Detecting Influences of Ontology Design Patterns in Biomedical Ontologies

Christian Kindermann(✉), Bijan Parsia, and Uli Sattler

University of Manchester, Manchester, UK
{christian.kindermann,bijan.parsia,uli.sattler}@manchester.ac.uk

Abstract. Ontology Design Patterns (ODP) have been proposed to facilitate ontology engineering. Despite numerous conceptual contributions for over more than a decade, there is little empirical work to support the often claimed benefits provided by ODPs. Determining ODP use from ontologies alone (without interviews or other supporting documentation) is challenging as there is no standard (or required) mechanism for stipulating the intended use of an ODP. Instead, we must rely on modelling features which are suggestive of a given ODP's influence. For the purpose of determining the prevalence of ODPs in ontologies, we developed a variety of techniques to detect these features with varying degrees of liberality. Using these techniques, we survey BioPortal with respect to well-known and publicly available repositories for ODPs. Our findings are predominantly negative. For the vast majority of ODPs we cannot find empirical evidence for their use in biomedical ontologies.

1 Introduction

The idea of Ontology Design Patterns (ODP) has been introduced as a means to facilitate ontology engineering [3,6]. Generally thought of as best practices and well-proven modelling solutions, a variety of different kinds of ODPs exist [1,3,7,26]. Despite conceptual contributions for more than a decade, there is very little empirical work to provide support for these claims. Ways of determining the prevalence of ODPs in practice is a first step into this direction. However, recognising ODP use from ontologies alone (without interviews or other supporting documentation) is challenging as there is no standard (or required) mechanism for stipulating the intended use of an ODP. In this paper, we take on this challenge and develop algorithmic techniques to automate the identification of a given ODP's influence.

The contributions are as follows: (i) we develop a variety of techniques to detect modelling features that are suggestive for a given ODP's influence, (ii) we characterise these techniques and discuss their informative value, (iii) and we perform an empirical study using these techniques to investigate the prevalence of ODPs in biomedical ontologies.

C. Ghidini et al. (Eds.): ISWC 2019, LNCS 11778, pp. 311–328, 2019.
https://doi.org/10.1007/978-3-030-30793-6_18

2 Background on Ontology Design Patterns

Different frameworks for working with patterns in Ontology Engineering have been proposed [3,4,6,12,20,22,25,28,29]. Each framework is based on a different approach for capturing the benefits of patterns and introduces its own terminology as well as its own notation. While these different approaches bear similarities to each other in some respects, there have been no efforts towards a standardisation process. Neither is there a generally accepted de facto standard for working with patterns in practice.

A unifying concept for a majority of frameworks for ODPs is a practical notion *pattern reuse*. Such notions often involve prefabricated components expressed in some representation formalism on the one hand, and operations to manipulate these components on the other.

Consider the following examples in which a pattern has been proposed to be reused as

- "[...] a first-order theory whose axioms are not part of the target knowledge base, but can be incorporated via renaming of their non-logical symbols [4]."
- "[*a*] distinguished ontolog[y]." The basic mechanism for its application is OWL ontology import in which pattern elements cannot be modified. Otherwise, common operations for patterns are "clone, specialisation, generalisation, composition, expansion" [20].
- "[...] an ontology fragment, including directly reusable elements (classes, properties, etc.) as well as demo-elements that would be replaced by the user's own. The directly reusable elements should typically be borrowed from upper level ontologies [28]."

Clearly, these ideas of pattern reuse are based on a set of predefined axioms that may or may not be modified. In the scope of this work, we will restrict our attention to ODPs of this kind, i.e., ODPs that are captured by a set of axioms or an OWL ontology. Such ODPs have been the focus of the academic literature for over a decade and are commonly classified into two types. One type addresses domain specific modelling problems, whereas the other is concerned with language specific modelling techniques. The former are generally discussed under the name of *Content Ontology Design Patterns* (CODP) and latter under the name of *Logical Ontology Design Patterns* (LODP).

CODPs are motivated as conceptual modelling solutions featuring a domain dependent signature, possibly extracted from Upper Level Ontologies to be applicable across different domains [20]. LODPs on the other hand are motivated as structural components that are domain-independent [6,21]. As a consequence, the former are characterised by a fixed set of unmodifiable axioms whereas the latter are characterised by a set of axioms containing variables.[1]

[1] These characterisations are not as clear-cut as they might appear. The discussion on the submission for the ODP `ContextSlices` http://ontologydesignpatterns.org/wiki/Submissions:Context_Slices exemplifies differences of opinion on the matter in the research community.

3 Pattern Detection

The lack of a generally agreed upon notion for ODP reuse poses a challenge for determining whether an ODP has in fact informed the design of a given ontology. Different approaches for ODP reuse result in different modelling features suggestive for a given ODP's influence. Therefore, we must design a detection mechanism that accounts for this uncertainty.

In the scope of this work, we limit our investigation to approaches that are based on ODPs documented with reusable components (cf. Sect. 2). Furthermore, we assume these components to be given in the form of ontologies or more generally sets of axioms. Given such a component $\mathcal{P}$, the problem of detecting modelling features which are suggestive of the ODP's influence in a given ontology $\mathcal{O}$ can be reduced to detecting features of $\mathcal{P}$ shared with $\mathcal{O}$. In the following, we formulate a list of non-exhaustive criteria that may be used to determine shared features between $\mathcal{P}$ and $\mathcal{O}$.

3.1 Detection Techniques

One of the earliest approaches for reusing an ODP's $\mathcal{P}$ proposed ontology import as the basic mechanism for reuse [20]. This approach has been adopted by the NeOn project[2] [21] and the large amount of work carried out in the context of this project has promulgated into the academic literature.

Import Containment. Detecting whether a given $\mathcal{P}$ of some ODP has been imported in an ontology $\mathcal{O}$ comes down to a straightforward analysis of $\mathcal{O}$'s import declaration. Given our primary concern of detecting an ODP's influence without any further qualification, we will generally equate an ontology with its import closure unless stated otherwise.

The analysis of $\mathcal{O}$'s import declarations is based on the two ways an ontology may be imported. Namely, *import by name* and *import by location*. Import by name is performed by interpreting the object of an import declaration as the name of an ontology in a predefined list of ontology repositories. If the object of an import declaration can be matched with the name of an ontology in said repositories, then the ontology is imported. Contrary, import by location is performed by interpreting the object of an import declaration as a physical location of an ontology. This location may be a location in the local file system.

Import by name allows for an unambiguous way to determine whether a given $\mathcal{P}$ has been imported, if its name in some ontology repository is known. Import by location on the other hand, poses a challenge due to the possibility of arbitrary renaming of local files. Nevertheless, it is reasonable to assume that the name of a local file is suggestive of its contents and to consider lexically similar import declarations as candidates for $\mathcal{P}$ reuse.

These consideration motivate a twofold detection procedure. First, check whether $\mathcal{P}$ is imported by using its known URL from a pattern repository as a

[2] http://neon-project.org/nw/Welcome_to_the_NeOn_Project.html.

name. If $\mathcal{P}$ is not found, test the import declarations in an ontology for lexically similar names to the one of $\mathcal{P}$. We refer to the former as the `ImportByURLCheck` and the latter as the `ImportByLocation` check.

Signature Overlap. It has been proposed to reuse a given $\mathcal{P}$ by copying its contents into a target $\mathcal{O}$ [24]. Copying any logical entities in $\mathcal{P}$ verbatim will result in syntactic traces, i.e, $\widetilde{\mathcal{P}} \cap \widetilde{\mathcal{O}} \neq \emptyset$, where $\widetilde{\mathcal{O}}$ denotes the signature of an ontology, i.e., its class, property, and individual names. Hence, we specify an `IRICheck` that tests for all logical axioms $\alpha \in \mathcal{P}$ whether the IRI of any entity name $e \in \widetilde{\alpha}$ occurs in $\mathcal{O}$. This occurrence test in $\mathcal{O}$ encompasses all of $\mathcal{O}$'s logical axioms as well as its non-logical components such as annotations and entity declarations.

In addition, we specify a `NamespaceCheck` that tests whether the object of a namespace declaration[3] in $\mathcal{P}$ can be matched within some IRI of entities in $\mathcal{O}$.

Lexical Variation. In addition to approaches preserving the IRIs of elements in $\mathcal{P}$ under reuse in $\mathcal{O}$, there are proposals allowing for the possibility of a renaming for copied elements [10]. In this case, the reuse of axioms $\alpha \in \mathcal{P}$ can be identified by some substitution[4] $\sigma : \widetilde{\mathcal{P}} \to \widetilde{\mathcal{O}}$ such that $\sigma(\alpha) \in \mathcal{O}$. However, with no information expressly declaring that $\mathcal{P}$ has been reused via some σ in $\mathcal{O}$, determining whether $\mathcal{P}$ has in fact been reused under some elusive substitution is a challenging task.

Based on the assumption that entities $p \in \widetilde{\alpha}$ ($\alpha \in \mathcal{P}$ being a logical axiom) exhibit lexical similarities to entities $\sigma(p) \in \widetilde{\mathcal{O}}$, we can attempt to generate candidate substitutions. Comparing an entity $p \in \widetilde{\mathcal{P}}$ with all entities $o \in \widetilde{\mathcal{O}}$ in terms of their lexical similarity, we can associate p with a set of possible renamings $R_p = \{r_1, \ldots, r_n\} \subseteq \widetilde{\mathcal{O}}$. Doing so for all entities $p_1, \ldots, p_n$ in $\mathcal{P}$'s signature results in a corresponding number of sets $R_{p_1}, \ldots, R_{p_n}$. Candidate substitutions σ are then generated by

$$\{\sigma \mid \sigma(p_i) \mapsto \pi_i(\boldsymbol{e}), \quad \boldsymbol{e} \in R_{p_1} \times \ldots \times R_{p_n}\},$$

where π_i is a projection map such that $\pi_i(\boldsymbol{e}) = e_i$ for $\boldsymbol{e} = (e_1, \ldots, e_n)$.

If $R_{p_1} \times \ldots \times R_{p_n}$ is non-empty, then there exists a candidate substitution σ. In that case, we specify a `SubstitutionContainmentCheck` that tests whether $\sigma(\alpha) \in \mathcal{O}$ holds for all $\alpha \in \mathcal{P}$ under some σ.

Logical Variation. Besides changing the signature of an ODP's $\mathcal{P}$, there have been proposals for ODP reuse based on reimplementing aspects of $\mathcal{P}$ by analogy [5]. In this case, both the logical structure as well as the signature

[3] https://www.w3.org/TR/2004/REC-owl-guide-20040210/#Namespaces.

[4] Substitutions are assumed to respect types, i.e., classes, properties, and individuals are only mapped to other classes, properties, and individuals respectively.

of axioms $\alpha \in \mathcal{P}$ may be subject to change. Based on motivations for logical rewritings of $\mathcal{P}$ [11], we specify a `SubstitutionEntailmentCheck` that tests whether there exists some substitution σ (generated as previously for the `SubstitutionContainmentCheck`) such that for all $\alpha \in \mathcal{P}$ it holds that $\mathcal{O} \models \sigma(\alpha)$.

Structural Axiom Agreement. In addition to detection techniques searching for positive evidence that is suggestive of a given ODP's $\mathcal{P}$, it is possible to test an ontology $\mathcal{O}$ for necessary structural conditions imposed by some notion of $\mathcal{P}$'s reuse. For example, positive evidence for $\mathcal{P}$ under the `SubstitutionContainmentCheck` requires an ontology $\mathcal{O}$ to contain structurally identical axioms to $\mathcal{P}$ since a simple renaming of entities in $\mathcal{P}$ does not affect the logical structure of axioms in $\mathcal{P}$. Therefore, if an ontology does not contain at least as many axioms of a given type as $\mathcal{P}$, then certain ways of $\mathcal{P}$'s reuse can be ruled out. Namely, any notion of ODP reuse that requires the explicit reuse of all axioms in $\mathcal{P}$.

Hence, we specify a structural `AxiomTypeCheck`, that tests whether $\mathcal{O}$ contains at least as many axioms of a given type[5] as $\mathcal{P}$.

Structural Expression Agreement. Orthogonal to a structural agreement in terms of axioms, we can specify structural expression checks that test whether some logical constructs or combination of logical constructs proposed by a given ODP's $\mathcal{P}$ occur in an ontology. For example, suppose a logical constructor, e.g. class union, occurs in some expression used in $\mathcal{P}$. If there is no such expression in a target ontology (as is often the case for biomedical ontologies conforming to the EL profile), then certain ways of reusing $\mathcal{P}$ can be ruled out.

In the context of this work, we specify expression checks for two logical structures that seem to be crucial for a fair number of ODPs. These structures are described by two LODPs, namely "Partition"[6] and "Nary-Relation"[7]. The former is characterised by a disjoint union of classes, whereas the latter is characterised by a class that is subsumed by at least two OWL restrictions. Accordingly, we define a `DisjointUnionCheck` that searches for the presence of disjoint unions as specified by the OWL Language Specification.[8] And furthermore, we define a `NaryRelationCheck` that searches for the presence of any class that is subsumed by at least two OWL restrictions.[9]

5 The types of axioms we consider in this study are all subclasses of the OWLAxiom interface http://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/model/OWLAxiom.html of the well-known OWL API.

6 http://odps.sourceforge.net/odp/html/Value_Partition.html.

7 http://odps.sourceforge.net/odp/html/Nary_Relationship.html.

8 https://www.w3.org/TR/owl2-syntax/#Disjoint_Union_of_Class_Expressions.

9 http://owlcs.github.io/owlapi/apidocs_5/org/semanticweb/owlapi/model/OWLRestriction.

3.2 Characterisation of Detection Techniques

The detection techniques presented above all target some features of a given ODP's $\mathcal{P}$ which are deemed to be suggestive for an ODP's influence. The characteristics of these features allow us to qualify what kind of ODP reuse the respective detection technique is capable of identifying. For example, the `IRICheck` selects $\mathcal{P}$'s signature as a target feature of $\mathcal{P}$ for its detection. By doing so, the `IRICheck` is capable of detecting any notion of ODP reuse that preserves some element of $\mathcal{P}$'s signature. In the table below, we associate each detection technique (that searches for positive evidence of a given ODP's influence) with a corresponding notion of ODP reuse.[10] In the second column, we describe the potential kind of influence $\mathcal{I}$ of an ODP's $\mathcal{P}$ in a given ontology $\mathcal{O}$ and in the third column, we describe the relationship between $\mathcal{I}$ (occurring in $\mathcal{O}$) and $\mathcal{P}$. The influence $\mathcal{I}$ can manifest in several different forms, e.g. axioms, entities, annotations, etc. (Table 1).

Table 1. Association between detection techniques and notions of ODP reuse

Detection technique	Influence $\mathcal{I}$ in $\mathcal{O}$	Relation between $\mathcal{I}$ and $\mathcal{P}$	Notion of reuse
`ImportByURL`	$\mathcal{O}$ imports $\mathcal{I}$	$\mathcal{I} = \mathcal{P}$	Import
`ImportByLocation`	$\mathcal{O}$ imports $\mathcal{I}$	$\mathcal{I} = \mathcal{P}$	Import
`IRICheck`	$\widetilde{\mathcal{I}} \subseteq \widetilde{\mathcal{O}}$	$\widetilde{\mathcal{I}} \cap \widetilde{\mathcal{P}} \neq \emptyset$	Signature
`NamespaceCheck`	$\mathcal{I}$ occurs in $\mathcal{O}$	$\mathcal{I}$ points to $\mathcal{P}$	Reference
`SContainmentCheck`	$\mathcal{I} \subseteq \mathcal{O}$	$\sigma(\mathcal{I}) = \mathcal{P}$	Renaming
`EContainmentCheck`	$\mathcal{O} \models \mathcal{I}$	$\sigma(\mathcal{I}) = \mathcal{P}$	Rewriting

Furthermore, we can qualify the detectable notions of ODP reuse according to a number of characteristics. For example, the `SubstitutionContainmentCheck` targets notions of ODP reuse that allow for some form of *lexical* variation. However, it cannot detect influences of notions of ODP reuse that allow for *logical* variations, e.g, logically equivalent rewritings. Neither can it detect any notion of *partial* reuse that possibly omits some semantically relevant aspect of a given ODP. This is due to the requirement of *all* axioms $\alpha \in \mathcal{P}$ to be explicitly contained in $\mathcal{O}$ under some substitution (renaming) σ of entities $e \in \widetilde{\alpha}$ (cf. Sect. 3.1). Contrary, the `IRICheck` is able to detect influences of notions of partial ODP reuse that allow for logical variation. It only requires the preservation of some element of a given ODP's signature $\widetilde{\mathcal{P}}$.

In Table 2, we summarise characteristics of notions for ODP reuse that our detection techniques capture. We indicate for each notion of reuse whether variations of lexical or logical features are taken into account and whether a partial or complete reuse of a given ODP is assumed.

[10] The `SubstitutionContainmentCheck` has been abbreviated by `SContainmentCheck` for presentation purposes.

Table 2. Characterisation of detectable notions of ODP reuse

Detection technique	Notion of reuse	Feature variation	Reuse format
`ImportByURL`	Import	-	Complete
`ImportByLocation`	Import	-	Complete
`IRICheck`	Signature	Logical	Partial
`NamespaceCheck`	Reference	Lexical & Logical	Partial
`SContainmentCheck`	Renaming	Lexical	Complete
`EContainmentCheck`	Rewriting	Lexical & Logical	Complete

In addition to the detection techniques that aim to identify concrete positive evidence of a given ODP's influence, we have motivated detection techniques that can provide negative evidence for an ODP's reuse. Such negative evidence is established by the *absence* of distinguished features of a given ODP's $\mathcal{P}$. Accordingly, we can associate such detection techniques with features of $\mathcal{P}$ that are necessarily preserved under certain notions of reuse. On the one hand, there is the `AxiomTypeCheck` that is generally applicable for any ODP under any notion of reuse that preserves the logical structure of the pattern's corresponding $\mathcal{P}$. On the other hand, there are more specialised detection techniques that are tailored towards ODPs containing distinguished structural components, i.e. the `DisjointUnionCheck` and the `NAryRelationCheck`.

3.3 Algorithm

Most techniques introduced in the previous section involve some form of string comparison between entities of $\mathcal{O}$ and $\mathcal{P}$. In order to maximise the recall of lexical detection techniques, we employ a threefold string matching procedure – each step increasing the degree of liberality in terms of lexical similarity between two strings s_1 and s_2.

The first part is a strict equality that requires all symbols occurring in s_1 to coincide with symbols in s_2 at their respective positions. The second part is an approximate string match between s_1 and s_2. Here, all symbols not in the Latin alphabet are removed from both s_1 and s_2 and the remaining characters are converted to lower case. Then, a test for string containment of s_1 in s_2 is performed. The third part consists of calculating a string similarity greater that 0.8 based on the Levensthein distance.[11]

A lexical association between two elements $e_1 \in \widetilde{\mathcal{P}}$ and $e_2 \in \widetilde{\mathcal{O}}$ is established by applying the above string comparison procedure to

(1) both IRI's of e_1 and e_2,
(2) both ShortFormIRI's of e_1 and e_2,

[11] The distance is implemented via https://rosettacode.org/wiki/Levenshtein_distance#Java. The similarity score between $[0, 1]$ of two strings s_1, s_2 is calculated by $\frac{M - LevenstheinDistance(s_1,s_2)}{M}$ where M is $\max(s_1.length, s_2.length)$.

(3) the IRI of e_1 and the annotations of e_2,
(4) the ShortFormIRI of e_1 and the annotations of e_2.[12]

Using this string comparison procedure in lexical techniques as characterised in the previous section, we specify Algorithm 1 (see below) to detect influences of a given ODP exhibiting lexical modelling features.

For ODPs that only a structural reusable component $\mathcal{P}$ without a domain specific signature we cannot sensibly apply Algorithm 1. Instead, we only run the structural detection techniques, i.e. `AxiomTypeCheck`, `DisjointUnionCheck`, and `NAryRelationCheck`.

4 Methods

In Sect. 2, we have characterised the status quo of academic research around ODPs by a diversity of ideas regarding both the notion of ODPs itself and ODP reuse. This motivates an investigation of the research question as to how prevalent ODPs influences in biomedical ontologies are. In the following, we describe our procedure for answering this question.

Algorithm 1. Pattern Detection

Input : Ontology $\mathcal{O}$, Pattern $\mathcal{P}$
Output: Suggestive evidence for influence of $\mathcal{P}$ in $\mathcal{O}$

if *ImportByURL*$(\mathcal{O}, \mathcal{P})$ **then**
  return *Import declarations in* $\mathcal{O}$ *containing* $\mathcal{P}$
if *ImportByLocation*$(\mathcal{O}, \mathcal{P})$ **then**
  return *Import declarations in* $\mathcal{O}$ *containing* $\mathcal{P}$
if *IRICheck*$(\mathcal{O}, \mathcal{P})$ **then**
  return *all* $e \in \mathcal{O}$ *that account for evidence of the check*
if *NamespaceCheck*$(\mathcal{O}, \mathcal{P})$ **then**
  return *all* $e \in \mathcal{O}$ *that account for evidence of the check*
if *AxiomTypeCheck*$(\mathcal{O}, \mathcal{P})$ **then**
  if *SubstitutionContainmentCheck*$(\mathcal{O}, \mathcal{P})$ **then**
    return *All* σ *such that* $\sigma(\mathcal{P}) \in \mathcal{O}$
  end
if *SubstitutionEntailmentCheck*$(\mathcal{O}, \mathcal{P})$ **then**
  return *All* σ *such that* $\mathcal{O} \models \sigma(\mathcal{P})$
end

4.1 Pattern Corpus

The most well-known catalogues for ODPs are (1) the ODP Semantic Web Portal,[13] and (2) the ODPs Public Catalog.[14] Both of these catalogues reflect the

[12] We also considered using the label of entities e_1 from $\mathcal{P}$. However, these either coincide with the ShortFormIRI of e_1 or are slight variations thereof. Such variations are captured by our string comparison procedure.
[13] http://ontologydesignpatterns.org.
[14] http://odps.sourceforge.net/odp/html/index.html.

focus of the academic literature on CODPs and LODPs and contain mostly submissions for these two types. We build our corpus of ODPs according to the following criteria.

(i) The pattern was categorised as either an LODP or CODP in catalogue (1).
(ii) The pattern was published together with an ontology as its reusable component or the pattern was published with an example ontology to demonstrate its reuse.
(iii) The reusable component or example ontology can be loaded and initialised with a reasoner by the OWL API.
(iv) A CODP is documented to belong to some biomedical related domain.

This selection procedure resulted in the selection of 47 out of 155 CODPs from (1), 4 out of 18 LODPs from (1), and all 16 ODPs from (2). Selected patterns according to criteria (iv) belong to at least one of the following domains: Agriculture, Biology, Cartography, Chemistry, Decision-making, Document Management, Earth Science or Geoscience, Ecology, Event Processing, Explanation, Fishery, General, Geology, Health-care, Management, Manufacturing, Materials Science, Organisation, Participation, Parts and Collections, Physics, Planning, Product Development, Scheduling, Software, Software Engineering, Social Science, Time, Work-flow.

4.2 Ontology Corpus

We used a publicly available snapshot of BioPortal from 2017.[15] Choosing the data set that contained all ontologies in their original state, we extracted all ontologies from the archive into one folder. Any ontology that could not be loaded or handled with a reasoner in the OWL API was excluded form the study. This procedure resulted in the exclusion of 78 out of 438 ontologies resulting in a study corpus of 360 ontologies.

4.3 Experimental Design

Our empirical investigation consists of four distinct experiments.

The first experiment is designed to provide positive indications for the prevalence of ODPs exhibiting lexical features in terms of class, property, and individual names. Algorithm 1 is run over all input combinations of ontologies from the ontology corpus and the 47 CODPs from catalogue (1).

The second experiment is designed to provide negative indications for ODPs exhibiting lexical features. Here, we run the `AxiomTypeCheck` over all input combinations of ontologies from the ontology corpus and the 47 CODPs from catalogue (1). The `AxiomTypeCheck` is performed under two conditions: (a) not including the imports closure of a given ODP's $\mathcal{P}$ and (b) including the imports closure of a given ODP's $\mathcal{P}$.

[15] https://zenodo.org/record/439510#.XKK-Nt-YVhE.

The third experiment is designed to provide positive indications for ODPs that do not exhibit lexical features by definition but focus on structural modelling aspects. The `DisjointUnionCheck` and the `NAryRelationCheck` are run over all ontologies from the ontology corpus to determine the prevalence of design structures often used in LODPs.

The fourth experiment is designed to provide negative indications for ODPs that do not exhibit lexical features by definition. Analogously to experiment two, the `AxiomTypeCheck`, is run over all input combinations of ontologies from the ontology corpus and LODPs from catalogue (1) as well as ODPs from catalogue (2).

We use OWL API version 5[16] to perform our experiments.

5 Results

5.1 Experiment 1: Positive Indications for CODPs

The results of experiment 1 for positive indications of ODPs exhibiting lexical design features are summarised in Table 3.[17] Each row reports on the data generated by each subcomponent of Algorithm 1. The reported numbers in each column encode the following information: "Overall $\mathcal{P}$" is a count for the total number of ODPs for which a detection technique has produced some evidence. "Overall $\mathcal{O}$" is a count for the total number of ontologies based on which a detection technique produced some evidence. "Max $\mathcal{P}$'s in $\mathcal{O}$" is a count for the maximum number of distinct ODPs for which some evidence could be produced in a given ontology. "Max $\mathcal{O}$'s for $\mathcal{P}$" is a count for the maximum count of distinct ontologies in which evidence for a given ODP could be produced.

Note, that evidence generated by `ImportByURL` is not counted again in subsequent detection techniques. In the following, we will provide further details on these results.

Table 3. Summary of generated evidence for CODPs

Detection technique	Overall $\mathcal{P}$	Overall $\mathcal{O}$	Max $\mathcal{P}$'s in $\mathcal{O}$	Max $\mathcal{O}$'s for $\mathcal{P}$
(1) `ImportByURL`	3	1	3	1
(2) `ImportByLocation`	5	6	1	2
(3) `IRICheck`	0	0	0	0
(4) `NamespaceCheck`	4	5	2	2
(5) `SContainmentCheck`	9	46	3	20
(6) `SEntailmentCheck`	0	0	0	0

[16] http://owlcs.github.io/owlapi/apidocs_5/.

[17] `SubstitutionContainmentCheck` has been abbreviated by `SContainmentCheck` for presentation purposes. Likewise for `SubstitutionEntailmentCheck`.

(1) The ImportByURL check detected three ODPs that were undisputedly reused by import, namely the AgentRole, ObjectRole, and Classification. Interestingly, this reuse by import was only detected due to AgentRole's occurrence in the corpus of ontologies. Since each ontology is contained in its own import closure, the detection of AgentRole is as expected. Likewise, the detection of ObjectRole and Classification is unsurprising since AgentRole imports both ObjecRole and Classification. Otherwise, the ImportByURL check did not produce any evidence for these or other ODPs in the corpus of ontologies.

(2) The ImportByLocation detected five import declarations as candidates for ODP reuse via import by location. For example, the pattern Region was generated as candidate in the "Ontology of Geographical Region" since it contained the ontology "http://www.owl-ontologies.com/GeographicalRegion.owl" in its import closure. However, in all cases, an inspection of the imported ontologies and the candidate ODPs did not reveal an obvious relationship.

(3) Except IRIs pertaining to AgentRole (which are not counted again), no other IRIs pertaining to some ODP could be detected in the corpus of ontologies.

(4) The NamespaceCheck performed with "http://ontologydesignpatterns.org" resulted in the detection of 5 entities in 3 different ontologies. In all cases, a "seeAlso" annotation referenced web pages related to ODPs. For example, the object property "part of" in the "human interaction network ontology" has been annotated with "rdfs:SeeAlso <http://ontologydesignpatterns.org/wiki/Submissions:PartOf>".

(5) The SubstitutionContainmentCheck generated candidate substitutions for 9 ODPs in 46 distinct ontologies. Two out of the ODPs account for 26 of the 46 ontologies in which substitutions could be generated. These two ODPS are TypesOfEntities and GOTop. The latter is also the pattern that has generated candidate substitutions in 20 distinct ontologies. Excluding both these ODPs would have resulted in an "Overall $\mathcal{O}$" count of 18 and a "Max $\mathcal{O}$'s for $\mathcal{P}$" of 10.

(6) The SubstitutionEntailmentCheck did not result in the generation of additional candidate substitutions.

5.2 Experiment 2: Negative Indications for CODPs

The results of experiment 2 for negative indications of ODPs exhibiting lexical design features are summarised in Table 4. The table is split in the middle by a double line. Each side contains the same information content only formulated differently.

For the left hand side, the percentage in the column "Ontologies" describes a lower bound for ontologies in the ontology corpus that exhibit at least as many axioms of a given type as the number of ODPs shown in columns "Patterns (a)" and "Patterns (b)", where (a) and (b) indicates the experimental condition as described in Sect. 4.3. For example, the first row expresses that at least 5% of all ontologies in the corpus have at least as many axioms of a given type as 42 out of the 47 tested ODPs.

The right hand side of the table, formulates the complementary implication of the left hand side. Continuing the example with the first row, we can infer that for $47 - 45 = 5$ ODPs, there exists only fewer than 5% of all ontologies containing at least as many axioms of a given type as the ODP. Consequently, it can be inferred that these patterns have not influenced 95% of ontologies by ODP reuse under import, complete copying, or copying with renaming.

Table 4. Result of `AxiomTypeCheck`

Ontologies	Patterns (a)	Patterns (b)	Patterns (a)'	Patterns (b)'	Ontologies'
at least 5%	42	32	5	15	less than 5%
at least 10%	38	26	9	21	less than 10%
at least 20%	16	11	31	36	less than 20%
at least 30%	4	2	43	45	less than 30%
at least 40%	3	1	44	46	less than 40%
at least 50%	2	1	45	46	less than 50%
at least 80%	2	1	45	46	less than 80%

5.3 Experiment 3: Positive Indications for LODPs

The `DisjointUnionCheck` found evidence in 24 ontologies. None of these instances made use of the syntactic shortcut "DisjointUnion" in OWL. The `NAryRelationCheck` revealed that nearly half of all ontologies (168 out of 360) contain at least one n-ary relation.

5.4 Experiment 4: Negative Indications for LODPs

The results of experiment 4 are reported in the same fashion as the results for experiment 2 (cf. Sect. 5.2) (Table 5).[18]

6 Discussion

The results of our investigation provide only scant evidence for influences of ODPs in biomedical ontologies. The negative results of the `ImportByURL` check show that a given ODP's component $\mathcal{P}$ is not reused in practice as originally envisioned by the NeOn project. Furthermore, the negative results of our `IRICheck` indicate that even parts of reusable components $\mathcal{P}$ do not directly influence the ontology engineering tasks in practice.

[18] Experiment 4 is not designed with two conditions for including or not including a given $\mathcal{P}$'s import closure as in Experiment 2. This is owed to the fact that ODPs focusing on logical modelling structures do not import other ontologies.

Table 5. Result of `AxiomTypeCheck`

Ontologies	Patterns	Patterns'	Ontologies'
at least 5%	13	7	less than 5%
at least 10%	11	9	less than 10%
at least 20%	6	14	less than 20%
at least 30%	1	19	less than 30%
at least 40%	1	19	less than 40%
at least 50%	1	19	less than 50%
at least 80%	1	19	less than 80%

Even though we could not find explicit evidence for any ODP being reused by import, we did find evidence by the mere presence of the AgentRole pattern in the corpus of ontologies. Through manual inspection of the original 438 ontologies in the BioPortal snapshot, we noticed that the AgentRole pattern was located in an archive file for the ontology ICPS. This archive also contained another pattern, namely Person. However, the ontology ICPS has been excluded during the process of the ontology corpus construction for the study. This observation raises the question whether our results are skewed by our ontology exclusion criteria for constructing the experimental ontology corpus. We can invalidate this concern due to the following. First, we downloaded a version of the BioPortal snapshot in which each ontology has been merged with its import closure. Then, we treated all ontologies as simple text files and reran the `NamespaceCheck`. Still, there is no positive hit to be reported.

Inspecting the positive evidence found by the `IRICheck`, it is quite clear that practitioners create their own entities instead of reusing IRIs from ODPs directly. Nevertheless, it remains unclear whether this is owed to a conscious modelling decision, mere personal preference, lack of know-how, or lack of tool support for ODPs.

Yet, there is a caveat with respect to reusing IRIs from ODPs that needs to be pointed out. Some ODPs published on http://ontologydesignpatterns.org are said to be "extracted from upper level ontologies". However, interestingly, their respective reusable components $\mathcal{P}$ are often self-contained ontologies not bearing any relation to upper level ontologies. This suggests that $\mathcal{P}$ is a somehow reimplemented fragment of the upper level ontology. This gets practitioners into the predicament of choosing between aligning their ontologies to an upper level ontology or an ODP (if they are so inclined in the first place). Hence, it is possible that practitioners prefer to work with the original upper level ontology rather than the extracted ODPs thereof.

Irrespective of any matter of renaming, the findings of our `AxiomTypeCheck` suggest that modelling features exhibited by most reusable components of ODPs are not highly prevalent in ontologies of the biomedical domain (the vast majority of ODPs contain axiom types that are not present in more than 70% of ontologies). It has been noted before that an ODP's required language expressivity is outside of the popular EL profile many biomedical ontologies conform to [11]. Moreover, it seems that a fair amount of published ODPs seem to propose property centric modelling approaches whereas ontologies in the biomedical domain tend to follow a class centric design.

Since a high percentage of ontologies do not contain at least the same number of axioms or axioms types as a given ODP, it is unsurprising to find a limited number of candidates under the `SubstitutinoContainmentCheck`. Likewise, it is equally unsurprising to find a limited number of candidates under the `SubstitutionEntailmentCheck`, given the observation that a fair number of ODPs make use of modelling techniques that are not expressible in the EL profile to which a lot of ontologies conform.

Given the above observation with respect to axiom types and differences in language requirements, we considered to relax the conditions of our substitution checks. Instead of requiring a substitution for *all* axioms $\alpha \in \mathcal{P}$, we only require a substitution for some subset $S \subseteq \mathcal{P}$ such that $\sigma(\alpha) \in \mathcal{O}$ holds for all $\alpha \in S$. Essentially, this corresponds to some notion of a *partial* reuse of $\mathcal{P}$. Allowing for arbitrary subsets $S \subseteq \mathcal{P}$ resulted in the generation of a large amount of spurious data due to our liberal lexical association procedure. Imposing some lower bound on the size of S is not straightforward as an ODP's $\mathcal{P}$ is often quite small to begin with. Limiting the search space for lexical associations in the target ontology $\mathcal{O}$ by some heuristics seems to be the most promising approach. For example, given a match between some $e \in \widetilde{\mathcal{P}}$ and $e' \in \widetilde{\mathcal{O}}$, limit the search for further lexical associations of elements in $\widetilde{\mathcal{P}}$ to the set $\{\alpha \in \mathcal{O} \mid e' \in \widetilde{\alpha}\}$ and proceed recursively. However, slight variations in heuristic search strategies result in drastic effects for the number of generated lexical associations. Overall, generating meaningful data for partial reuse of a given ODP's $\mathcal{P}$ turns out to be a challenging research endeavour in and of itself.

6.1 Limitations

Despite our intention to maximise the recall of our detection mechanism, there are a few limitations. Some patterns in our corpus are not intended to be directly reused via some reusable component $\mathcal{P}$. The ODP UpperLevelOntology[19] is such an example. This pattern motivates to align a given ontology to a chosen upper level ontology. Since all our detection techniques are agnostic to influences of upper level ontologies and only target lexical as well as structural modelling features, the prevalence of ontologies aligned to upper level ontologies is not determined and our negative results are inconclusive.

Another limitation is the manner in which we try to establish lexical associations between entities of ODPs and entities of domain ontologies. Entities of ODPs are arguably of general nature and might not easily be associated via with domain specific entities on a purely lexical basis. Instead, one might need to consider lexical relationships based on hyponyms and hypernyms. However, doing

[19] http://odps.sourceforge.net/odp/html/Upper_Level_Ontology.html.

so would require an more overall more sophisticated lexical matching procedure to prevent spurious associations.

The choice of both the ontology corpus as well as the ODP corpus limit the generalisability of our findings. Despite BioPortal's popularity for empirical research, based on a large variety of ontologies differing in size and complexity that are authored by a number of independent groups for diverse intents and purposes [16], there is still a possibility that the used BioPortal snapshot in our study is not representative for biomedical ontologies in general. Likewise, it is possible that the constructed corpus of ODPs is not representative of patterns that are relevant for the biomedical domain. However, if we (hypothetically) assume that the design of many biomedical ontologies is indeed informed by a pattern-based approach, then this would raise several questions such as why these patterns would not be readily available in well-known public repositories, or why would an ontology not document and advertise its pattern-based design explicitly.

6.2 Related Work

Empirical work on ODP reuse often falls into one of two categories. On the one hand, there are user studies that investigate how a given set of ODPs affects the completion of an ontology engineering tasks in an experimental setting. On the other hand, there are field studies that investigate qualities of ODP reuse outside an artificially created experimental setting.

Existing user studies reveal mixed user perceptions. ODPs are sometimes deemed useful [2] but are also often met with scepticism [8,10] and experiences from ontology engineers reveal tangible limitations of ODP reuse in practice [14,23].

Existing field studies on ODP reuse either aim to detect the reuse of known ODPs, or aim for the discovery of regularities in ontologies that may be interpreted as the reuse of (potentially unknown) ODPs.

Ontology enrichment has motivated one of the first attempts to automatically identify the reuse of ODPs in ontologies [19]. It is argued that the identification of partial ODP reuse may allow for ontology refinement by completing the missing parts of a pattern. The proposed mechanism to identify the partial reuse of known ODPs heavily depends on a lexical association procedure that is based on a number of heuristics. However, a large scale evaluation of the proposed mechanism is not performed.

The idea of using lexical associations between entities occurring in ontologies and entities occurring in ODPs has motivated the proposal of a detection mechanism that uses WordNet[20] to provide background knowledge for establishing such lexical associations [13]. However, a first empirical evaluation suggests that the results are "probably not reliable" because the background knowledge provided by WordNet is used in a way that skews the data towards patterns including a certain signature and produces spurious results.

[20] https://wordnet.princeton.edu/.

Acknowledging the limitations of lexical associations for the purpose of detecting ODP reuse, it has been proposed to combine lexical and structural aspects of an ODP's design into detection procedures [27]. The idea is to use query languages, e.g. SPARQL, to probe an ontology for axioms that satisfy structural constraints imposed by an ODP's design. Only if such axioms are found, a lexical association procedure is applied to identify a potential ODP's reuse. A preliminary evaluation suggests that the precision of the proposed approach needs to be improved by using query engines that are tailored towards OWL ontologies, e.g. SPARQL-DL.

Another study combining both lexical and structural aspects of an ODP for its detection aims disregards ODP reuse under lexical variation of its entities as this is considered an ill-defined task [18]. Here, a lexical search is performed to determine whether all entities of a given ODP occur in a target ontology. Only if instances for all entities of an ODP are found, then a structural comparison between both the ontology's and the ODP's axioms is performed under some notion of normalisation. A large scale study using this approach reveals the reuse of a small number of structurally simple ODP in biomedical ontologies.

Contrary to the negative results of studies searching for evidence of the reuse of published ODP, studies on regularities in ontologies report recurring pattern of axioms both within as well as across a large number of biomedical ontologies in BioPortal [15,17].

7 Conclusion

The results of our empirical evaluation corroborate the findings of previous studies to some degree [18]. Our pattern detection mechanism could not provide much concrete evidence for ODPs influence in biomedical ontologies. Even liberal notions for ODP reuse which can only be considered suggestive of a given ODP's influence do not allow for a different conclusion. While this negative finding appears unconstructive, we will qualify its implications in light of the nature of our chosen detection techniques.

The structural detection techniques, `AxiomTypeCheck` and `DisjointUnionCheck`, indicate that modelling solutions proposed by ODPs differ significantly compared with ontologies authored by practitioners. The data collected by the `AxiomTypeCheck` shows that the design of most biomedical ontologies are class centric while the design of ODPs published in catalogue (1) (cf. Sect. 4.1) places an emphasise on roles. As for disjoint unions, six out of 16 ODPs published in catalogue (2) (cf. Sect. 4.1) feature a disjoint union. Yet, only 7% of ontologies in our study make use of such an expression. Overall, it seems that currently, ODPs do not provide solutions to *common* ontology design tasks for ontology engineers in the biomedical domain. In particular, the scarce positive evidence for ODPs suggest that practitioners in the biomedical domain seem to limit the reuse of ODPs to the realm of annotations (cf. results of the `NamespaceCheck`).

Overall, there seems to be a discrepancy between the lack of reuse of publicly available ODPs on the one hand and ontology engineering techniques that give rise to regular logical structures in biomedical ontologies on the other hand, as shown in [15,17]. However, this discrepancy may be reconciled by motivating a data driven approach that automatically generates or at least informs the development of practically relevant ODPs. In such a scenario, detection techniques, such as the ones presented in this paper, can serve as some kind of quality measure for newly discovered pattern. After a pattern is discovered, one can either gauge its practical relevance by determining its prevalence in other ontologies or by monitoring the uptake of the discovered pattern by practitioners over time. The desire for such work has already been expressed [9].

References

1. Blomqvist, E.: Ontology patterns: typology and experiences from design pattern development. In: The Swedish AI Society Workshop, Uppsala University, 20–21 May 2010, no. 048, pp. 55–64. Linköping University Electronic Press (2010)
2. Blomqvist, E., Gangemi, A., Presutti, V.: Experiments on pattern-based ontology design. In: K-CAP, pp. 41–48. ACM (2009)
3. Blomqvist, E., Sandkuhl, K.: Patterns in ontology engineering-classification of ontology patterns. In: ICEIS 2005: Proceedings of the Seventh International Conference on Enterprise Information Systems, Miami, USA, 25–28 May 2005 (2005)
4. Clark, P.: Knowledge patterns. In: Gangemi, A., Euzenat, J. (eds.) EKAW 2008. LNCS (LNAI), vol. 5268, pp. 1–3. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87696-0_1
5. de Almeida Falbo, R., Guizzardi, G., Gangemi, A., Presutti, V.: Ontology patterns: clarifying concepts and terminology. In: WOP, CEUR Workshop Proceedings, vol. 1188. CEUR-WS.org (2013)
6. Gangemi, A.: Ontology design patterns for semantic web content. In: Gil, Y., Motta, E., Benjamins, V.R., Musen, M.A. (eds.) ISWC 2005. LNCS, vol. 3729, pp. 262–276. Springer, Heidelberg (2005). https://doi.org/10.1007/11574620_21
7. Guizzardi, G.: Theoretical foundations and engineering tools for building ontologies as reference conceptual models. Semant. Web **1**(1–2), 3–10 (2010)
8. Hammar, K.: Ontology design patterns in use: lessons learnt from an ontology engineering case. In: Proceedings of the 3rd International Conference on Ontology Patterns, vol. 929, pp. 13–24. CEUR-WS.org (2012)
9. Hammar, K., et al.: Collected research questions concerning ontology design patterns. In: Ontology Engineering with Ontology Design Patterns. Studies on the Semantic Web, vol. 25, pp. 189–198. IOS Press (2016)
10. Hammar, K., Presutti, V.: Template-based content ODP instantiation. In; The 7th Workshop on Ontology and Semantic Web Patterns. IOS Press (2017)
11. Horridge, M., Aranguren, M.E., Mortensen, J., Musen, M.A., Noy, N.F.: Ontology design pattern language expressivity requirements. In: WOP, CEUR Workshop Proceedings, vol. 929. CEUR-WS.org (2012)
12. Hou, C.-S.J., Noy, N.F., Musen, M.A.: A template-based approach toward acquisition of logical sentences. In: Musen, M.A., Neumann, B., Studer, R. (eds.) IIP 2002. ITIFIP, vol. 93, pp. 77–89. Springer, Boston, MA (2002). https://doi.org/10.1007/978-0-387-35602-0_8
13. Khan, M.T., Blomqvist, E.: Ontology design pattern detection-initial method and usage scenarios. In: SEMAPRO 2010, The Fourth International Conference on Advances in Semantic Processing, pp. 19–24 (2010)

14. Lantow, B., Sandkuhl, K., Tarasov, V.: Ontology reuse. In: KEOD, pp. 163–170. SciTePress (2015)
15. Lawrynowicz, A., Potoniec, J., Robaczyk, M., Tudorache, T.: Discovery of emerging design patterns in ontologies using tree mining. Semant. Web **9**(4), 517–544 (2018)
16. Matentzoglu, N., Bail, S., Parsia, B.: A corpus of OWL DL ontologies. In: Description Logics, CEUR Workshop Proceedings, vol. 1014, pp. 829–841. CEUR-WS.org (2013)
17. Mikroyannidi, E., Manaf, N.A.A., Iannone, L., Stevens, R.: Analysing syntactic regularities in ontologies. In: OWLED, CEUR Workshop Proceedings, vol. 849. CEUR-WS.org (2012)
18. Mortensen, J., Horridge, M., Musen, M.A., Noy, N.F.: Modest use of ontology design patterns in a repository of biomedical ontologies. In: WOP, CEUR Workshop Proceedings, vol. 929. CEUR-WS.org (2012)
19. Nikitina, N., Rudolph, S., Blohm, S.: Refining ontologies by pattern-based completion. In: WOP, CEUR Workshop Proceedings, vol. 516. CEUR-WS.org (2009)
20. Presutti, V., Gangemi, A.: Content ontology design patterns as practical building blocks for web ontologies. In: Li, Q., Spaccapietra, S., Yu, E., Olivé, A. (eds.) ER 2008. LNCS, vol. 5231, pp. 128–141. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-87877-3_11
21. Presutti, V., et al.: D2.5.1: a library of ontology design patterns: reusable solutions for collaborative design of networked ontologies (2008). http://www.neon-project.org/
22. Renée Reich, J.: Onthological design patterns for the integration of molecular biological information. In: German Conference on Bioinformatics, pp. 156–166 (1999)
23. Rodriguez-Castro, B., Ge, M., Hepp, M.: Alignment of ontology design patterns: class as property value, value partition and normalisation. In: Meersman, R., et al. (eds.) OTM 2012. LNCS, vol. 7566, pp. 682–699. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-33615-7_16
24. Ruy, F.B., Reginato, C.C., Santos, V.A., Falbo, R.A., Guizzardi, G.: Ontology engineering by combining ontology patterns. In: Johannesson, P., Lee, M.L., Liddle, S.W., Opdahl, A.L., López, Ó.P. (eds.) ER 2015. LNCS, vol. 9381, pp. 173–186. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-25264-3_13
25. Staab, S., Erdmann, M., Maedche, A.: Engineering ontologies using semantic patterns. In: OIS@IJCAI, CEUR Workshop Proceedings, vol. 47. CEUR-WS.org (2001)
26. Suárez-Figueroa, M.C., et al.: D5.1.1 NeOn modelling components, March 2007. http://www.neon-project.org
27. Sváb-Zamazal, O., Scharffe, F., Svátek, V.: Preliminary results of logical ontology pattern detection using SPARQL and lexical heuristics. In: WOP, CEUR Workshop Proceedings, vol. 516. CEUR-WS.org (2009)
28. Svátek, V.: Design patterns for semantic web ontologies: motivation and discussion. In: 7th Conference on Business Information Systems (BIS-2004) (2004)
29. Vrandecic, D.: Explicit knowledge engineering patterns with macros. In: Proceedings of the Ontology Patterns for the Semantic Web Workshop a the ISWC 2005, Galway, Ireland (2005)