

RESEARCH

Open Access



Towards semantic interoperability: finding and repairing hidden contradictions in biomedical ontologies

Luke T. Slater^{1,2*} , Georgios V. Gkoutos^{1,2,3,4,5,6} and Robert Hoehndorf⁷

Abstract

Background: Ontologies are widely used throughout the biomedical domain. These ontologies formally represent the classes and relations assumed to exist within a domain. As scientific domains are deeply interlinked, so too are their representations. While individual ontologies can be tested for consistency and coherency using automated reasoning methods, systematically combining ontologies of multiple domains together may reveal previously hidden contradictions.

Methods: We developed a method that tests for hidden unsatisfiabilities in an ontology that arise when combined with other ontologies. For this purpose, we combined sets of ontologies and use automated reasoning to determine whether unsatisfiable classes are present. In addition, we designed and implemented a novel algorithm that can determine justifications for contradictions across extremely large and complicated ontologies, and use these justifications to semi-automatically repair ontologies by identifying a small set of axioms that, when removed, result in a consistent and coherent set of ontologies.

Results: We tested the mutual consistency of the OBO Foundry and the OBO ontologies and find that the combined OBO Foundry gives rise to at least 636 unsatisfiable classes, while the OBO ontologies give rise to more than 300,000 unsatisfiable classes. We also applied our semi-automatic repair algorithm to each combination of OBO ontologies that resulted in unsatisfiable classes, finding that only 117 axioms could be removed to account for all cases of unsatisfiability across all OBO ontologies.

Conclusions: We identified a large set of hidden unsatisfiability across a broad range of biomedical ontologies, and we find that this large set of unsatisfiable classes is the result of a relatively small amount of axiomatic disagreements. Our results show that hidden unsatisfiability is a serious problem in ontology interoperability; however, our results also provide a way towards more consistent ontologies by addressing the issues we identified.

Keywords: Ontology interoperability, Automated reasoning

Background

Ontologies are used to describe and organise domain knowledge in the biomedical sciences. Ontologies use classes to characterise the kinds of things that exist within a domain as well as axioms that provide constraints for these classes and conditions that must be satisfied within the domain. Most ontologies in biology are domain-specific and focus on a single domain. Creating ontologies that reference and extend other biomedical ontologies is

*Correspondence: l.slater.1@bham.ac.uk

¹ College of Medical and Dental Sciences, Institute of Cancer and Genomic Sciences, University of Birmingham, Birmingham B15 2TT, UK

Full list of author information is available at the end of the article



common practice, as it promotes a unified understanding of the biomedical domain by defining terms and groups of terms in the context of their relationships with classes from related domains, and in the common context of higher level domains. Reusing the formalised knowledge from other domain ontologies also enables the reuse of expertise from ontology developers in other domains.

The majority of biomedical ontologies are now being developed in the Web Ontology Language (OWL) [1], a formal model-theoretic language based on description logics [2]. OWL ontologies enable the use of automated reasoners, which in turn enable the deductive inference of knowledge implied by the explicit assertions made in the ontologies. Furthermore, these inferences can be examined to determine whether an ontology's classes are satisfiable, and whether an ontology is consistent. A class is satisfiable if it can have an instance, and is unsatisfiable if it contains a contradiction such that an instance of the class would force any model of the ontology to contain a logical contradiction; an ontology is inconsistent if it contains at least one instance of a logical contradiction. Unsatisfiable classes and inconsistencies arise most frequently by violation of a disjointness axiom. For example, if an ontology contains an axiom asserting that a disease and a phenotype are disjoint, then any class that is a subclass of both disease and phenotype is unsatisfiable. An ontology which contains any instances of an unsatisfiable class is inconsistent, while an ontology which contains any unsatisfiable classes is termed incoherent.

Automated reasoners can also be used to generate explanations for an unsatisfiability. An explanation is a small set of axioms which are sufficient to reproduce the contradiction. An explanation can be used to diagnose the cause of the class becoming unsatisfiable.

The Open Biomedical Ontologies (OBO) Foundry is a collection of ontologies that use a shared set of design principles, and encourages re-use of terms amongst them [3]. The ontologies are built using the framework provided by common upper-level ontology, the Basic Formal Ontology (BFO) [4], and include many large and widely used domain ontologies describing areas such as chemical entities [5], phenotypes [6], and model organisms [7]. Using standard upper-level ontologies is intended to support consistency between multiple ontologies and knowledge integration across domains [8].

From a technical perspective, OWL caters for the inclusion (i.e., import) of complete ontologies so that they can be reused and built upon. Importing an ontology amounts to including all the entities and axioms of another ontology in the importing ontology. While this is a provision of simple modularity, it enables re-use of classes and axioms across ontologies, and it enables automated reasoners to detect joint consistency.

However, full import of an ontology is not always sensible or feasible. Even when an ontology makes heavy use of the classes and axioms in another ontology, only a subset of the classes are likely to be relevant within another ontology.

For example, the Hypertension Ontology (HTN) [9] expands upon the hypertension classes in the Human Phenotype Ontology (HP) [6] and the Disease Ontology (DO) [10], but is not concerned with any terms in those ontologies besides those directly related to hypertension. To include all of the classes in HP and DO in HTN is vulnerable to potential issues resulting from the inclusion of irrelevant classes. Loading the ontology would take longer, in particular when imported ontologies are retrieved over the internet. Editing an ontology may become challenging when many classes from other ontologies are included on account of the large number of additional classes that must be loaded, classified, and possibly visualised. Overall, an ontology importing a large number of other ontologies becomes more difficult to use with the relevant classes being hidden within the hierarchy of the imported ontologies.

In response to these technical challenges, the research community has investigated different models for ontology modularisation. Particularly, work has investigated locality-based module extraction [11], which can be used to improve reasoner-based query performance and support large-scale ontology development and re-use [12].

The MIREOT (Minimum Information to Reference an External Ontology Term) guidelines were originally developed to support inclusion of classes from non-OBO Foundry ontologies without needing to align to their axiomatisation, and has become a standard for term re-use and inclusion throughout the biomedical ontology community [13].

MIREOT relaxes the import of other ontologies through including all axioms and instead focuses on the reuse of individual classes from other ontologies. Particularly, the MIREOT guidelines stipulate that three pieces of information are necessary to "reference" an external ontology class:

<i>Source ontology</i>	The Internationalised Resource Identifier (IRI) of the ontology which contains the class being included.
<i>Source class</i>	The IRI of the class to import, as defined in the external ontology.
<i>Direct Superclass</i>	The IRI of the direct superclass of the imported class in the referencing ontology.

Utilizing these three pieces of information, an external ontology class can be referenced. By including MIREOT definitions for each relevant external class, a module is formed within the imported ontology without fully importing any external ontologies. While this method allows ontologies to reuse classes in a scalable and efficient manner, the inclusion of external classes without the context of the external ontology's axioms means that contradictions may arise that cannot be detected using an automated reasoner that evaluates only the importing ontology. This may lead ontology developers to build upon another class in a way that contradicts its original definition. Furthermore, subsequent versions of the source ontology may re-axiomatise a subject class in a way which renders its use in the importing ontology incompatible with it.

Previous work has discussed in a general sense the kinds of conceptualisation mismatches that can occur in ontologies[14]. Another work axiomatised the Gene Ontology (GO) with respect to upper level ontologies, identifying large sets of conflicting ontologies, and detailing a basic semi-automated approach to resolving contradictions[15].

More recently, our prior analysis of the Experimental Factor Ontology (EFO) [16] showed that the use of MIREOT has the potential to cause inconsistency and unsatisfiability across the set of ontologies the EFO references [17]. While our previous work revealed problems with EFO, the extent and exact characterisation of this problem throughout the entire biomedical ontology ecosystem has not yet been explored. It is also unknown whether there are common roots to widespread unsatisfiability. More importantly, while identifying unsatisfiable classes and inconsistencies is important, it would be much more useful to resolve them, ideally automatically or semi-automatically. It is not clear whether the unsatisfiability can be automatically repaired.

We explore interoperability and hidden unsatisfiability throughout the OBO Foundry ontologies. To do this we extend the unMIREOT tool described by our previous work, and generalise it to reveal hidden contradictions in any combination of OWL ontologies, and identify the axioms involved [17]. This analysis reveals many cases of incoherency and inconsistency throughout the ontology ecosystem.

Based on the information revealed by our analysis, we present a novel algorithm that generates explanations for unsatisfiability, and uses these explanations to systematically identify a small list of axioms that can be removed from an ontology to repair all cases of unsatisfiability and generate a novel ontology that is both consistent and coherent. The list is formed by automatically evaluating explanations for unsatisfiable classes. We then use the

algorithm to report on sources of the contradictions we found throughout the OBO ontologies, and the axioms that are most frequently involved.

Our method and tools allows detection of unsatisfiable classes and the systematic, semi-automatic repair of ontologies. Applying our approach will lead to higher quality ontologies maintaining consistency in the rapidly evolving web of knowledge that spans biology and biomedicine. All our results and software are freely available at <https://github.com/bio-ontology-research-group/UNMIREOT>.

Methods

Ontologies and ontology versions

All non-deprecated and obtainable OBO ontologies were downloaded using the permanent download links given by the OBO Foundry database at

<http://obofoundry.org/registry/ontologies.yml>. A total of 132 ontologies were obtained on 28/03/2018.

Our experiments concern two sets of ontologies described by this database. First, the OBO Foundry ontologies, which are judged as satisfying the OBO Foundry principles, and are therefore tightly integrated and also widely used across many domains. The second is the wider set of ontologies included in the OBO database. In the rest of this paper, we will refer to the core ontologies as the OBO Foundry ontologies, while the wider set of ontologies will be referred to as the OBO ontologies. There are 9 OBO Foundry ontologies considered in this work, and a further 123 OBO ontologies.

Implementation and experimental setup

For all experiments, we use the OWLAPI 5.1.4 [18]. To classify the ontologies and to retrieve unsatisfiability explanations, we use the Elk reasoner version 0.5.0-SNAPSHOT [19].

Elk supports the OWL 2 EL profile, a fragment of OWL that supports tractable (i.e., polynomial-time) reasoning, but lacks support for many logic operators. In particular, OWL 2 EL does not support the use of negation in class descriptions or use of the universal quantifier. The only type of axiom in OWL 2 EL that could result in an explicit contradiction is the disjointness axiom. In our previous investigation, we attempted to evaluate other reasoners for the purpose of identifying additional unsatisfiable classes, finding that either the reasoner did not finish, or only found a small amount of additional unsatisfiability. It is possible that a more expressive reasoner would reveal additional contradictions for ontology combinations with which the reasoner would finish, such as through evaluation of logical negation. Furthermore, the repeated use of satisfiability checking required by the

unsatisfiability justification investigation necessitate a low running time.

We also used Protégé with Elk to examine some of the combined ontologies for particular cases of unsatisfiability [20] that we give detailed explanations for in the results. Justifications were equivalent to those revealed by the tool.

Results

Combining ontologies and detecting inconsistencies

We created a tool to combine sets of ontologies. This works by loading all source ontologies with OWLAPI, creating a new meta-ontology, and copying all classes and axioms into it. This tool is also available via the unMIR-EOT GitHub.

We combined all of the OBO Foundry ontologies into one meta-ontology. Figure 1 summarises the ontologies in this set. We did not include the ontologies referenced in the imports closures of the OBO Foundry ontologies, since in all cases these ontologies were included in the larger set of OBO ontologies, and therefore their combined consistency would be evaluated later. Subsequently, we evaluated the combined ontology for unsatisfiability and its causes.

The 9 OBO Foundry ontologies combined consist of 402,868 logical axioms and 207,105 named classes. The use of an automated reasoner on the combined OBO Foundry meta-ontology determined that 636 of these classes are unsatisfiable. Table 1 shows the number of unsatisfiable classes and the ontology to which they belong. The origin ontology of the classes was determined using the class IRI prefix. While each of these classes is unsatisfiable due to a different set of axioms, there may be a small set of axioms that are shared by several cases of unsatisfiability. We developed an algorithm to identify a small set of axioms that are sufficient to explain all unsatisfiable classes in an ontology; if this set of axioms is removed from an ontology, all cases of unsatisfiability are resolved. We apply this algorithm to the combined OBO Foundry ontologies in order to derive a coherent version, removing two axioms. The algorithm, and the axioms it removes, are described in detail in the *Efficient ranking and repairing of axioms* section.

Table 1 Unsatisfiable class counts in OBO Foundry

Ontology	Unsatisfiable class count
CHEBI	37
GO	565
OBI	34

BFO Basic Formal Ontology[33]
CHEBI Chemical Entities of Biological Interest[5]
DO Disease Ontology[10]
GO Gene Ontology[38]
OBI Ontology for Biomedical Investigations[39]
PATO Phenotypic Quality Ontology[40]
PO Plant Ontology[41]
XAO Xenopus Anatomy and Development Ontology[42]
ZFA Zebrafish Anatomy and Development Ontology[43]

Fig. 1 Ontologies included in the OBO Foundry

We combine this coherent version of the OBO Foundry meta-ontology with each of the OBO ontologies individually, classifying the resulting merged ontology, using an automated reasoner to determine if there are any unsatisfiable classes; if we identify unsatisfiable classes we count their number. Out of all 132 ontologies that we use in this experiment, we revealed unsatisfiable classes in 50 ontologies. The 10 OBO ontologies with the most unsatisfiable classes are listed in Table 2. The total number of unsatisfiable classes across all OBO ontologies is 866,494 and the total number of unique unsatisfiable classes is 312,398. Of these, 8,893 are obsolete classes, which are intentionally unsatisfiable (and thus not considered an error). In addition, the Ontology of Vaccine Adverse Events (OVAE) [21], Food Ontology (FOODON) [22], Plant Trait Ontology (TO) [23], Gazetteer (GAZ) [24], Porifera (PORO) [25], Plant Experimental Conditions Ontology (PECO) [23], Oral Health and Disease Ontology (OHD) [26], and Statistics Ontology (STATO) [27] became inconsistent.

Efficient ranking and repairing of axioms

Our algorithm for identifying the causes for unsatisfiability in ontologies builds upon a black-box algorithm

Table 2 The ten ontologies with the most unsatisfiable classes in the OBO ontologies, when combined with a repaired version of the merged OBO Foundry ontology

Ontology name	Unsatisfiable class count
Unified Phenotype Ontology (UPHENO) [44]	106,126
Monarch Disease Ontology (MONDO) [45]	97,619
Ontology for MIRNA Target (OMIT) [46]	63,015
Molecular Process Ontology (MOP) [47]	57,355
Name Reaction Ontology (RXNO) [48]	57,330
Human Phenotype Ontology (HP) [6]	46,075
Mammalian Phenotype Ontology (MP) [7]	43,806
Cell Ontology (CL) [49]	34,685
Ontology of Biological Attributes (OBA) [50]	26,523
Ontology of Adverse Events (OAE) [51]	20,566

for computing a justification for one unsatisfiable class. A justification is a minimal set of axioms which explain why the class is unsatisfiable. The black-box algorithm we employ creates an empty ontology containing only the class that is unsatisfiable; it then adds new axioms from the original ontology to it, until the class becomes unsatisfiable. Axioms that are not necessary for the class to become unsatisfiable are then removed using a backwards stepwise approach, eventually producing a minimal set of axioms that constitute a justification for the unsatisfiability of the class in the original ontology. Justification algorithms are usually used as debugging tools to direct ontology developers towards the causes of unsatisfiability. For this reason, they are often integrated into ontology development environments such as the Protégé software [28].

The naive algorithm, for finding a minimal set of justifications that can be removed to repair all cases of unsatisfiability, uses the black box algorithm to compute justifications for all unsatisfiable classes in the ontology, and then removes the axiom that appears most frequently in the set of all justifications. Subsequently, it then repeats this step until all cases of unsatisfiability are solved. This algorithm works well when manually investigating a small number of unsatisfiable classes in relatively small ontologies (thousands of classes rather than tens or hundreds of thousands of classes).

In our evaluation of the OBO ontologies, we revealed a very large number of unsatisfiable classes across many ontologies, some of which are very large. In the most prolific case, the Unified Phenotype Ontology (UPHENO) gives rise to 106,126 unsatisfiable classes, containing only 133,480 classes in total. Such a large number of unsatisfiable classes makes the naive algorithm intractable. In the worst case, our black-box algorithm has to add all axioms from the ontology, and then remove all but one of these axioms in order to find a single justification for one class, leading to a time complexity of $\mathcal{O}(n \cdot m)$ where n is the number of axioms and m the number of unsatisfiable classes; since each step further involves computing satisfiability, which has cubic complexity in the number of classes (and relations) [19], it is obvious that the algorithm will not scale to large numbers of unsatisfiable classes.

We develop an improved algorithm for finding a small set of axioms to remove from an ontology to repair all cases of unsatisfiability by a consideration of the problem according to the hitting set problem.

In the theory of system diagnosis, we consider a series of conflict sets, each describing a conflicting set of system components—a subset of elements from a universal set of system components. A hitting set is one which intersects every conflict set, and the hitting set problem is the

problem of computing all the minimal hitting sets for the conflict sets [29].

The problem is useful in cases where repairing or removing all of the elements in a hitting set would repair a system. The hitting set problem is equivalent to the set cover problem [30], and both problems are known to be NP-complete through reduction to the boolean satisfiability problem [31].

Our problem can be reduced to the hitting set problem, because an unsatisfiability justification can be considered as a conflicting set of axioms which can be resolved by removing one of its members from the ontology. To completely remove all axioms causing unsatisfiable classes in an ontology, all justifications must be resolved.

A hitting set of axioms to remove from the ontology to repair all axioms, therefore, must have a non-empty intersection with every unsatisfiability justification. The problem of finding all justifications for a single entailment in an ontology has previously been reduced to the hitting set problem, and then solved using Reiter's Hitting Set Tree (HST) algorithm [32]. The problem we need to solve is similar, however we need to identify a hitting set of axioms that resolve *all* cases of unsatisfiability in the ontology instead of just the axioms that cause unsatisfiability of a single class.

We develop an algorithm that exploits the fact that classes transitively inherit unsatisfiability through subclass axioms; if C is unsatisfiable and the ontology contains $D \sqsubseteq C$ as an axiom, then D will also be unsatisfiable. Consequently, we prioritise resolving unsatisfiabilities for classes that have the largest number of (asserted) subclasses in the ontology; when we resolve the cause of such a class becoming unsatisfiable, we also resolve the inherited causes of unsatisfiability for their subclasses without explicitly needing to generate a justification for them. In the worst case, this optimisation step will have no effect, because any class may have multiple causes of unsatisfiability independent from its parent class. If that is the case, the performance would be equivalent to the naive algorithm described above. However, commonly, if we assume that there are only a small number of overall causes of unsatisfiability in the ontology, we will reduce the number of justifications generated significantly.

Our algorithm is shown in Fig. 2. The algorithm takes an ontology O as input and determines the set of unsatisfiable classes in O , $v(O)$; the algorithm then removes from $v(O)$ all classes that have an asserted superclass in $v(O)$. This step ensures that for each cluster of unsatisfiability, the most general class within the ontology taxonomy is examined first. The algorithm then selects the group of classes with the highest number of directly asserted subclasses, and either generates justifications for all of these classes or for a random sample of them if the number

```

Data: o = Given ontology
Result: Small set of axioms required to repair ontology
Load and classify ontology o
x = unsatisfiable classes
while x > 0 do
  y = x without leaf classes (zero subclasses in o)
  if size(y) < size(x) then
    y = y without classes which have a superclass in set y
    z = group classes in y by total number of direct subclasses in o
    x = max(z.key)
  end
  if size(x) > 25 then
    | x = randomly sample 25 classes from x
  end
  c = implicated axioms for each class in x
  Count axioms in c, and remove the maximally implicated axiom from o
  Reclassify ontology o
end

```

Fig. 2 Algorithm for automatic diagnosis and repair of unsatisfiable classes in an ontology

of direct subclasses is above a threshold n . We selected a value of $n = 25$ manually, since it provides a relatively large sampling size for justification components, as well as providing a hard limit for explanations that must be obtained on every turn, thus limiting execution time. It is possible that different values of n , as well as random sampling of classes when the number of classes is above n , could affect the final list of axioms produced, depending on the structure of the ontology. The most frequently occurring axiom in these justifications is then removed, and the ontology is reclassified, to produce another set of unsatisfiable classes, upon which the process is repeated; the algorithm terminates when all unsatisfiable classes have been resolved.

In the selection step, our algorithm uses asserted subclasses instead of inferred subclasses because each unsatisfiable class may be affected by unwanted inferences, and is an inferred equivalent to *owl:Nothing*. It is possible that a class has more direct subclasses than another yet a fewer number of total subclasses; however, this effect is controlled by removing any classes with a superclass in the set of unsatisfiable classes $\nu(O)$. Figure 3 describes an example of the selection process the algorithm uses.

Throughout execution of the algorithm, we record statistics on the set of classes that become satisfiable after the removal of each axiom. These statistics enable ontology developers to identify problematic axioms that affect groups of ontologies, and manually resolve them.

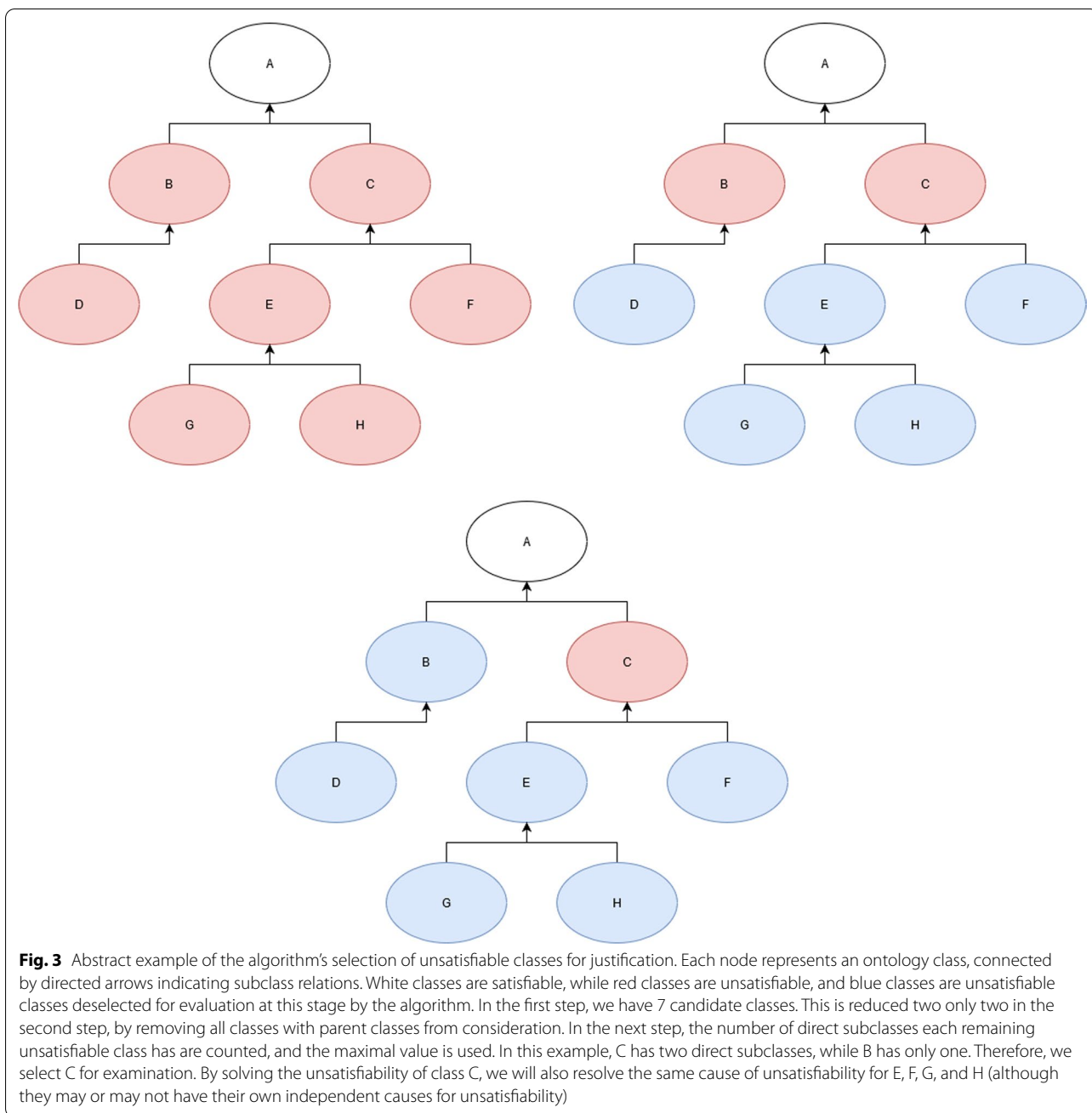
Application to OBO foundry

We applied our algorithm first to the merged OBO Foundry ontology, finding that two axioms could be removed to solve all cases of unsatisfiability:

- 1 'realizable entity' (BFO:0000017)
SubClassOf: 'specifically dependent
continuant' (BFO:0000020) with 599 classes
repaired, and
- 2 'molecular entity' (CHEBI:23367)
SubClassOf: 'material entity'
(BFO:0000040) with 37 classes
repaired.

These two axioms are members of the smallest set of axioms that suffices to remove all unsatisfiabilities. We could also consider the unsatisfiable classes as a result of violating disjointness axioms; in particular, all the unsatisfiable classes are also subclasses of two or more classes that are asserted to be disjoint. The removal of each of the subclass axioms above solves multiple disjointness axiom violations. For the first axiom that contributes to the most unsatisfiable classes, the classes it accounts for each violate one or more of these three different disjointness axioms:

- 1 'independent
(BFO:0000004) DisjointWith: 'spe-



cifically dependent continuant' (BFO:0000020)

2 DisjointClasses: 'independent continuant' (BFO:0000004), 'specifically dependent continuant' (BFO:0000020), 'generically dependent continuant' (BFO:0000031)

3 'continuant' (BFO:0000002) DisjointWith: 'occurrent' (BFO:0000003)

The second case is affected by two disjointness axioms:

1 'independent continuant' (BFO:0000004) DisjointWith: 'specifically dependent continuant' (BFO:0000020)

2 DisjointClasses: 'independent continuant' (BFO:0000004), 'specifically

dependent continuant' (BFO:0000020),
'generically dependent continuant'
(BFO:0000031)

The two disjointness axioms shown for the second case are included in the three axioms shown for the first set, and the disjointness axiom between independent continuant and specifically dependent continuant is a consequence of the others. In total, therefore, three disjointness axioms account for all cases of hidden unsatisfiability throughout the OBO Foundry ontologies. Removing the subclass axioms removes fewer axioms and solves the cases of unsatisfiability because they prevent classes from violating multiple disjointness axioms. For example, in the case of removing the subclass relationship between molecular entity (CHEBI:22367) and material entity (BFO:0000040), some subclasses of 'molecular entity' violate the first disjointness axiom and some violate the second. By removing the subclass axiom, however, molecular entities are no longer subclasses of the parent class of material entity, independent continuant (BFO:0000004), for which two disjointness axioms are asserted.

Among the wider set of OBO ontologies we found that a set of only 117 axioms could be removed from ontologies to solve all unsatisfiability for all 866,494 unsatisfiable classes. Of these, 51 involved a BFO class. Table 3 shows the top ten axioms ranked by the number of unique unsatisfiable classes they are responsible for repairing when removed, while the full set of axioms is available in the Github repository associated with this experiment.

Inconsistency analysis

Our experiments identify contradictions that lead to unsatisfiable classes in the OBO ontologies and highlight the axioms that can be removed to solve most cases of unsatisfiability. Our experiments further reveal which disjointness axioms are most frequently violated. However, merely removing the axioms does not necessarily resolve the underlying issues in how domain knowledge is modeled.

For example, although 599 unsatisfiable classes are repaired in OBO Foundry ontologies by removing the subclass axiom, 'realizable entity' (BFO:0000017) SubClassOf: 'specifically dependent continuant' (BFO:0000020), this does not entail that this axiom, or the disjointness axioms it is related to, are themselves incorrect. Instead, the unsatisfiable classes arise through the different, mutually exclusive, uses of these classes by more specific axioms. In particular, 87 of these 599 classes are MAP kinase activity (GO:0004707) and its subclasses. The violated disjointness axiom is the fundamental BFO distinction between continuant (BFO:0000002) and occurrent (BFO:0000003). A continuant is something that is present as a whole at a time point and maintains its identity over time while an occurrent unfolds through time and has temporal parts [33]. They are often used in biomedical ontologies to refer to material entities and processes, respectively.

As shown in Fig. 4, MAP kinase activity is a subclass of continuant (indirectly through several other classes) by means of being a molecular function. It is also a subclass of part of some MAPK cascade, which is a subclass of intracellular signal transduction. This class stands in an

Table 3 Top ten axioms accounting for the most hidden cases of unsatisfiability across OBO ontologies

Axiom	Class count
'processual entity' (UBERON:0000000) DisjointWith: 'anatomical entity' (UBERON:0001062)	102,501
'anatomical entity' (UBERON:0001062) SubClassOf: 'processual entity' (UBERON:0000000)	63,349
miRNA_target_gene_primary_transcript (NCRO:0000001) SubClassOf: nc_primary_transcript (SO:0000483)	59,887
'has role' (RO:0000087) Range: role (BFO:0000023))	57,438
'processual entity' (UBERON:0000000) SubClassOf: 'occurrent' (BFO:0000003)	41,770
'continuant' (BFO:0000002) DisjointWith: 'occurrent' (BFO:0000003)	31,943
'connected anatomical structure' (CARO:0000003) SubClassOf: 'material anatomical entity' (CARO:0000006)	31,639
'independent continuant' (BFO:0000004) DisjointWith: 'specifically dependent continuant' (BFO:0000020), 'generically dependent continuant' (BFO:0000031)	30,203
'realizable entity' (BFO:0000017) SubClassOf: 'specifically dependent continuant' (BFO:0000020)	21,603
'organ' UBERON:0000062 SubClassOf: 'has 2D boundary' RO:0002002 some 'anatomical surface' (UBERON:0006984)	20,539

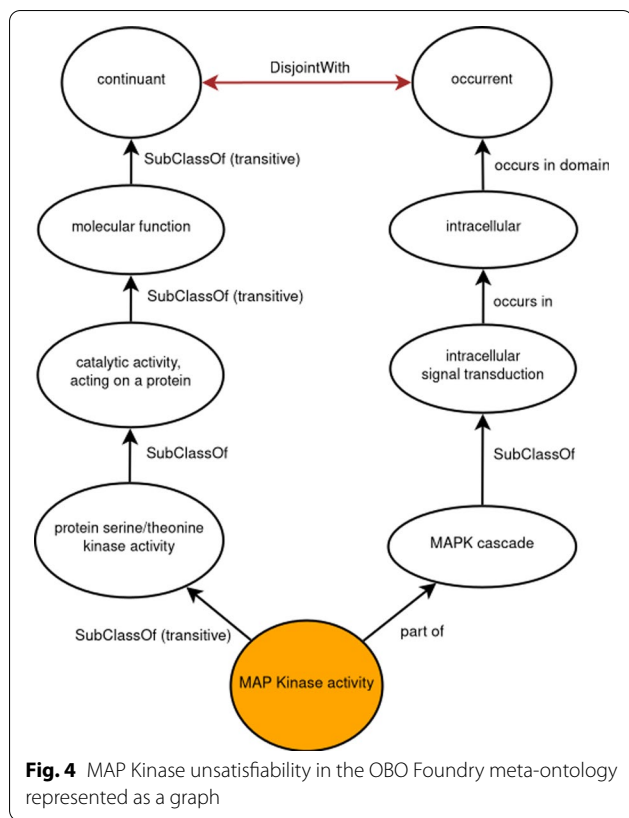


Fig. 4 MAP Kinase unsatisfiability in the OBO Foundry meta-ontology represented as a graph

occurs in relationship with intracellular. The object property occurs in contains a restriction of its domain, asserting that something that occurs in something else must be an occurrent. Consequently, MAPK cascade, a kind of intracellular signal transduction, is an occurrent.

Then, because MAP kinase activity is part of a MAPK cascade, it too is an occurrent. The reason for this is that the part of (BFO:0000050) relationship must be between two things of the same kind; its description states “two distinct things cannot be part of each other” which is enforced by assertions in the Relation Ontology (RO) that state occurrent is a subclass of part of only occurrent, and continuant is a subclass of part of only continuant. This means that the reasoner infers from the assertion that MAP kinase activity is a part of MAPK cascade, that it too must be an occurrent. Therefore, MAP kinase activity must be both a continuant and an occurrent, which is the source of its unsatisfiability.

In addition to the 87 classes that due to the axioms related to MAP kinase activity, all 599 unsatisfiable classes that can be removed by removing the 'realizable entity' (BFO:0000017) SubClassOf 'specifically dependent continuant'

(BFO:0000020) axiom are subclasses of the class description:

- 'molecular_function' and 'occurs in' some 'intracellular'

This is fundamentally the same cause of unsatisfiability as MAP kinase activity: that is they are subclasses of continuant via molecular_function, and occurrent via being something or a part of something that occurs in intracellular. There are actually 1306 total classes which are subclasses of 'occurs in' some 'intracellular', but 707 of these are not subclasses of continuant and are therefore not unsatisfiable.

These contradictions are not revealed by an automated reasoner used with the Gene Ontology alone, because the Gene Ontology imports occurs in (BFO:0000066) from the Relation Ontology using MIREOT, without the axioms of the Relation Ontology. Consequently, the axiom that asserts the domain of occurs in is not imported. The contradiction is revealed when the ontologies are combined and the imported class is extended with the restrictions declared in its original definition.

The long chain of inferences required to detect this unsatisfiability explains why it is easy for an ontology developer to assert a contradictory axiom, especially when the full set of axioms is not available to a reasoner during ontology development. The shared inheritance of continuant and occurrent are hidden behind several layers of subclass axioms and domain and range restrictions on object properties. Furthermore, colloquially, there may also be occasional confusion between parthood and participation in a process[34]. The problems could be fixed without any removal of axioms by using the participates in (RO:0000056) or has participant (RO:0000057) relations instead of the part of relations in some axioms[35].

Indeed, many of the axioms that were highlighted for removal imply issues deriving from improper use of BFO. For example, in the OBO ontologies experiment, 57,438 classes were made satisfiable by removing the restriction that the role a class has must be a kind of role.

All tools described in this paper, including those to obtain, merge, analyse, and repair ontologies, as well as the full results of the experiment, and tools to recreate the experiment, are available at <https://github.com/bio-ontology-research-group/UNMIREOT>.

Discussion

We have identified a high prevalence of hidden unsatisfiability throughout a major biomedical ontology ecosystem, the OBO ontologies. These ontologies include

widely used ontologies that form a crucial part of the bioinformatics infrastructure. We also developed a novel algorithm that can diagnose incoherent ontologies by identifying a small set of axioms that resolve all cases of unsatisfiability. We demonstrated this across the OBO Foundry, and found that relatively few axioms can be removed to resolve all unsatisfiable classes. Nevertheless, the fact that many of the axioms removed belong to BFO, the upper-level ontology that most OBO ontologies use as a foundation, indicates that this ontology is not used consistently throughout all ontologies. Also of note is that several ontologies were inconsistent when combined with the set of OBO Foundry ontologies. These ontologies likely had similar problems to the other ontologies we examined, but actually included instances of the unsatisfiable classes—turning an incoherent ontology into an inconsistent one. Our algorithm reveals that it suffices to remove or change 117 axioms to repair all issues we identified; while our algorithm can automatically remove these axioms, the number of problematic axioms is small enough for them to be manually investigated; this sets out a way forward towards a logically consistent set of biomedical ontologies.

We extended our previous work, which examined one ontology for cases of hidden unsatisfiability caused by classes it referenced [17]. The tool can now be used to evaluate any ontology for hidden unsatisfiability, having been generalised such that it can combine and evaluate any group of arbitrary ontologies specified (in this case, those defined by the OBO registries). The other major extension to the work is in the novel algorithm allowing for the derivation of a small set of axioms accounting for all cases of unsatisfiability in very large ontologies, while the previous tool only counted unsatisfiable classes. The tool also includes scripts for automating hidden unsatisfiability checking for groups of ontologies.

In the previous work, we discussed several limitations of the tool. Foremost, that it relied on the ontology being included in the AberOWL ontology repository. This is no longer the case, as the tool simply operates on downloaded files (or for the OBO experiment, downloads files directly from the links provided in the registry). It also mentioned that the unMIREOT step does not work transitively, however this limitation is not relevant in the context of the change to working with provided sets of ontologies to combine.

While our algorithm removes a small set of axioms to make an ontology coherent, it does not repair the root cause of the contradiction. In one case we showed that a large number of unsatisfiable classes in the Gene Ontology were caused by a mistaken use of a parthood relationship. This cause for unsatisfiability was complex, but would have been revealed by an automated reasoner

had the axioms of MIREOT-ed classes been included. This indicates that the unconstrained use of MIREOT has introduced a new challenge for ontology interoperability, which must now be addressed. The question remains, however, of how best to balance the challenges of developing ontologies with the hardware resources and tools available, while at the same time maintaining consistency and interoperability between ontologies. Our results illustrate how the unMIREOT tool can be used to help ontology developers identify problematic axioms in their ontologies, and explore them to diagnose causes of contradiction.

While we have shown that there are large clusters of unsatisfiability across the OBO Foundry, it is unclear whether or to what extent these issues are affecting ontology-based analysis techniques. Incorrect inferences could affect the results of gene enrichment analysis, inter-ontology phenotype mapping, semantic similarity tasks, or any analysis that relies on ontology axiomatisation. In the future, we intend to explore this by implementing a reference task that relies on multiple combined ontologies, and comparing the performance before and after repairing the unsatisfiable classes.

While cases of unsatisfiability can be repaired by the unMIREOT approach, it does not actually repair the root causes of disagreements. While examination of the output can help to identify the root cause, this can still be a time consuming and complicated process. It is possible that algorithmic tools could be developed to aid ontology developers in identifying the most informative cause of the inconsistency, or instead to create a set of minimally destructive axioms to remove from the ontologies.

One approach to preventing contradictions from entering ontology releases in the future is the use of full ontology inclusion and testing during the development process, as part of an integration testing process. It would be possible to incorporate the unMIREOT tool in such a workflow or ontology release tool [36]. The OBO ontologies use a shared central build system which can be configured to validate ontologies against scripts that check for problems. By using a powerful build server to combine ontologies with the ontologies they refer to and check for inconsistencies before release, developers would be able to continue to use MIREOT while ensuring continuing compatibility.

It is also possible that either the MIREOT or OBO guidelines should be revised, to include more information in a class reference. Including more axioms related to referenced classes would allow for local consistency checking with an automated reasoner. Because many axioms are inherited, and restrictions are placed transitively, the axioms of an entire ontology or at least a derived module would need to be imported. This could

be recommended in the case of small, high-level ontologies such as BFO and RO, which should not cause performance or space issues. Without actually including the ontology in the imports closure, however, it would not solve the problem of sourcing ontologies becoming out of date with the ontologies they reference.

Since the study described in this article was performed in 2018, we re-ran the OBO Foundry component of the experiment on updated ontologies downloaded on 07/07/2020. We found that there remained 39 unsatisfiable classes in the OBO Foundry, accounted for by one axiom. Furthermore, we combined a repaired version of the updated OBO Foundry meta-ontology with the recently published Coronavirus Infectious Disease Ontology (CIDO) [37]. The unMIREOT tool discovered 8,346 unsatisfiable classes in this combination, accounted for by 7 axioms. These results are available on the unMIREOT GitHub repository.

Conclusions

We identified a large set of hidden unsatisfiability across a broad range of biomedical ontologies, and we found that this large set of unsatisfiable classes is the result of a relatively small amount of axiomatic disagreements. Our results show that hidden unsatisfiability is a serious problem in ontology interoperability; however, our results also provide a way towards more consistent ontologies by addressing the issues we identified.

Abbreviations

OBO: Open biomedical ontologies; OWL: Web Ontology Language; BFO: Basic Formal Ontology; HTN: Hypertension Ontology; HP: Human Phenotype Ontology; DO: Disease Ontology; MIREOT: Minimum Information to Reference an External Ontology Term; IRI: Internationalised Resource Identifier; EFO: Experimental Factor Ontology; OVAE: Ontology of Vaccine Adverse Events; FOODON: Food Ontology; TO: Plant Trait Ontology; GAZ: Gazetteer; PORO: Porifera; PECO: Plant Experimental Conditions Ontology; OHD: Oral Health and Disease Ontology; STATO: Statistics Ontology; UPHENO: Unified Phenotype Ontology; HST: Hitting Set Tree; GO: Gene Ontology; RO: Relation Ontology; CIDO: Coronavirus Infectious Disease Ontology; CHEBI: Chemical Entities of Biological Interest; OBI: Ontology for biomedical investigations; PATO: Phenotypic quality ontology; PO: Plant ontology; XAO: Xenopus Anatomy and Development Ontology; ZFA: Zebrafish Anatomy and Development Ontology; MAP: Mitogen-activated protein; MONDO: Monarch Disease Ontology; OMIT: Ontology for MIRNA Target; MOP: Molecular Process Ontology; RXNO: Name Reaction Ontology; MP: Mammalian Phenotype Ontology; CL: Cell Ontology; OBA: Ontology of Biological Attributes; OAE: Ontology of Adverse Events.

Acknowledgements

We would like to thank Dr Paul Schofield and Dr Egon Willighagen for helpful conversations surrounding the described experiments. A pre-print of this article appeared at <https://www.biorxiv.org/content/10.1101/2020.05.16.099309v1>.

About this supplement

This article has been published as part of BMC Medical Informatics and Decision Making, Volume 20 Supplement 10 2020: Quality Assurance and Enrichment of Biological and Biomedical Ontologies and Terminologies. The full contents of the supplement are available at <https://bmcmmedinformdecismak.biomedcentral.com/articles/supplements/volume-20-supplement-10>.

Authors' contribution

RH and LTS conceived of the study and experimental design. LTS performed all experiments and implemented the software. LTS drafted the manuscript. RH and GVG supervised the project. All authors revised and approved the manuscript for submission.

Funding

Publication costs were funded by Nanocommons H2020-EU (731032). GVG and LTS acknowledge support from support from the NIHR Birmingham ECMC, NIHR Birmingham SRMRC, Nanocommons H2020-EU (731032) and the NIHR Birmingham Biomedical Research Centre and the MRC HDR UK (HDRUK/CFC/01), an initiative funded by UK Research and Innovation, Department of Health and Social Care (England) and the devolved administrations, and leading medical research charities. The views expressed in this publication are those of the authors and not necessarily those of the NHS, the National Institute for Health Research, the Medical Research Council or the Department of Health.

Availability of data and materials

The datasets generated and/or analysed during the current study are available in the UNMIREOT GitHub repository, <http://github.com/bio-ontology-research-group/UNMIREOT>.

Ethics approval and consent to participate

Not applicable, as this study did not use human material or data.

Consent for publication

Not applicable.

Competing interests

The authors declare that they have no competing interests.

Author details

¹ College of Medical and Dental Sciences, Institute of Cancer and Genomic Sciences, University of Birmingham, Birmingham B15 2TT, UK. ² Institute of Translational Medicine, University Hospitals Birmingham, NHS Foundation Trust, Birmingham B15 2TT, UK. ³ NIHR Experimental Cancer Medicine Centre, Birmingham B15 2TT, UK. ⁴ NIHR Surgical Reconstruction and Microbiology Research Centre, Birmingham B15 2TT, UK. ⁵ NIHR Biomedical Research Centre, Birmingham B15 2TT, UK. ⁶ MRC Health Data Research UK (HDR UK Midlands, Birmingham B15 2TT, UK. ⁷ Computer, Electrical and Mathematical Sciences and Engineering Division, Computational Bioscience Research Center, King Abdullah University of Science and Technology, Thuwal 23955, Saudi Arabia.

Received: 11 November 2020 Accepted: 16 November 2020

Published: 15 December 2020

References

- McGuinness DL, Van Harmelen F. OWL web ontology language overview. *W3C Recomm.* 2004;10(10):2004.
- Baader F, Calvanese D, McGuinness DL, Nardi D, Patel-Schneider PF, editors. *The description logic handbook: theory, implementation and applications*. 2nd ed. Cambridge: Cambridge University Press; 2007. <https://doi.org/10.1017/CBO9780511711787>.
- Smith B, Ashburner M, Rosse C, Bard J, Bug W, Ceusters W, Goldberg LJ, Eilbeck K, Ireland A, Mungall CJ, Leontis N, Rocca-Serra P, Ruttenberg A, Sansone S-A, Scheuermann RH, Shah N, Whetzel PL, Lewis S, The OBI Consortium. The OBO foundry: coordinated evolution of ontologies to support biomedical data integration. *Nat Biotechnol.* 2007;25(11):1251–5. <https://doi.org/10.1038/nbt1346>.
- Arp R, Smith B, Spear AD. *Building ontologies with basic formal ontology*. Cambridge: The MIT Press; 2015.
- de Matos P, Dekker A, Ennis M, Hastings J, Haug K, Turner S, Steinbeck C. ChEBI: a chemistry ontology and database. *J Cheminformatics.* 2010;2(S1):6.
- Köhler S, Doelken SC, Mungall CJ, Bauer S, Firth HV, Bailleul-Forestier I, Black GCM, Brown DL, Brudno M, Campbell J, FitzPatrick DR, Eppig JT, Jackson AP, Freson K, Girdea M, Helbig I, Hurst JA, Jähn J, Jackson LG, Kelly AM, Ledbetter DH, Mansour S, Martin CL, Moss C, Mumford A, Ouwehand

- WH, Park S-M, Riggs ER, Scott RH, Sisodiya S, Vooren SV, Wapner RJ, Wilkie AOM, Wright CF, Vulto-van Silfhout AT, de Leeuw N, de Vries BBA, Washington NL, Smith CL, Westerfield M, Schofield P, Ruef BJ, Gkoutos GV, Haendel M, Smedley D, Lewis SE, Robinson PN. The human phenotype ontology project: linking molecular biology and disease through phenotype data. *Nucleic Acids Res.* 2014;42(Database issue):966–74. <https://doi.org/10.1093/nar/gkt1026>.
7. Smith CL, Goldsmith C-AW, Eppig JT. The Mammalian Phenotype Ontology as a tool for annotating, analyzing and comparing phenotypic information. *Genome Biol.* 2004;6(1):7. <https://doi.org/10.1186/gb-2004-6-1-r7>.
 8. Schorlemmer M, Kalfoglou Y. Using information-flow theory to enable semantic interoperability. Edinburgh: The University of Edinburgh; 2003. p. 3.
 9. Hicks A, Miller MA, Stoeckert C, Mowery D. The Hypertension Ontology. 2019. <https://doi.org/10.5281/zenodo.2605329>.
 10. Schriml LM, Arze C, Nadendla S, Chang Y-WW, Mazaitis M, Felix V, Feng G, Kibbe WA. Disease ontology: a backbone for disease semantic integration. *Nucleic Acids Res.* 2012;40(D1):940–6. <https://doi.org/10.1093/nar/gkr972>.
 11. Del Vescovo C, Gessler DDG, Klinov P, Parsia B, Sattler U, Schneider T, Winget A. Decomposition and modular structure of bioportal ontologies. In: Aroyo L, Wely C, Alani H, Taylor J, Bernstein A, Kagal L, Noy N, Blomqvist E, editors. The semantic web—ISWC 2011. Lecture notes in computer science. Berlin: Springer; 2011. p. 130–45. https://doi.org/10.1007/978-3-642-25073-6_9.
 12. Grau BC, Parsia B, Sirin E, Kalyanpur A. Modularizing OWL ontologies. In: K-CAP 2005 workshop on ontology management; 2005.
 13. Courtot M, Gibson F, Lister AL, Malone J, Schober D, Brinkman RR, Rutenberg A. MIREOT: the minimum information to reference an external ontology term. *Appl Ontol.* 2011;6(1):23–33. <https://doi.org/10.3233/AO-2011-0087>.
 14. Smart PR, Engelbrecht PC. An analysis of the origin of ontology mismatches on the semantic web. In: Gangemi A, Euzenat J, editors. Knowledge engineering: practice and patterns. Lecture notes in computer science. 120th ed. Berlin: Springer; 2008. p. 135. https://doi.org/10.1007/978-3-540-87696-0_13.
 15. Hoehndorf R, Dumontier M, Oellrich A, Rebholz-Schuhmann D, Schofield PN, Gkoutos GV. Interoperability between Biomedical Ontologies through relation expansion, upper-level ontologies and automatic reasoning. *PLoS ONE.* 2011;. <https://doi.org/10.1371/journal.pone.0022006>.
 16. Malone J, Adamusiak T, Holloway E, Parkinson H. Developing an application ontology for annotation of experimental variables—experimental factor ontology. *Nat Proceed.* 2009;. <https://doi.org/10.1038/npre.2009.3806.1>.
 17. Slater L, Gkoutos GV, Schofield PN, Hoehndorf R. To MIREOT or not to MIREOT? A case study of the impact of using MIREOT in the experimental factor ontology (EFO). In: ICBO/BioCreative; 2016.
 18. Horridge M, Bechhofer S. The OWL API: a java API for OWL ontologies. *Semant Web.* 2011;2(1):11–21. <https://doi.org/10.3233/SW-2011-0025>.
 19. Kazakov Y, Krötzsch M, Simančík F. The incredible ELK. *J Autom Reason.* 2014;53(1):1–61. <https://doi.org/10.1007/s10817-013-9296-3>.
 20. Noy NF, Crubézy M, Ferguson RW, Knublauch H, Tu SW, Vendetti J, Musen MA. Protégé-2000: an open-source ontology-development and knowledge-acquisition environment. *AMIA Annu Symp Proc.* 2003;2003:953.
 21. Marcos E, Zhao B, He Y. The Ontology of Vaccine Adverse Events (OVAE) and its usage in representing and analyzing adverse events associated with US-licensed human vaccines. *J Biomed Semant.* 2013;4:40. <https://doi.org/10.1186/2041-1480-4-40>.
 22. Dooley DM, Griffiths EJ, Gosal GS, Buttigieg PL, Hoehndorf R, Lange MC, Schriml LM, Brinkman FSL, Hsiao WWL. FoodOn: a harmonized food ontology to increase global food traceability, quality control and data integration. *NPJ Sci Food.* 2018;. <https://doi.org/10.1038/s41538-018-0032-6>.
 23. Cooper L, Meier A, Laporte M-A, Elser JL, Mungall C, Sinn BT, Cavaliere D, Carbon S, Dunn NA, Smith B, Qu B, Preece J, Zhang E, Todorovic S, Gkoutos G, Doonan JH, Stevenson DW, Arnaud E, Jaiswal P. The Plantome database: an integrated resource for reference ontologies, plant genomics and phenomics. *Nucleic Acids Res.* 2018;46:1168–80. <https://doi.org/10.1093/nar/gkx1152>.
 24. Team F. FAIRsharing record for: Gazetteer. FAIRsharing; 2015. <https://doi.org/10.25504/FAIRSHARING.WKDJPB>.
 25. Thacker RW, Díaz MC, Kerner A, Vignes-Lebbe R, Segerdell E, Haendel MA, Mungall CJ. The Porifera Ontology (PORO): enhancing sponge systematics with an anatomy ontology. *J Biomed Semant.* 2014;5(1):39. <https://doi.org/10.1186/2041-1480-5-39>.
 26. Schleyer TK, Rutenberg A, Duncan W, Haendel M, Torniai C, Acharya A, Song M, Thyvalikakath TP, Liu K, Hernandez P. An ontology-based method for secondary use of electronic dental record data. *AMIA Summits Transl Sci Proc.* 2013:234–238.
 27. Team F. FAIRsharing record for: statistics ontology. FAIRsharing; 2015. <https://doi.org/10.25504/FAIRSHARING.NA5XP>.
 28. Kalyanpur A, Parsia B, Sirin E, Cuenca-Grau B. Repairing unsatisfiable concepts in OWL ontologies. In: Sure Y, Domingue J, editors. The semantic web: research and applications. Lecture notes in computer science. Berlin: Springer; 2006. p. 170–84. https://doi.org/10.1007/11762256_15.
 29. Reiter R. A theory of diagnosis from first principles. *Artif Intell.* 1987;32(1):57–95. [https://doi.org/10.1016/0004-3702\(87\)90062-2](https://doi.org/10.1016/0004-3702(87)90062-2).
 30. Karp R. Reducibility among combinatorial problems. *Complex Comput Comput.* 1972;40:85–103. https://doi.org/10.1007/978-3-540-68279-0_8.
 31. Cook SA. The complexity of theorem-proving procedures. In: Proceedings of the third annual ACM symposium on theory of computing. STOC '71, pp. 151–158. Association for Computing Machinery, Shaker Heights, Ohio, USA; 1971. <https://doi.org/10.1145/800157.805047>.
 32. Kalyanpur A, Parsia B, Horridge M, Sirin E. Finding all justifications of OWL DL entailments. In: Aberer K, Choi K-S, Noy N, Allemang D, Lee K-I, Nixon L, Golbeck J, Mika P, Maynard D, Mizoguchi R, Schreiber G, Cudré-Mauroux P, editors. The semantic web. Lecture notes in computer science. Berlin: Springer; 2007. p. 267–80. https://doi.org/10.1007/978-3-540-76298-0_20.
 33. Arp R, Smith B. Function, role, and disposition in basic formal ontology. *Nat Proceed.* 2008;. <https://doi.org/10.1038/npre.2008.1941.1>.
 34. Guarino N, Wely C. An overview of ontoclean, pp. 201–220; 2009. https://doi.org/10.1007/978-3-540-92673-3_9.
 35. Smith B, Ceusters W, Klagges B, Köhler J, Kumar A, Lomax J, Mungall C, Neuhaus F, Rector AL, Rosse C. Relations in biomedical ontologies. *Genome Biol.* 2005;6(5):46. <https://doi.org/10.1186/gb-2005-6-5-r46>.
 36. Jackson RC, Balhoff JP, Douglass E, Harris NL, Mungall CJ, Overton JA. ROBOT: a tool for automating ontology workflows. *BMC Bioinform.* 2019;. <https://doi.org/10.1186/s12859-019-3002-3>.
 37. He Y, Yu H, Ong E, Wang Y, Liu Y, Huffman A, Huang H-H, Beverley J, Hur J, Yang X, Chen L, Omenn GS, Athey B, Smith B. CIDO, a community-based ontology for coronavirus disease knowledge and data integration, sharing, and analysis. *Sci Data.* 2020;7(1):181. <https://doi.org/10.1038/s41597-020-0523-6>.
 38. Ashburner M, Ball CA, Blake JA, Botstein D, Butler H, Cherry JM, Davis AP, Dolinski K, Dwight SS, Eppig JT, Harris MA, Hill DP, Issel-Tarver L, Kasarskis A, Lewis S, Matese JC, Richardson JE, Ringwald M, Rubin GM, Sherlock G. Gene ontology: tool for the unification of biology. *Nat Genet.* 2000;25(1):25–9. <https://doi.org/10.1038/75556>.
 39. Bandrowski A, Brinkman R, Brochhausen M, Brush MH, Bug B, Chibucos MC, Clancy K, Courtot M, Derom D, Dumontier M, Fan L, Fostel J, Fragoso G, Gibson F, Gonzalez-Beltran A, Haendel MA, He Y, Heiskanen M, Hernandez-Boussard T, Jensen M, Lin Y, Lister AL, Lord P, Malone J, Manduchi E, McGee M, Morrison N, Overton JA, Parkinson H, Peters B, Rocca-Serra P, Rutenberg A, Sansone S-A, Scheuermann RH, Schober D, Smith B, Soldatova LN, Stoeckert CJ, Taylor CF, Torniai C, Turner JA, Vita R, Whetzel PL, Zheng J. The ontology for biomedical investigations. *PLoS ONE.* 2016;. <https://doi.org/10.1371/journal.pone.0154556>.
 40. Gkoutos GV, Mungall C, Dolken S, Ashburner M, Lewis S, Hancock J, Schofield P, Kohler S, Robinson PN. Entity/quality-based logical definitions for the human skeletal phenotype using PATO. In: 2009 annual international conference of the IEEE engineering in medicine and biology society, 2009, pp 7069–7072. <https://doi.org/10.1109/IEMBS.2009.5333362>.
 41. Jaiswal P, Avraham S, Ilic K, Kellogg EA, McCouch S, Pujar A, Reiser L, Rhee SY, Sachs MM, Schaeffer M. Plant ontology (PO): a controlled vocabulary of plant structures and growth stages. *Comp Funct Genom.* 2005;6(7–8):388–97.
 42. Segerdell E, Bowes JB, Pollet N, Vize PD. An ontology for *Xenopus* anatomy and development. *BMC Dev Biol.* 2008;8(1):92.
 43. Sprague J, Bayraktaroglu L, Clements D, Conlin T, Fashena D, Frazer K, Haendel M, Howe DG, Mani P, Ramachandran S. The Zebrafish

- information network: the zebrafish model organism database. *Nucleic Acids Res.* 2006;34(suppl-1):581–5.
44. Köhler S, Doelken SC, Ruef BJ, Bauer S, Washington N, Westerfield M, Gkoutos G, Schofield P, Smedley D, Lewis SE, Robinson PN, Mungall CJ. Construction and accessibility of a cross-species phenotype ontology along with gene annotations for biomedical research. *F1000Research.* 2014;2:30. <https://doi.org/10.12688/f1000research.2-30.v2>.
 45. Shefchek KA, Harris NL, Gargano M, Matentzoglou N, Unni D, Brush M, Keith D, Conlin T, Vasilevsky N, Zhang XA, Balhoff JP, Babb L, Bello SM, Blau H, Bradford Y, Carbon S, Carmody L, Chan LE, Cipriani V, Cuzick A, Rocca MD, Dunn N, Essaid S, Fey P, Grove C, Gourdine J-P, Hamosh A, Harris M, Helbig I, Hoatlin M, Joachimiak M, Jupp S, Lett KB, Lewis SE, McNamara C, Pendlington ZM, Pilgrim C, Putman T, Ravanmehr V, Reese J, Riggs E, Robb S, Roncaglia P, Seager J, Segerdell E, Similuk M, Storm AL, Thaxon C, Thesen A, Jacobsen JOB, McMurry JA, Groza T, Köhler S, Smedley D, Robinson PN, Mungall CJ, Haendel MA, Munoz-Torres MC, Osumi-Sutherland D. The Monarch Initiative in 2019: an integrative data and analytic platform connecting phenotypes to genotypes across species. *Nucleic Acids Res.* 2020;48(D1):704–15. <https://doi.org/10.1093/nar/gkz997>.
 46. Huang J, Tan M, Dou D, He L, Townsend C, Hayes PJ. Ontology for micro-RNA target prediction in human cancer. In: Proceedings of the First ACM international conference on bioinformatics and computational biology; 2010. p. 472–474.
 47. Ramos L, Gil R, Anastasiou D, Martin-Bautista MJ. Towards a machine of a process (MOP) ontology to facilitate e-commerce of industrial machinery. *Comput Ind.* 2014;65(1):108–15.
 48. Sankar P, Aghila G. Design and development of chemical ontologies for reaction representation. *J Chem Inf Model.* 2006;46(6):2355–68.
 49. Bard J, Rhee SY, Ashburner M. An ontology for cell types. *Genome Biol.* 2005;6(2):21.
 50. Dönitz J, Wingender E. The ontology-based answers (OBA) service: a connector for embedded usage of ontologies in applications. *Front Genet.* 2012;3:197.
 51. He Y, Sarntivijai S, Lin Y, Xiang Z, Guo A, Zhang S, Jagannathan D, Toldo L, Tao C, Smith B. OAE: the ontology of adverse events. *J Biomed Semant.* 2014;5(1):29.

Publisher's Note

Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Ready to submit your research? Choose BMC and benefit from:

- fast, convenient online submission
- thorough peer review by experienced researchers in your field
- rapid publication on acceptance
- support for research data, including large and complex data types
- gold Open Access which fosters wider collaboration and increased citations
- maximum visibility for your research: over 100M website views per year

At BMC, research is always in progress.

Learn more biomedcentral.com/submissions

