2.d) shows the proportion of ontologies in the corpus, binned by the ratio of defined to undefined concepts within an ontology. Table (2.c) presents the relative distribution of defined and undefined ontologies, binned by the number of logical axioms; Table (2.a) shows the distribution of OWL profiles in defined and undefined ontologies; and Table (2.b) shows the OWL constructor usage in defined and undefined ontologies. Apart from some outliers, the results show an even distribution of defined and undefined ontologies, w.r.t. size, and OWL profiles and constructors, thus definability may occur in any type of ontology, regardless of the employed DL language, the size of an ontology, the conceptualised domain of interest, or its origin (source of

that are not part of the OWL 2 datatype map and no custom datatype definition was given).

[9] `http://www.csc.liv.ac.uk/~dgeleta/ontodef.html`

[10] `http://owl.cs.manchester.ac.uk/tools/repositories/`

[11] `http://oaei.ontologymatching.org`

Table 4. Minimal Definition Signatures and their corresponding Definition Patterns

$\mathcal{O}$	Minimal Definition Signatures (MDSs)									Definition Patterns									
	|MDS|	per Def. Entity				Cardinality				Concept									Role
		min	avg	med	max	min	avg	med	max	1	2	3	4	5	6	7	8	Cmb.	11
Conference corpus																			
1	48	1	1.09	1	3	1	1.06	1	3	9.68%	0.00%	0.00%	0.00%	29.03%	19.35%	3.23%	0.00%	0.00%	38.71%
2	109	1	1.54	1	14	1	2.07	2	5	2.08%	0.00%	0.00%	0.00%	8.33%	4.17%	0.00%	0.00%	2.08%	83.33%
3	30	1	3.33	2	8	1	1.43	1	2	0.00%	0.00%	0.00%	0.00%	46.67%	3.33%	0.00%	0.00%	43.33%	6.67%
4	112	1	2.80	1	34	1	3.05	3	6	7.34%	0.00%	0.00%	21.10%	5.50%	5.50%	0.92%	0.00%	21.10%	38.53%
5	56	1	1.75	1	5	1	1.68	2	5	26.79%	1.79%	7.14%	0.00%	0.00%	0.00%	21.43%	0.00%	14.29%	28.57%
6	31	1	1.55	1	6	1	1.13	1	2	2.68%	0.00%	0.00%	3.57%	3.57%	0.89%	0.89%	0.89%	62.50%	25.00%
	64.33	**1.00**	**2.01**	**1.17**	**11.67**	**1.00**	**1.74**	**1.67**	**3.83**	**8.09%**	**0.30%**	**1.19%**	**4.11%**	**15.52%**	**5.54%**	**4.41%**	**0.15%**	**23.88%**	**36.80%**
LargeBio corpus																			
7	2583	1	1.32	1	5	2	5.07	4	33	67.52%	0.00%	0.00%	22.45%	0.00%	0.00%	0.00%	0.00%	10.03%	0.00%
8	3929	1	1.36	1	30	1	4.80	4	720	16.03%	0.00%	0.00%	17.54%	0.00%	0.00%	0.00%	0.00%	66.43%	0.00%
9	31831	1	1.08	1	5	1	5.59	6	15	16.90%	0.00%	0.00%	15.53%	0.00%	0.00%	0.00%	0.00%	67.55%	0.00%
	12781.00	**1.00**	**1.25**	**1.00**	**13.33**	**1.33**	**5.15**	**4.67**	**256.00**	**33.48%**	**0.00%**	**0.00%**	**18.51%**	**0.00%**	**0.00%**	**0.00%**	**0.00%**	**48.00%**	**0.00%**

creation). The only property, which affects the level of definability in an ontology, is, unsurprisingly, the granularity of conceptualisation.

The *second experiment* investigates the feasibility of the algorithms for computing definability by analysing their behaviour and performance. The corpus used in this experiment is made of small ontologies that conceptualise the conference domain (Conference track, OAEI corpus), and 3 vast biomedical ontologies (LargeBio track, OAEI corpus). The aim of this experiment is to assess the time taken by the proposed approximation when computing MDS over a variety of ontologies. Table 3 presents the characteristics of the *sample corpus* (DL expressivity, number of logical axioms, and the number of concepts and role names, $(\mathcal{C} \cup \mathcal{R})$), and the experiment results. The three numbered partitions show the results of the definability computation steps, where each step is measured in terms of the computation time (this is given either in seconds, or in hours in some cases), and the number of implicit definability checks ($\#Imp$). The first stage establishes the definability status of each concept and role, hence $Def\%$ denotes the ratio of defined entities in the ontology signature. In the other two stages, where first the disjoint MDSs, then all MDSs are computed, $\mathcal{M}$ denotes the MDS to defined entity ratio. In general, as it can be anticipated, the larger, more expressive ontologies take much longer to compute than the smaller, less expressive ones. The definability status and the disjoint MDS computation stages are feasible for both small and large ontologies; whereas obtaining the complete set of MDSs (stage 3) is a considerably more costly operation, in most of the cases[12]. However, despite the cost of the last stage, the difference between the number of MDSs found during stage 2 (on average 1.70 MDSs per defined entity in the small, and 1.16 in the large ontologies) and 3 (2.41, and 1.25 MDSs), in many cases is negligible (0.71, and 0.09 MDSs more per entity). A notable case is the small *edas* ontology, where the first two stages take only 49.47 *seconds* to complete, however, the last stage takes 23.22 *hours*, although the MDSs to defined entities ratio has more than doubled (from 1.18 to 2.80). In the *confOf* ontology, the last stage also shows a significant increase, from 2.33 MDSs per entity to 3.33, but in this case the computation time is close (8.03 sec) to the sum of the two prior steps (5.59 sec). Table 4 provides further information

[12] The MDS expansion (Alg. 5) is restricted to computing an MDS union size $\mathcal{S} \leq 20$. This excluded no entities in the Conference, and 441 in the LargeBio corpus.

about the computed MDSs (for each ontology in the sample corpus), where the left partition shows the total number of different MDSs in the ontology (|MDS|), the number of MDSs per defined entity, and the cardinality of MDSs; the right partition presents the distribution of MDSs w.r.t. to the corresponding definition patterns[13]. MDS per entity scores show that in most ontologies, about half of the defined entities have only one definition, and there are only a few entities in each ontologies with large number of MDSs. The average cardinality of an MDS is considerably low, with 1.74 entities per MDS in the small, and 5.15 in the large ontologies; however, there are some extreme cases, such as the *SNO_nci* ontology, where one MDS contains 720 entities. In the Conference corpus, 52.51% of all MDSs correspond to a single entity (i.e. |MDS| = 1) definition pattern. Out of all MDSs in the Conference corpus only 23.88% of all cases in this corpus are combined, thus for the majority of cases, our rule-based rewriting approach is sufficient; however, in the LargeBio corpus it only covers 52.00% of all MDSs.

6 Definability and Ontology Alignment

Definability can be exploited in a number of novel contexts; in particular in this paper we argue that MDSs coupled with justification-based explanations [12] can support ontology alignment by identifying seemingly unrelated entities that are used to describe defined entities, and can thus be used to produce (in the case of identifiable patterns) new definition axioms (Section 4). Ontology alignment addresses the problem of mapping terms in heterogeneous ontologies and aims to create *alignments*, i.e. sets of correspondences between semantically related entities in different ontologies [7]. Over the past decade more than several alignment approaches have emerged [7]. However, neither state of the art matching systems, nor evaluation measures that assess the quality of alignments, have considered the notion of rewriting. As rewriting permits defined entities to be expressed in syntactically different but semantically equivalent forms, we argue, that an entity is rewritable under an alignment if the entities of its MDS are mapped by the alignment, thus rewriting entails a new type of correspondence, based on the definitions of entities. For instance, given an ontology $\mathcal{O} = \{\mathsf{C} \equiv \mathsf{A} \sqcup \mathsf{B}, \mathsf{B} \sqsubseteq \neg\mathsf{A}\}$, and alignment $\mathcal{A} = \{\langle \mathsf{C}, \mathsf{C}', \equiv\rangle, \langle \mathsf{B}, \mathsf{B}', \equiv\rangle\}$, which maps $\mathcal{O}$ to $\mathcal{O}'$, the implicitly defined concept A is rewritable w.r.t. the alignment, yielding $\langle \mathsf{A}, \mathsf{C}' \sqcap \neg\mathsf{B}', \equiv\rangle$, a definability-based correspondence. This *complex correspondence* describes a relation between a defined entity (or its description) in one ontology, and a complex concept (or role) in an aligned ontology. We suggest that definability-based mappings can potentially 1) increase *alignment coverage* [4] (the ratio of elements of the ontology which are mapped[14]) as they can cover otherwise uncovered entities; 2) increase *coverage retention*, i.e. removing some mappings from an alignment does not necessarily effect its expressive capacity (i.e. coverage) as some entities

[13] Pattern numbers reference Table 1, *Cmb.* denotes those MDS cases that do not correspond to any individual pattern, but to a combination of patterns. Patterns that have no MDS in the sample corpus are omitted for brevity.

[14] An entity e is covered by a mapping c in an alignment $\mathcal{A}$ iff $\{\exists c \in \mathcal{A} | c : \langle e, e', r\rangle\}$.

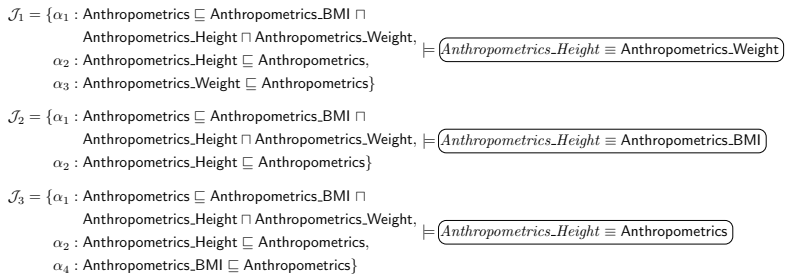

Fig. 4. Modelling error: three concepts that should be different, are semantically equivalent to each other (unwanted synonyms in the Bioportal corpus, bp26.owl).

may be mapped by both an asserted, and a definability-based correspondence; 3) increase *compactness*, i.e. for a given knowledge-based task signature, only a subset of an alignment may be necessary to provide coverage.

7 Definability and Ontology Modelling

The empirical analysis presented in previous section has also highlighted how the computation of MDSs can help in identifying modelling errors in ontologies.

We have formalised three types of errors, each of which can be automatically detected[15], but their repairs requires the involvement of an ontology engineer and a domain expert. *(1) Implicit definability by an empty signature:* the only concept in any ontology, which requires no signature for its definition is $\top$. If a named concept is definable by an empty signature, then the ontology is most likely to contain an error, or purposely define the concept as the synonym of $\top$. For example in the *cocus.owl* ontology of the Conference corpus $\mathsf{Person} \equiv \top$. By examining the document, it becomes obvious that this is unintentional, as the ontology contains many other concepts (such as Conference) that are definitely not semantically related to Person. *(2) Unwanted synonym(s):* These occur when two or more concepts, meant to convey different meaning, are wrongly represented as interchangeable synonyms of one another. Fig. 4 shows three different ways of defining the concept Anthropometrics_Height. Obviously, A._Height, A._Weight and A._BMI are semantically related, but different concepts. However, in TBox $\mathcal{T}$ where $(\mathcal{J}_1 \cup \mathcal{J}_2 \cup \mathcal{J}_3) \subseteq \mathcal{T}$, these concepts are defined as equivalent. The correction requires expert knowledge[16]. *(3) Redundant concept(s):* this is not necessarily an error, but a discrepancy between

[15] Error #1 and #2 can also be detected without MDSs, via classifying the ontology and inspecting the resulting tree for unsatisfiable or equivalent concepts, respectively.

[16] Anthropometrics means measurement of the size and proportions of the human body. Axioms $\alpha_2, \alpha_3, \alpha_4$ are correct, as *height, weight* and *BMI* are all type of measurements that make up the general class Anthropometrics, but axiom α_1 is incorrect as *height* and *weight* measurements would share nothing in common, i.e. their intersection would be empty. The correct representation would be to describe Anthropometrics as a disjoint union of these concepts.

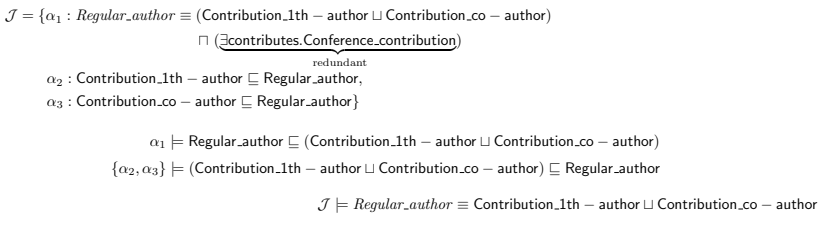

Fig. 5. Redundant concepts in explicit definition (Conference corpus, conference.owl).

the *intended meaning* (formalised by explicit definition axioms), and the *actual meaning* (the alternative explicit definition, which corresponds to an MDS of the defined concept). Fig. 5 presents an example of this case, here Regular_author is defined by axiom α_1, however, its signature is not a minimal, because its subset $\{\mathsf{Contribution\_1th - author}, \mathsf{Contribution\_co - author}\}$ can also be used to define the concept, as it is implied by the justification. An argument can be made, that an explicit concept definition ought to be a succinct representation, meaning that it should only consist of those entities that are necessary to unambiguously describe the concept. However a knowledge engineer may add semantically redundant entities to certain definitions in order to aid human comprehension. This occurs frequently, e.g. in *SNO_nci*, 76.28% of all MDSs had redundant concepts in explicit definitions, and therefore is not considered an error.

8 Conclusions

In this paper we have presented a novel way to compute the complete set of definition signatures, and introduced a set of new application areas of concept rewriting that motivated the development of this method. In order to justify the viability of these new areas, a large and diverse set of ontologies were subjected to definability computation. This has confirmed the hypothesis that definability is prevalent in any type of ontology, although it is more likely to occur in more expressive, and semantically richer ontologies. Hence the exploitation of MDSs could indeed benefit the previously described application areas. In addition it was shown, that definability computation is feasible for most real world ontologies, and in some cases, it can be useful in dynamic environments as well, due to the fact that a subset of MDSs can be found in polynomial time. However, as the approach does not scale for larger, very expressive ontologies, a more efficient approach should be developed.

References

1. Baader, F.: The description logic handbook: theory, implementation, and applications. Cambridge University Press (2003)
2. Beth, E.W.: On Padoa's method in the theory of definition. Indagationes Mathematicae 15, 330 – 339 (1953)

3. Cheatham, M., Dragisic, Z., Euzenat, J., Faria, D., Ferrara, A., Flouris, G., Fundulaki, I., Granada, R., Ivanova, V., Jiménez-Ruiz, E., Lambrix, P., Montanelli, S., Pesquita, C., Saveta, T., Shvaiko, P., Solimando, A., dos Santos, C.T., Zamazal, O.: Results of the Ontology Alignment Evaluation Initiative 2015. In: Proceedings of the 10th International Workshop on Ontology Matching. pp. 60–115 (2015)
4. David, J., Euzenat, J., Šváb-Zamazal, O.: Ontology similarity in the alignment space. In: The Semantic Web–ISWC 2010, pp. 129–144. Springer (2010)
5. Del Vescovo, C., Klinov, P., Parsia, B., Sattler, U., Schneider, T., Tsarkov, D.: Empirical study of logic-based modules: Cheap is cheerful. In: The Semantic Web–ISWC 2013, pp. 84–100. Springer (2013)
6. Donnelly, K.: SNOMED-CT: The advanced terminology and coding system for eHealth. Studies in health technology and informatics 121, 279 (2006)
7. Euzenat, J., Shvaiko, P.: Ontology Matching, Second Edition. Springer (2013)
8. Geleta, D., Payne, T.R., Tamma, V.: Computing Minimal Definition Signatures in Description Logic Ontologies. Tech. rep., University of Liverpool (2016), `https://intranet.csc.liv.ac.uk/research/techreports/tr2016/ulcs-16-003.pdf`
9. Glimm, B., Horrocks, I., Motik, B., Stoilos, G., Wang, Z.: HermiT: an OWL 2 reasoner. Journal of Automated Reasoning 53(3), 245–269 (2014)
10. Grau, B.C., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., Sattler, U.: OWL 2: The Next Step for OWL. Web Semant. 6(4), 309–322 (Nov 2008)
11. Hoogland, E., et al.: Definability and interpolation: Model-theoretic investigations. Institute for Logic, Language and Computation (2001)
12. Horridge, M.: Justification Based Explanation in Ontologies. Ph.D. thesis, University of Manchester (2011)
13. Horridge, M., Bechhofer, S.: The OWL API: A Java API for OWL ontologies. Semantic Web 2(1), 11–21 (2011)
14. Matentzoglu, N., Bail, S., Parsia, B.: A Snapshot of the OWL Web. In: Alani, H., Kagal, L., Fokoue, A., Groth, P., Biemann, C., Parreira, J., Aroyo, L., Noy, N., Welty, C., Janowicz, K. (eds.) The Semantic Web – ISWC 2013, Lecture Notes in Computer Science, vol. 8218, pp. 331–346. Springer Berlin Heidelberg (2013)
15. Rosse, C., Mejino Jr, J.L.: The foundational model of anatomy ontology. In: Anatomy Ontologies for Bioinformatics, pp. 59–117. Springer (2008)
16. Sattler, U., Schneider, T., Zakharyaschev, M.: Which kind of module should i extract? Description Logics 477, 78 (2009)
17. Scharffe, F., Zamazal, O., Fensel, D.: Ontology alignment design patterns. Knowledge and information systems 40(1), 1–28 (2014)
18. Seylan, I., Franconi, E., De Bruijn, J.: Effective query rewriting with ontologies over DBoxes. In: IJCAI. vol. 9, pp. 923–929. Citeseer (2009)
19. Sirin, E., Parsia, B., Grau, B.C., Kalyanpur, A., Katz, Y.: Pellet: A practical OWL-DL reasoner. Web Semantics: science, services and agents on the World Wide Web 5(2), 51–53 (2007)
20. Stuckenschmidt, H., Predoiu, L., Meilicke, C.: Learning Complex Ontology Alignments A Challenge for ILP Research. In: Proceedings of the 18th International Conference on Inductive Logic Programming (2008)
21. Ten Cate, B., Franconi, E., Seylan, I.: Beth Definability in Expressive Description Logics. J. Artif. Intell. Res.(JAIR) 48, 347–414 (2013)