

Towards a Functional Approach to Modular Ontologies using Institutions

Daniel POKRYWCZYŃSKI^{a,1} and Grant MALCOLM^a

^a*Department of Computer Science, University of Liverpool, UK*

Abstract. We propose a functional view of ontologies that emphasises their role in determining answers to queries, irrespective of the formalism in which they are written. A notion of framework is introduced that captures the situation of a global language into which both an ontology language and a query language can be translated, in an abstract way. We then generalise existing notions of robustness from the literature, and relate these to interpolation properties that support modularisation of ontologies.

Keywords. Ontologies, modularity, integration, institutions, interpolation

Introduction

In what we might term the *standard approach*, an ontology is a communicative, essentially syntactic artefact. Gruber's definition [14], 'an ontology is an explicit specification of a conceptualization', is often elaborated to maintain explicitness of ontologies while also capturing their semantic use in sharing information. For example, Studer et al [25] combine Gruber's definition with that of Borst [4], giving what has become a standard definition: 'an ontology is a formal, explicit specification of a shared conceptualization'. This emphasis on sharing information is also present, for example, in Uschold and Grüninger's [26] statement that an ontology is 'used to refer to the shared understanding of some domain of interest'.

In the standard approach, the function of an ontology is to state, explicitly, a conceptualisation. However, as well as reading and writing ontologies, in practice one also wants to *use* existing ontologies, perhaps to browse its induced concept hierarchy, or to access instance data, or perhaps to create a new ontology that extends either an entire ontology or a manageably small fragment of one. Or perhaps one may want to test whether one ontology is in some way consistent with another, or provides the same information regarding some subset of concepts. Using ontologies in these ways may raise some problems.

One problem stems from the requirement that ontologies be formal. It obviously helps to meet this requirement if the ontology is written in a precise language, and a large number of languages have been developed specifically for writ-

¹Corresponding Author: Daniel Pokrywczyński, Department of Computer Science, University of Liverpool, Liverpool L69 3BX, UK; E-mail: Daniel.Pokrywczyński@liv.ac.uk

ing ontologies. The majority of these are formal languages, though there are also a number of graphical notations that are widely used. It is not the case that one language or notation is ‘better’ than all others, but one may be found more appropriate than others for a particular domain or application. In any case, choosing a particular notation for an ontology restricts the use that may be made of it. Most likely, an ontology, or its component parts, written in one language may be incorporated only into other ontologies that are written in the same language; in order to combine two or more existing ontologies, it may be necessary to translate at least one of them into another language.

Even within one language, different ontologies may use different vocabularies. For instance, this may happen if groups of primary ontology users use different natural languages and thus concept and role names are from different natural languages. Again, this makes ontologies difficult to use together. To overcome it we have to find a correspondence between the vocabularies, see e.g. [9].

Another problem stems from the requirement that ontologies be explicit. It may not be convenient, or even possible, to present all the required information explicitly in an ontology. Many of the languages used to write ontologies are *logics*, whose semantics provide a notion of consequence that provides implicit information. To obtain this information, it is necessary to query an ontology: to deduce or infer information that is implicit in the ontology’s statement. We may wish to obtain the induced concept hierarchy, or to access instance data using the ontology (e.g., as in Ontology-Based Data Access [5]). Obviously, an answer to a query may best be obtained when the query and the ontology are written in the same logical language; if they are not then, again, the ontology or the query may need to be translated into another language.

Of course a combination of the above problems may appear, so we may have two ontologies (or an ontology and a query) expressed in different formalisms, with their vocabularies also coming from different natural languages. In some sense, those who are developing and using ontologies may face problems similar to those that the builders of the Tower of Babel had. A great amount of work has been spent gathering a great amount of knowledge in a vast number of ontologies. But these ontologies are often formulated in different formal languages and use different vocabularies and thus in many cases we are unable to use them together.

As the core of a solution to these problems we propose a fresh view on what ontologies are. Contrary to the standard approach our primary focus is not on the way ontologies are built or what formalisms are used to construct them, but on the way we can use them. We want to identify an ontology with its function. Thus we adopt an abstract view of an ontology as a black box providing answers to queries about some vocabulary of interest. We call this the *functional approach*.

Identifying the function of ontologies is also important for modularity of ontologies. The significance of modularity may be observed in ontology development and application. In ontology development, it allows for re-use of existing ontologies and also for distributing work among independent groups of designers when developing a large scale ontology. In ontology application, modularity allows reasoning over only the relevant part of ontology, thus increasing efficiency.

To say that some part of an ontology is a module means that this part may function independently. Therefore we have to set the context in which the ontology

(and its part) are to function. That is, we have to fix a language that will be used to answer queries. This allows us to say that two ontologies are equivalent if in a fixed language they give the same answers to all the queries over a fixed vocabulary. For a module of an ontology, we say that it functions independently if in a fixed language both the module and the entire ontology give the same answers to all queries over a fixed vocabulary.

These intuitions are reflected in the concept of *inseparability*, which relates two ontologies that give the same answers to all queries over a fixed language. In the standard approach two ontologies are indistinguishable if and only if they have the same axioms. But as observed in [23] computing the syntactic difference between ontologies consisting of axioms is hardly useful. Whereas in the functional approach two ontologies are indistinguishable with respect to some vocabulary Σ if they are inseparable wrt the fixed query language over Σ , regardless of what the axioms are and what formalisms these ontologies are written in. One of the benefits of this is that instead of a big ontology we can freely use a part which has the same consequences relative to the vocabulary of interest: we say that this part of an ontology can ‘function independently’. Similarly, we can freely replace different ontologies with each other if they are indistinguishable in the functional approach. In [15], Konev et al studied inseparability for a variety of description logics, and proposed certain robustness properties, which they were able to relate to standard properties that support modularisation, such as interpolation properties. Our goal in this paper is to generalise their results in a way that is independent, as far as possible, from the particular formalisms in which ontologies are expressed.

The relationships between many ontology languages are well understood, and translating between them, or embedding them into richer languages, is often straightforward. Of course, the details of how one particular language is translated into another are necessarily *ad hoc*. In this paper, we introduce a notion of *framework* that captures the situation of a ‘global’ language into which both an ‘ontology’ language and a ‘query’ language can be translated, in a more general and abstract way. To do this, we use *institutions*, which were introduced by Goguen and Burstall [10, 12] to treat logics with model-theoretic semantics in a systematic way. A great many logics can be represented as institutions [8], including many logics for specifying ontologies [20]. An advantage of this is that institutions also allow a systematic treatment of translation between languages, and recent research has applied this to problems such as alignment and integration [6, 18, 19, 24, 27]. Within a framework, it is possible to capture a general notion of consequence, whereby an ontology answers a query, when both are translated into the global language. This in turn gives rise to a language-independent notion of *inseparability*. Frameworks and inseparability are introduced in Section 2, together with some useful results that allow us to work with this general notion of consequence.

Our main goal is to study how ontologies may be combined at an abstract level. In general, combining ontologies may lead to undesirable results, if the ontologies are not, in some sense, consistent with each other. Section 3 generalises the robustness properties of [15], and relates these to interpolation properties.

1. Institutions and Ontologies

Due to limitations of space, in this section we simply introduce the main definitions, of institution and of comorphism, that we will need in following sections. Moreover, we will suppress some technical details wherever we feel that an intuitive understanding of the concepts involved can be better given by illustrative example; for instance, we gloss over the notion of ‘inclusive’ institution, which has a simple intuitive sense (vocabularies are like sets, and can have unions and intersections), but a somewhat involved technical definition. More comprehensive introductions to institutions, that provide useful motivation for the definitions, can be found in the original paper by Goguen and Burstall [10, 12], or the more recent [13]. For overviews of particular relevance to ontologies see [11, 24].

We begin abruptly with the definition of institution, which captures the notion of a logic with a model theory.

Definition 1.1. An **institution** \mathcal{I} is a tuple $\mathcal{I} = (\text{Sig}^{\mathcal{I}}, \text{Sen}^{\mathcal{I}}, \text{Mod}^{\mathcal{I}}, \models^{\mathcal{I}})$, where $\text{Sig}^{\mathcal{I}}$ is a category of **signatures**, and for any signature Σ , $\text{Sen}^{\mathcal{I}}(\Sigma)$ is a set of Σ -**sentences**, $\text{Mod}^{\mathcal{I}}(\Sigma)$ is a category of Σ -**models**, and $\models_{\Sigma}^{\mathcal{I}}$ is the **satisfaction relation** between Σ -models and Σ -sentences. Moreover, $\text{Sen}^{\mathcal{I}}$ and $\text{Mod}^{\mathcal{I}}$ are required to be *functorial*: that is, for each signature morphism $\sigma : \Sigma \rightarrow \Sigma'$, there is a function $\text{Sen}^{\mathcal{I}}(\sigma) : \text{Sen}^{\mathcal{I}}(\Sigma) \rightarrow \text{Sen}^{\mathcal{I}}(\Sigma')$ (we usually just write σ instead of $\text{Sen}^{\mathcal{I}}(\sigma)$); moreover, there is a *model-reduct functor* $\text{Mod}^{\mathcal{I}}(\sigma) : \text{Mod}^{\mathcal{I}}(\Sigma') \rightarrow \text{Mod}^{\mathcal{I}}(\Sigma)$ — note that this reverses the direction of the signature morphism.

These are required to satisfy the following **satisfaction condition**: for all signature morphisms $\sigma : \Sigma \rightarrow \Sigma'$, all Σ' -models M and all Σ -sentences e

$$\text{Mod}^{\mathcal{I}}(\sigma)(M) \models_{\Sigma}^{\mathcal{I}} e \text{ iff } M \models_{\Sigma'}^{\mathcal{I}} \sigma(e)$$

In this paper, we concentrate on \mathcal{EL} [2] and First Order Logic (FOL), so we illustrate this definition with these two logics. Signatures concern the basic vocabulary of the logic. For example, in the institution of \mathcal{EL} , the signatures are pairs of sets, consisting of the primitive concept names and role names. For FOL , the signatures would be families $(\Pi_n)_{n \in \omega}$, where each Π_n is the set of n -ary relation symbols. To say that $\text{Sig}^{\mathcal{I}}$ is a category means that we are also interested in morphisms between signatures. In \mathcal{EL} , a signature morphism $(P, R) \rightarrow (P', R')$ is a pair (f, g) of functions, with $f : P \rightarrow P'$ and $g : R \rightarrow R'$, mapping concept names to concept names and role names to role names. In FOL , the signature morphisms $\Pi \rightarrow \Pi'$ are families of functions $f_n : \Pi_n \rightarrow \Pi'_n$, for $n \in \omega$, mapping n -ary symbols to n -ary symbols.

In \mathcal{EL} and FOL , it is easy to see how signature morphisms extend to sentences. In \mathcal{EL} , (P, R) -sentences (i.e., the set $\text{Sen}^{\mathcal{EL}}(P, R)$) are the general concept inclusions, $C \sqsubseteq D$, where the concepts C and D are built from the primitive concept names in P and closed under intersection and existential quantification over roles in R . A signature morphism $(f, g) : (P, R) \rightarrow (P', R')$ extends to a mapping from (P, R) -sentences to (P', R') -sentences by replacing $p \in P$ with $f(p)$, and $r \in R$ with $g(r)$. Similarly, in FOL , a signature morphism $f : \Pi \rightarrow \Pi'$ extends to a mapping of formulae by replacing each n -ary symbol π with $f_n(\pi)$.

An institution is **inclusive** if the category of signatures has ‘inclusion morphisms’, with unions and intersections, and these inclusions are preserved when extended to sentences. For the sake of brevity, we shall treat this informally by examples, rather than giving technical details (for which the interested reader can refer to [13]). In \mathcal{EL} , as in FOL , the inclusions, unions, etc., are done pointwise; for example, $(P_1, R_1) \cup (P_2, R_2) = (P_1 \cup P_2, R_1 \cup R_2)$.

In \mathcal{EL} and FOL , Σ -models and their morphisms are the standard notions of interpretative structures and homomorphisms for those models. Model reducts work in the familiar way; for example, given an \mathcal{EL} signature morphism $(f, g) : (P, R) \rightarrow (P', R')$ and (P', R') -model M , the reduct $\text{Mod}^{\mathcal{EL}} \sigma(M)$ is the (P, R) -model that interprets $p \in P$ as M interprets $f(p)$, and similarly for role names. Moreover, the satisfaction relations are the standard notions of satisfaction in \mathcal{EL} and FOL . It is straightforward to see that the satisfaction condition holds for both these institutions.

The satisfaction relation may be extended to a relation of consequence. If E is a set of Σ -sentences and e is a Σ -sentence, we write $E \models_{\Sigma} e$ to mean that every Σ -model that satisfies every sentence in E also satisfies e . In \mathcal{EL} , such a set of sentences is called a *Tbox*, and we shall refer to such a set as an *ontology*.

One benefit of viewing logics as institutions is the ability to relate different logics in a systematic way: the notion of *comorphism* specifies how a ‘weaker’ logic can be ‘embedded’ in a ‘richer’ logic.

Definition 1.2. A comorphism $\mu : \mathcal{I} \rightarrow \mathcal{I}'$ consists of:

- a functor $\Phi^{\mu} : \text{Sig}^{\mathcal{I}} \rightarrow \text{Sig}^{\mathcal{I}'}$
- a natural transformation $\alpha^{\mu} : \text{Sen}^{\mathcal{I}} \rightarrow \text{Sen}^{\mathcal{I}'} \Phi^{\mu}$ (i.e., a family of functions $\alpha_{\Sigma}^{\mu} : \text{Sen}^{\mathcal{I}}(\Sigma) \rightarrow \text{Sen}^{\mathcal{I}'}(\Phi^{\mu}\Sigma)$)
- a natural transformation $\beta^{\mu} : \text{Mod}^{\mathcal{I}'} \Phi^{\mu} \rightarrow \text{Mod}^{\mathcal{I}}$ (i.e., a family of functors, mapping models to models and homomorphisms to homomorphisms, $\beta_{\Sigma}^{\mu} : \text{Mod}^{\mathcal{I}'}(\Phi^{\mu}\Sigma) \rightarrow \text{Mod}^{\mathcal{I}}(\Sigma)$)

such that the following **satisfaction condition** is satisfied: for all \mathcal{I} -signatures Σ , all $\Phi^{\mu}(\Sigma)$ -models M , and all Σ -sentences e ,

$$\beta_{\Sigma}^{\mu}(M) \models_{\Sigma}^{\mathcal{I}} e \text{ iff } M \models_{\Phi^{\mu}(\Sigma)}^{\mathcal{I}'} \alpha_{\Sigma}^{\mu}(e) .$$

As an example, we describe how \mathcal{EL} can be translated into the richer logic FOL , giving a comorphism $\varepsilon : \mathcal{EL} \rightarrow FOL$. This is also known as the *standard translation* [3]. Given an \mathcal{EL} signature (P, R) , let $\Phi^{\varepsilon}(P, R)$ be the FOL signature whose unary relation symbols are P , whose binary relation symbols are R , and whose n -ary relation symbols are \emptyset for all other n .

Any \mathcal{EL} concept C built from the signature $\Sigma = (P, R)$ can be translated into a FOL formula $[C]^x$, built from $\Phi^{\varepsilon}(\Sigma)$ and with free variable x , as follows: for $p \in P$, $[p]^x = p(x)$, and inductively, $[C_1 \sqcap C_2]^x = [C_1]^x \wedge [C_2]^x$, and $[\exists r.C]^x = (\exists x')r(x, x') \wedge [C]^{x'}$, where x' is a new variable (we assume, as usual, a denumerable set of variables). Now for any Σ -sentence $C_1 \sqsubseteq C_2$, we define $\alpha_{\Sigma}^{\varepsilon}(C_1 \sqsubseteq C_2)$ to be the FOL sentence $(\forall x)[C_1]^x \Rightarrow [C_2]^x$.

Finally, we need to specify how $FOL \Phi^\varepsilon(\Sigma)$ -models give rise to $\mathcal{EL} \Sigma$ -models. Let M be a $\Phi^\varepsilon(\Sigma)$ -model; it has a domain, Δ , as well as an interpretation $\pi^M \subseteq \Delta^n$ for each n -ary relation symbol π . We let $\beta_\Sigma^\varepsilon(M)$ have domain Δ , and for $p \in P$, we let $p^{\beta_\Sigma^\varepsilon(M)} = p^M$, which works because p is a unary predicate symbol in $\Phi^\varepsilon(\Sigma)$. Role names are treated analogously. We omit the details of how β^ε works on homomorphisms (as well as naturality of α^ε , etc.), and leave the interested reader to check that the satisfaction condition holds for the comorphism ε .

2. Σ -entailment and Inseparability in Frameworks

In this part we focus on the fundamental notions of description logics; Σ -entailment, Σ -inseparability wrt a query language, and Σ -conservative extension. This section generalises the results presented by Konev et al in [15]. Our goal is to present these notions in an institutional setting, and thus independently of particular ontology or query languages. To achieve this we introduce the notion of a framework, which allows us to study entailment even if an ontology language differs from a query language.

Definition 2.1. A **query basis** is an inclusive comorphism $\eta : \mathcal{Q} \rightarrow \mathcal{G}$.

A **framework over a query basis** $\eta : \mathcal{Q} \rightarrow \mathcal{G}$ is an inclusive comorphism $\mu : \mathcal{L} \rightarrow \mathcal{G}$. We call \mathcal{L} the **ontology language**, \mathcal{G} the **global language**, and \mathcal{Q} the **query language** of the framework.

The intuition behind this construct is that given an ontology and a query represented in two institutions, \mathcal{L} and \mathcal{Q} respectively, we chose a global institution \mathcal{G} s.t. there are comorphisms $\mathcal{L} \xrightarrow{\mu} \mathcal{G}$ and $\mathcal{Q} \xrightarrow{\eta} \mathcal{G}$. Using these comorphisms we can translate the ontology and the query into \mathcal{G} , then we can check whether the query is a consequence of the ontology.

We allow more than one framework over a query basis. Figure 1 is a graphical representation of frameworks μ_1 and μ_2 over query basis η . This situation would

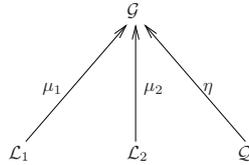


Figure 1. Frameworks μ_1 and μ_2 over a query basis η

arise if we wanted to merge two ontologies written in different languages.

Notation 1. Given a framework $\mu : \mathcal{L} \rightarrow \mathcal{G}$ over a query basis η , and Λ a $\text{Sig}^{\mathcal{L}}$ -signature, we say that $O \subseteq \text{Sen}^{\mathcal{L}}(\Lambda)$ is a Λ -ontology for μ .

The consequence relation is a basic notion in our studies, it is used to solve query answering problems and is the base for other definitions. Therefore before continuing we define this notion in the framework setting.

Definition 2.2. Let $\mu : \mathcal{L} \rightarrow \mathcal{G}$ be a framework over η , and let O be a Λ -ontology for μ and $\varphi \in \text{Sen}^{\mathcal{Q}}(\Sigma)$ be a query, with Σ in $\text{Sig}^{\mathcal{Q}}$. We say that φ is a *consequence of O wrt η* (written $O \models_{\Sigma}^{\eta} \varphi$) iff

$$\alpha_{\Lambda}^{\mu}(O) \models_{\Phi^{\mu}(\Lambda) \cup \Phi^{\eta}(\Sigma)}^{\mathcal{G}} \alpha_{\Sigma}^{\eta}(\varphi).$$

This says that in a framework μ a query φ is a consequence of an ontology O if, and only if O translated along comorphism μ into \mathcal{G} entails in \mathcal{G} , with respect to the union of translations of signatures Λ and Σ , a translation of φ along η .

For any institution \mathcal{G} , the identity $1_{\mathcal{G}} : \mathcal{G} \rightarrow \mathcal{G}$ is a query basis, and also provides a framework over itself, which is just the institution \mathcal{G} , and so the notion of consequence in a framework generalises entailment in an institution.

Proposition 2.3. For framework $1_{\mathcal{G}} : \mathcal{G} \rightarrow \mathcal{G}$ over query basis $1_{\mathcal{G}} : \mathcal{G} \rightarrow \mathcal{G}$, consequence in the framework is just consequence in \mathcal{G} ; i.e., $O \models_{\Sigma}^{1_{\mathcal{G}}} \varphi$ iff $O \models_{\Lambda \cup \Sigma}^{\mathcal{G}} \varphi$ for any Λ -ontology O and Σ -sentence φ .

Following Aiguier and Barbier [1], we introduce a notion of conservativity for comorphisms; before we do so, we require a notation for consequences in a ‘smaller’ signature:

Notation 2. Let \mathcal{I} be an institution. Let $\sigma : \Sigma \rightarrow \Sigma'$ be a signature morphism and let $O \subseteq \text{Sen}^{\mathcal{I}}(\Sigma')$. We define $O_{\sigma} = \{\varphi \in \text{Sen}^{\mathcal{I}}(\Sigma) \mid O \models_{\Sigma'}^{\mathcal{I}} \sigma(\varphi)\}$.

Definition 2.4 (Conservativity). An institution comorphism $(\Phi, \alpha, \beta) : \mathcal{I} \rightarrow \mathcal{I}'$ is *conservative* iff every signature morphism $\sigma : \Sigma \rightarrow \Sigma'$ and every set of sentences $O \subseteq \text{Sen}(\Sigma')$ in \mathcal{I} satisfy:

$$\alpha_{\Sigma}(O_{\sigma}) \models_{\Sigma'}^{\mathcal{I}'} (\alpha_{\Sigma'}(O))_{\Phi(\sigma)}.$$

Conservativity means that moving to a richer logic introduces no new consequences; the same holds for conservative frameworks:

Proposition 2.5. For any framework $\mu : \mathcal{L} \rightarrow \mathcal{G}$ over query basis μ itself, with μ conservative, and for any Λ -ontology O for μ , and $\varphi \in \text{Sen}^{\mathcal{L}}(\Lambda')$, we have:

$$O \models_{\Lambda'}^{\mu} \varphi \quad \text{iff} \quad O \models_{\Lambda \cup \Lambda'}^{\mathcal{L}} \varphi.$$

Lemma 2.6. As comorphism $\varepsilon : \mathcal{EL} \rightarrow \text{FOL}$ is conservative, we have that for every \mathcal{EL} -signatures Λ, Λ' and any Λ -ontology O for μ , and $\varphi \in \text{Sen}^{\mathcal{EL}}(\Lambda')$:

$$O \models_{\Lambda'}^{\varepsilon} \varphi \quad \text{iff} \quad O \models_{\Lambda \cup \Lambda'}^{\mathcal{EL}} \varphi.$$

The notion of Σ -entailment is basic to many studies of description logics in the literature for instance [15–17, 21, 22]. Closely related notions are *inseparability* and *conservative extension*. Here we present these notions in the framework setting.

Very often we are interested in comparing ontologies written in different languages (frameworks) over a given query basis η . That is, we have two frameworks $\mu_1 : \mathcal{L}_1 \rightarrow \mathcal{G}$ and $\mu_2 : \mathcal{L}_2 \rightarrow \mathcal{G}$; we shall refer to such a situation as a *binary framework*, with notation $\mathfrak{F} = (\mu_1, \mu_2)$.

Definition 2.7 (Σ -entailment and inseparability). For query basis η and frameworks μ_1, μ_2 over η , Λ_1 -ontology O_1 for μ_1 and Λ_2 -ontology O_2 for μ_2 , we say that

- O_1 Σ -entails O_2 wrt η , and write $O_1 \sqsubseteq_{\Sigma}^{\eta} O_2$, iff for all $\varphi \in \text{Sen}^{\mathcal{Q}}(\Sigma)$ we have:

$$O_2 \models_{\Sigma}^{\eta} \varphi \text{ implies } O_1 \models_{\Sigma}^{\eta} \varphi.$$

- O_1 and O_2 are Σ -inseparable wrt η , written $O_1 \approx_{\Sigma}^{\eta} O_2$, iff:

$$O_1 \sqsubseteq_{\Sigma}^{\eta} O_2 \text{ and } O_2 \sqsubseteq_{\Sigma}^{\eta} O_1.$$

- O_2 is a Σ -conservative extension of O_1 wrt η iff $\alpha_{\Lambda_2}^{\mu_2}(O_2) \supseteq \alpha_{\Lambda_1}^{\mu_1}(O_1)$ and O_1 and O_2 are Σ -inseparable wrt η .
- O_2 is a conservative extension of O_1 wrt η iff O_2 is a Σ -conservative extension of O_1 wrt η for all $\Sigma \in |\text{Sig}^{\mathcal{Q}}|$.

For any query basis η and signature Σ in \mathcal{Q} , the relation \approx_{Σ}^{η} wrt η is an equivalence relation.

In the situation of the previous Definition, we say that φ separates O_1 and O_2 iff $O_1 \models_{\Sigma}^{\eta} \varphi$ and $O_2 \not\models_{\Sigma}^{\eta} \varphi$ or vice versa.

Lemma 2.8 states that if we have an inclusion $\Sigma \hookrightarrow \Sigma'$, then \mathfrak{F} -satisfaction for Σ is the same as for Σ' for all Σ -queries (but not for Σ' -queries). In other words extending the \mathcal{Q} -signature with fresh symbols has no impact on the consequence relation in the framework for the queries formulated in the original signature.

Notation 3. In what follows we miss out inclusions, i.e. if $\iota : \Sigma \hookrightarrow \Sigma'$ and $\varphi \in \text{Sen}(\Sigma)$ we write $\alpha_{\Sigma'}^{\eta}(\varphi)$ for $\text{Sen}(\iota)(\alpha_{\Sigma}^{\eta}(\varphi))$.

Lemma 2.8. For a framework $\mu : \mathcal{L} \rightarrow \mathcal{G}$ over query basis $\eta : \mathcal{Q} \rightarrow \mathcal{G}$ and signatures Λ in $\text{Sig}^{\mathcal{L}}$, Σ, Σ' in $\text{Sig}^{\mathcal{Q}}$, s.t. $\Sigma \hookrightarrow \Sigma'$, an ontology $O \subseteq \text{Sen}^{\mathcal{L}}(\Lambda)$, and a query $\varphi \in \text{Sen}^{\mathcal{Q}}(\Sigma)$, the following property holds:

$$O \models_{\Sigma}^{\eta} \varphi \text{ iff } O \models_{\Sigma'}^{\eta} \varphi.$$

This lemma says that if we have a consequence relative to some signature, then we also have that consequence relative to any smaller signature. This property also extends to entailment:

Proposition 2.9. For any binary framework $\mathfrak{F} = (\mu_1, \mu_2)$ over query basis η , and Λ_1 -ontology O_1 for μ_1 , Λ_2 -ontology O_2 for μ_2 , and signatures $\Sigma \hookrightarrow \Sigma'$ in $\text{Sig}^{\mathcal{Q}}$:

$$\text{if } O_1 \sqsubseteq_{\Sigma'}^{\eta} O_2 \text{ then } O_1 \sqsubseteq_{\Sigma}^{\eta} O_2,$$

Note that the opposite direction does not hold because it would imply extending the signature over which queries may be expressed.

Lemma 2.10. For framework μ over query basis $\eta : \mathcal{Q} \rightarrow \mathcal{G}$, if we have a comorphism $\lambda : \mathcal{G} \rightarrow \mathcal{G}'$, there is a framework $\mu' = \mu; \lambda$ over query basis $\eta' = \eta; \lambda$, and we have:

$$O \models_{\Sigma}^{\eta} \varphi \text{ implies } O \models_{\Sigma}^{\eta'} \varphi$$

for any Λ -ontology for μ and any query $\varphi \in \text{Sen}^{\mathcal{Q}}(\Sigma)$, with $\Sigma \in \text{Sig}^{\mathcal{Q}}$. Moreover, if β^{λ} is surjective, then the converse implication also holds.

The following corollary tells us that for any framework $\mu : \mathcal{L} \rightarrow \mathcal{G}$ over a query basis $\eta : \mathcal{Q} \rightarrow \mathcal{G}$ with comorphism $\rho : \mathcal{L} \rightarrow \mathcal{Q}$ and two Λ -ontologies O_1 and O_2 , saying that $O_1 \sqsubseteq_{\Phi^{\rho}(\Lambda)}^{\eta} O_2$ is the same as saying that every formula $\varphi \in O_2$ translated along ρ is a consequence of O_1 wrt η .

Corollary 2.11. Given a framework $\mathcal{L} \xrightarrow{\mu} \mathcal{G}$ over a query basis $\mathcal{Q} \xrightarrow{\eta} \mathcal{G}$ with a comorphism $\mathcal{L} \xrightarrow{\rho} \mathcal{Q}$, s.t. $\mu = \rho; \eta$, and Λ -ontologies O_1 and O_2 , we have: $O_1 \sqsubseteq_{\Phi^{\rho}(\Lambda)}^{\eta} O_2$ iff $O_1 \models_{\Phi^{\rho}(\Lambda)}^{\eta} \varphi$, for all $\varphi \in \alpha_{\Lambda}^{\rho}(O_2)$.

3. Robustness Properties and the Craig Interpolation Property

In this section we consider how ontologies may be combined. If we have frameworks μ_1 and μ_2 over the same query basis η , then a Λ_1 -ontology O_1 for μ_1 can be combined with a Λ_2 -ontology O_2 for μ_2 by taking the union of $\alpha^{\mu_1}(O_1)$ and $\alpha^{\mu_2}(O_2)$ in \mathcal{G} , even if the ontology languages of μ_1 and μ_2 are different. This justifies using a notation for union for ontologies, so we can write, for example, $O_1 \cup O_2 \sqsubseteq_{\Sigma}^{\eta} O_1$.

In the previous section, we studied some properties of \approx_{Σ}^{η} , but whenever we are using Σ -inseparability we are interested in determining what robustness properties it has, i.e., how safely ontologies can be combined. We introduce three types of robustness properties in the framework setting.

Definition 3.1. For any binary framework $\mathfrak{F} = (\mu_1, \mu_2)$ over query basis η we say that \mathfrak{F} is *robust under*:

- *vocabulary extension* if for all signatures Λ_1 in $\text{Sig}^{\mathcal{L}^1}$, Λ_2 in $\text{Sig}^{\mathcal{L}^2}$, Σ, Σ' in $\text{Sig}^{\mathcal{Q}}$, s.t. $\Sigma \hookrightarrow \Sigma'$ and $\Phi^{\eta}(\Sigma') \cap (\Phi^{\mu_1}(\Lambda_1) \cup \Phi^{\mu_2}(\Lambda_2)) \hookrightarrow \Phi^{\eta}(\Sigma)$, all ontologies $O_1 \subseteq \text{Sen}^{\mathcal{L}^1}(\Lambda_1)$ and $O_2 \subseteq \text{Sen}^{\mathcal{L}^2}(\Lambda_2)$ the following holds:

$$O_1 \sqsubseteq_{\Sigma}^{\eta} O_2 \text{ implies } O_1 \sqsubseteq_{\Sigma'}^{\eta} O_2,$$

- *joins* if for all signatures Λ_1 in $\text{Sig}^{\mathcal{L}^1}$, Λ_2 in $\text{Sig}^{\mathcal{L}^2}$ and Σ in $\text{Sig}^{\mathcal{Q}}$, s.t. $\Phi^{\mu_1}(\Lambda_1) \cap \Phi^{\mu_2}(\Lambda_2) \hookrightarrow \Phi^{\eta}(\Sigma)$ and all ontologies $O_1 \subseteq \text{Sen}^{\mathcal{L}^1}(\Lambda_1)$ and $O_2 \subseteq \text{Sen}^{\mathcal{L}^2}(\Lambda_2)$, the following holds for $i = 1, 2$:

$$O_1 \approx_{\Sigma}^{\eta} O_2 \text{ implies } O_i \approx_{\Sigma}^{\eta} O_1 \cup O_2.$$

- *replacement in framework* $\mu : \mathcal{L} \rightarrow \mathcal{G}$ if for all signatures Λ_1 in $\text{Sig}^{\mathcal{L}^1}$, Λ_2 in $\text{Sig}^{\mathcal{L}^2}$, Λ in $\text{Sig}^{\mathcal{L}}$ and Σ in $\text{Sig}^{\mathcal{Q}}$, s.t. $\Phi^\mu(\Lambda) \cap (\Phi^{\mu_1}(\Lambda_1) \cup \Phi^{\mu_2}(\Lambda_2)) \hookrightarrow \Phi^\eta(\Sigma)$, for all ontologies $O_1 \subseteq \text{Sen}^{\mathcal{L}^1}(\Lambda_1)$, $O_2 \subseteq \text{Sen}^{\mathcal{L}^2}(\Lambda_2)$, $O \subseteq \text{Sen}^{\mathcal{L}}(\Lambda)$, the following holds:

$$O_1 \sqsubseteq_{\Sigma}^{\eta} O_2 \text{ implies } O_1 \cup O \sqsubseteq_{\Sigma}^{\eta} O_2 \cup O.$$

We briefly discuss the intuitions behind those three types of robustness.

Robustness under vocabulary extension assures us that we can extend signature Σ with fresh symbols, which do not occur in O_1 nor O_2 and this extension has no impact on Σ -inseparability of O_1 and O_2 . If the notion of inseparability does not have this type of property then it is potentially difficult to use. It is so because we may be unable to interpret the meaning of $O_1 \approx_{\Sigma}^{\eta} O_2$ because we could have $O_1 \not\approx_{\Sigma \cup \Sigma'}^{\eta} O_2$, for Σ' containing only symbols that are not used in O_1 nor O_2 .

Robustness under joins. We will use typical example to present the importance of this type of robustness. First we introduce a proposition which is a consequence of robustness under joins:

Proposition 3.2. *For any frameworks μ, μ_1, μ_2 over η which are pair-wise robust under joins, signatures Λ_1 in $\text{Sig}^{\mathcal{L}^1}$, Λ_2 in $\text{Sig}^{\mathcal{L}^2}$ and Λ in $\text{Sig}^{\mathcal{L}}$, s.t. $\Phi^{\mu_1}(\Lambda_1) \cap \Phi^{\mu_2}(\Lambda_2) \hookrightarrow \Phi^{\mu}(\Lambda)$, and ontologies $O \subseteq \text{Sen}^{\mathcal{L}}(\Lambda)$, $O_1 \subseteq \text{Sen}^{\mathcal{L}^1}(\Lambda_1)$, $O_2 \subseteq \text{Sen}^{\mathcal{L}^2}(\Lambda_2)$, if $\alpha_{\Lambda}^{\mu}(O) \cup \alpha_{\Lambda_i}^{\mu_i}(O_i)$ is a conservative extension of $\alpha_{\Lambda}^{\mu}(O)$, then also $\alpha_{\Lambda}^{\mu}(O) \cup \alpha_{\Lambda_1}^{\mu_1}(O_1) \cup \alpha_{\Lambda_2}^{\mu_2}(O_2)$ is a conservative extension of $\alpha_{\Lambda}^{\mu}(O)$.*

For an illustration let us suppose that two groups of ontology designers independently refine an ontology $O \subseteq \text{Sen}(\Lambda)$ by creating their own set of axioms, say $O_1 \subseteq \text{Sen}(\Lambda_1)$ and $O_2 \subseteq \text{Sen}(\Lambda_2)$ respectively, where $\Lambda, \Lambda_1, \Lambda_2 \in |\text{Sig}|$. Both teams ensure that their ontologies are conservative extensions of O . Assume that these groups decide to merge their ontologies at some point, they would like $O \cup O_1 \cup O_2$ to be a conservative extension of O , but the fact that O_1 and O_2 are conservative extensions of O is not sufficient to guarantee that. The solution for that is to make sure that $\Lambda, \Lambda_1, \Lambda_2 \in |\text{Sig}|$, s.t. $\Lambda_1 \cap \Lambda_2 \hookrightarrow \Lambda$. Then by Proposition 3.2 we get that $O \cup O_1 \cup O_2$ is a conservative extension of O .

Robustness under replacement. This type of robustness is very important when we want to reuse some modules of existing ontologies in new applications. For instance, assume that a group of ontology designers is developing an ontology O for some medical institution. Part of this ontology is supposed to use terminology over Σ . They know that there is another ontology O' , which already defines symbols of Σ , so instead creating this part of O from scratch they would prefer to reuse O' in order to get descriptions of Σ . But additionally they want to avoid importing whole O' , therefore they decide to import a Σ -module O_{Σ} of O' . Now, if we know that the framework \mathfrak{F} used for answering the queries is robust under replacement then from $O_{\Sigma} \approx_{\Sigma}^{\eta} O'$ follows that $O \cup O_{\Sigma} \approx_{\Sigma}^{\eta} O \cup O'$. Thanks to that ontology designers know that indeed they can import only O_{Σ} instead O' and still get the same set of consequences.

We present the institution-independent formulation of Craig interpolation property introduced in [7]. This formulation is capable of capturing any square

of signature morphisms, whereas previous formulations considered only squares with intersection and union of signatures.

Definition 3.3 (Craig interpolation square). A commutative square of signature morphisms

$$\begin{array}{ccc} \Sigma & \xrightarrow{\sigma_1} & \Sigma_1 \\ \sigma_2 \downarrow & & \downarrow \sigma'_1 \\ \Sigma_2 & \xrightarrow{\sigma'_2} & \Sigma' \end{array}$$

is a *Craig interpolation square* if and only if for every set E_1 of Σ_1 -sentences and set E_2 of Σ_2 -sentences such that $\text{Sen}(\sigma'_1)(E_1) \models_{\Sigma'} \text{Sen}(\sigma'_2)(E_2)$, there exists a set E of Σ -sentences such that $E_1 \models_{\Sigma_1} \text{Sen}(\sigma_1)(E)$ and $\text{Sen}(\sigma_2)(E) \models_{\Sigma_2} E_2$. The set E is called an interpolant of E_1 and E_2 . (A special case of interpolation is a situation with $\Sigma = \Sigma_1 \cap \Sigma_2$ and $\Sigma' = \Sigma_1 \cup \Sigma_2$.)

An institution \mathcal{I} has the *weak Craig interpolation property* (or simply weak interpolation) iff every commutative square of signature morphisms is a Craig interpolation square.

An institution \mathcal{I} has the *Craig interpolation property* iff it has weak interpolation and there always exists an interpolant which is finite.

Now we study correlations of different types of robustness and interpolation. The following proposition shows that weak interpolation in \mathcal{G} implies robustness under vocabulary extension of η .

Proposition 3.4. *Let $\mathfrak{F} = (\mu_1, \mu_2)$ be a binary framework over query basis $\eta : \mathcal{Q} \rightarrow \mathcal{G}$, with η conservative and such that for every signature Σ in $\text{Sig}^{\mathcal{Q}}$, β_{Σ}^{η} is surjective. Moreover, let there be a comorphism $\rho_2 : \mathcal{L}_2 \rightarrow \mathcal{Q}$ such that $\mu_2 = \rho_2; \eta$. Then, if \mathcal{G} has weak interpolation then \mathfrak{F} is robust under vocabulary extension.*

The following proposition can be understood as a partial converse of Proposition 3.4.

Proposition 3.5. *For any framework $1_{\mathcal{G}} : \mathcal{G} \rightarrow \mathcal{G}$ over query basis $1_{\mathcal{G}}$ robust under vocabulary extension for possibly infinite ontologies we have that \mathcal{G} has weak interpolation.*

Definition 3.6. An institution \mathcal{I} is closed under Boolean operators if for every $\varphi, \psi \in \text{Sen}^{\mathcal{I}}(\Sigma)$ there are Σ sentences $\neg\varphi$ and $\varphi \wedge \psi$ such that for every Σ -model \mathcal{M} , the following holds:

$$\begin{aligned} \mathcal{M} \models_{\Sigma} \varphi \text{ and } \mathcal{M} \models_{\Sigma} \psi & \text{ iff } \mathcal{M} \models_{\Sigma} \varphi \wedge \psi \\ \mathcal{M} \models_{\Sigma} \neg\varphi & \text{ iff } \mathcal{M} \not\models_{\Sigma} \varphi. \end{aligned}$$

The following proposition shows a correlation between robustness under joins of \mathfrak{F} and interpolation.

Proposition 3.7. *For any binary framework $\mathfrak{F} = (\mu_1, \mu_2)$ over query basis η , s.t. there is a comorphism $\mathcal{L}_2 \xrightarrow{\rho_2} \mathcal{Q}$ and \mathcal{Q} is closed under Boolean operators, if \mathcal{Q} has weak interpolation, then \mathfrak{F} is robust under joins.*

Theorem 3.8. *Any binary framework $\mathfrak{F} = (\mu_1, \mu_2)$ over query basis $1 : FOL \rightarrow FOL$ is robust under vocabulary extension, joins and it has robustness under replacement in any framework μ over the query basis 1_{FOL} .*

The following proposition shows a correlation between interpolation and robustness under joins for any framework $\mu : \mathcal{Q} \rightarrow \mathcal{G}$ over query basis $\mu : \mathcal{Q} \rightarrow \mathcal{G}$, where \mathcal{Q} is an institution of any fragment of first-order logic closed under Boolean operators.

Proposition 3.9. *Let \mathcal{Q} be an institution of any fragment of first-order logic closed under Boolean operators, let $\mu : \mathcal{Q} \rightarrow \mathcal{G}$ be a framework over query basis $\mu : \mathcal{Q} \rightarrow \mathcal{G}$, such that μ is robust under joins for possibly infinite ontologies. Then \mathcal{Q} has interpolation.*

Corollary 3.10. *Let \mathcal{Q} be an institution of any fragment of first-order logic closed under Boolean operators, let $\mu : \mathcal{Q} \rightarrow FOL$ be a framework over query basis $\mu : \mathcal{Q} \rightarrow FOL$, such that μ is robust under joins for possibly infinite ontologies. Then \mathcal{Q} has interpolation.*

4. Conclusions

We have proposed a functional approach to ontologies that focusses on ontologies as providing answers to queries. Using institutions and institution comorphisms allows us to address heterogeneous ontologies in a language-independent way. Our notion of framework captures the situation where queries may be posed to ontologies written in different formalisms, and allows us to generalise the robustness results of [15].

Our future work will focus on using frameworks for further investigation of properties of institutions of description logics. In particular we would like to turn our attention to description logics with individuals in their signatures. We want to investigate how given an institution \mathcal{I} can we generate its counterpart institution \mathcal{I}' with individuals in its signature and what are the relations between these two (morphisms and comorphisms). This allows us to consider the problem of querying the instance data, i.e. posting queries to ontology together with an ABox that stores instances of classes and relations.

References

- [1] M. Aiguier and F. Barbier. An Institution-independent Proof of the Beth Definability Theorem. *Studia Logica*, 85(3):333–359, 2007.
- [2] F. Baader, S. Brandt, and C. Lutz. Pushing the \mathcal{EL} envelope. In L. P. Kaelbling and A. Saffiotti, editors, *Proceedings of the Nineteenth International Joint Conference on Artificial Intelligence (IJCAI'05)*, pages 364–369. Professional Book Center, 2005.
- [3] F. Baader, D. Calvanese, D. L. McGuinness, D. Nardi, and P. F. Patel-Schneider, editors. *The Description Logic Handbook: Theory, Implementation, and Applications*. Cambridge University Press, 2003.
- [4] W. N. Borst. *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. PhD thesis, Universiteit Twente, Enschede, September 1997.

- [5] D. Calvanese, D. Lembo, M. Lenzerini, and R. Rosati. DL-lite: Tractable description logics for ontologies. In *Proc. of the 20th Nat. Conf. on Artificial Intelligence (AAAI 2005)*, pages 602–607, 2005.
- [6] M. Codescu and T. Mossakowski. Heterogeneous colimits. In *ICSTW '08: Proceedings of the 2008 IEEE International Conference on Software Testing Verification and Validation Workshop*, pages 131–140, Washington, DC, USA, 2008. IEEE Computer Society.
- [7] R. Diaconescu. An Institution-independent Proof of Craig Interpolation Theorem. *Studia Logica*, 77(1):59–79, 2004.
- [8] R. Diaconescu. *Institution-independent Model Theory*. Birkhäuser Basel, 2008.
- [9] J. Euzenat and P. Shvaiko. *Ontology matching*. Springer-Verlag, Heidelberg (DE), 2007.
- [10] J. Goguen and R. Burstall. Introducing institutions. In *Proceedings, Logics of Programming Workshop*, volume 164, pages 221–256. Springer, 1984.
- [11] J. A. Goguen. What is a concept? In F. Dau, M.-L. Mugnier, and G. Stumme, editors, *ICCS*, volume 3596 of *Lecture Notes in Computer Science*, pages 52–77. Springer, 2005.
- [12] J. A. Goguen and R. M. Burstall. Institutions: abstract model theory for specification and programming. *J. ACM*, 39(1):95–146, 1992.
- [13] J. A. Goguen and G. Roşu. Institution morphisms. *Formal Asp. Comput.*, 13(3-5):274–307, 2002.
- [14] T. R. Gruber. A Translation Approach to Portable Ontology Specifications. *Knowl. Acquis.*, 5(2):199–220, June 1993.
- [15] B. Konev, C. Lutz, D. Walther, and F. Wolter. Formal properties of modularisation. In C. Parent, S. Spaccapietra, and H. Stuckenschmidt, editors, *Ontology Modularisation*, LNCS. Springer, 2008.
- [16] B. Konev, D. Walther, and F. Wolter. The logical difference problem for description logic terminologies. In *IJCAR '08: Proceedings of the 4th international joint conference on Automated Reasoning*, pages 259–274. Springer, 2008.
- [17] R. Kontchakov, F. Wolter, and M. Zakharyashev. Can you tell the difference between DL-lite ontologies? In *KR*, pages 285–295, 2008.
- [18] O. Kutz, D. Lücke, and T. Mossakowski. Heterogeneously structured ontologies - integration, connection, and refinement. In T. Meyer and M. A. Orgun, editors, *Knowledge Representation Ontology Workshop (KROW 2008)*, volume 90 of *CRPIT*, pages 41–50, Sydney, Australia, 2008. ACS.
- [19] O. Kutz and T. Mossakowski. Modules in transition - conservativity, composition, and colimits. In B. C. Grau, V. Honavar, A. Schlicht, and F. Wolter, editors, *WoMO*, volume 315 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2007.
- [20] O. Kutz and T. Mossakowski. Conservativity in structured ontologies. In M. Ghallab, C. D. Spyropoulos, N. Fakotakis, and N. M. Avouris, editors, *ECAI*, volume 178 of *Frontiers in Artificial Intelligence and Applications*, pages 89–93. IOS Press, 2008.
- [21] C. Lutz, D. Walther, and F. Wolter. Conservative extensions in expressive description logics. In M. Veloso, editor, *Proceedings of the Twentieth International Joint Conference on Artificial Intelligence (IJCAI'07)*, pages 453–458. AAAI Press, 2007.
- [22] C. Lutz and F. Wolter. Conservative extensions in the lightweight description logic \mathcal{EL} . In F. Pfenning, editor, *Proceedings of the 21th Conference on Automated Deduction (CADE-21)*, Lecture Notes in Artificial Intelligence, pages 84–99. Springer, 2007.
- [23] N. F. Noy and M. Musen. Promptdiff: A fixed-point algorithm for comparing ontology versions. In *Proceedings of AAAI*, pages 744–750, 2002.
- [24] M. Schorlemmer and Y. Kalfoglou. Institutionalising ontology-based semantic integration. *Appl. Ontol.*, 3(3):131–150, 2008.
- [25] R. Studer, V. R. Benjamins, and D. Fensel. Knowledge engineering: Principles and methods. *Data Knowl. Eng.*, 25(1-2):161–197, 1998.
- [26] M. Uschold and M. Grüninger. Ontologies: Principles, methods and applications. *Knowledge Engineering Review*, 11:93–136, 1996.
- [27] A. Zimmermann, M. Krötzsch, J. Euzenat, and P. Hitzler. Formalizing ontology alignment and its operations with category theory. In *Proceeding of the 2006 conference on Formal Ontology in Information Systems*, pages 277–288, Amsterdam, The Netherlands, The Netherlands, 2006. IOS Press.