What Causes Pneumonia?
The Case for a Standard Semantics for "may" in OWL

Alan Rector, Robert Stevens, Nick Drummond

School of Computer Science, University of Manchester
rector@cs.manchester.ac.uk

Abstract: One of users' most frequent questions about OWL is "how do I say 'may'?", as in "Bacteria may cause pneumonia." In many fields, particularly biomedicine, a high proportion of knowledge that users wish to capture in ontologies is of this form. The issue is closely related to dealing with fillers other existential restrictions, most obviously in partonomies: "All cars have engines", therefore "Engines may be parts of cars", but not "All engines are part of cars." There is no standard mechanism in the OWL language specification, API, reasoners, or query languages for dealing with these issues. In the absence of a standard, users have found various work arounds – some partial solutions, some clearly incorrect. In this paper we examine the intuitions to be captured and four such work arounds. We then sketch an approach to extending OWL and associated query languages that is a reasonable approximation to the intuitions, provides standard semantics so that there are standard answers to queries, and falls within the capability of existing reasoners, although it would require extensions to APIs and tools.

1. Introduction

One of users' most frequent questions about OWL is "How do I say 'may'?", as in "People may own dogs" or "Bacteria may cause pneumonia." In many fields, particularly biomedicine, a high proportion of knowledge that users wish to capture in ontologies is of this form. Users may be unclear on the precise semantics, but ultimately, they wish answers to queries such as "What do people own? or "What causes pneumonia?".

The OWL language standard, associated query languages, APIs and Reasoners do not provide an answer to these questions. Some potential users have cited their absence a prime reason for not adopting OWL. Other users have found "work arounds" that are at best non-standard and at worst lead to incorrect results if extended.

A closely related question arises for the reciprocals for existential restrictions. It is most graphically illustrated by partonomies. "All cars have (some) engine(s)"; but "not all engines are parts of cars". Therefore, intuitively, "Engines *may* be parts of cars."[1] What then should be the answer to a query about the parts of cars? There is a draft working note from the SW Best Practices Working Group [1], but the work-arounds presented are at best problematic. The most widely used is to require false axioms to get the desired answers – *e.g.* that all engines are parts of cars – so that the correct answers to the query "What are parts of cars" is obtained.

Since users do not have a single semantics for these questions, we focus the discussion by asking first what the answers, intuitively, users expect to queries involving "may", and then what semantics we might assign to them to achieve these results in OWL.

Finally, we want to differentiate this issue from two closely related questions that might be solved by related means: "sanctioning" / "property attachment" and "typical" / "defaults with exceptions". All three are badly needed by users, but conflating them makes it more difficult to specify solutions.

Although we proffer a proposal, our prime purpose is to set out requirements and raise an issue with the community which is a persistent source of frustration to our users, a critical barrier to some, and which has resulted in users inventing various "work-arounds", some useful, some dubious, and some incorrect.

Note on notation: For brevity and clarity we use the Manchester OWL Syntax throughout. [2]

2. Intuition to be captured

Assuming we have a knowledge base that somehow represents:

- *"People may own dogs"*
- *"Bacteria may cause pneumonia"*
- *"Cars have (parts) engines"*

What answers do we want to the queries:

- *"What do people own?"*
- *"What causes pneumonia"*
- *"What are the parts of cars"*

Our contention is that the sensible answers in each case are:

- *"Dogs that are owned by people"* – since not all dogs are owned, which might be logically equivalent to a named entity, *e.g.* "Pet dogs".
- *"Bacteria that cause pneumonia"* – since not all bacteria, not even all bacteria of very pathogenic types, cause pneumonia.
- *"Car engines"* – since there are many other kinds of engines.

We do not want to demand that there be individuals in any of these categories in any given ontology nor that all possible such categories be explicitly defined and named, but we do wish to guarantee that the categories could have instances without causing a contradiction – *i.e.* that the corresponding classes be satisfiable.

[1] The trivial case where there are no cars is not one that would occur to users, and they would regard the answers that follow from it as irrelevant, or simply wrong.

We want the entailment that, if somebody does own something, then they may own it, but not the converse. (Likewise, we would like the contrapositive, to say that if somebody may not own something, then they do not own it.)

We want to capture the notion that there is something special about the classes involved. An appropriate answer to “What causes pneumonia?” is not “Living things”, even though bacteria are living things. (An answer on a medical exam of “Living things” would be marked wrong.) An appropriate answer to “What are the parts of cars” is not “machines”, even though engines are machines. Hence, the simplistic interpretation of first order logic axioms following the pattern to “∃ x y. Pneumonia(x) & Bacteria(y) & causes(x, y)” is insufficient on its own, since this also implies “∃ x y. Pneumonia(x) & Living_thing(y) & causes(x,y)”[2]. In fact, we contend that the fundamental problem is that the intuitive meanings that are mission critical to users cannot be captured in a strictly first order formalism.

The problem is that we wish to say that there is something special about the classes “Person”, “Bacteria”, and “Engine” themselves that goes beyond what we can say about their members.

The answers also need to be reciprocal even though existential restrictions in description logics are not reciprocal. For example, if people may own dogs, then dogs may be owned by people; if bacteria may cause pneumonia, then pneumonia may be caused by bacteria. Furthermore, we would argue that if “all cars have parts engines” then “engines may be part of cars”, *i.e.* that an existential restriction implies a reciprocal “may” statement.

A more complex question is what the answer to the queries about subclasses should be, *e.g.* “What dogs may children own?” or “What streptococci cause focal pneumonia?” or “What parts do Ferraris have?” To these queries we would suggest that the intuitive answers remains “Dogs owned by children”, “Streptococci that cause focal pneumonia” and “Ferrari engines”, respectively. However, except in the last case we would not be surprised to get back the answer “none” or even “unsatisfiable”. For example, there might be laws against children owning anything, or focal pneumonia might be caused only by staphylococci. In the first two cases we have said nothing about “All pneumonia” or “all people”, whereas in the third, we have said something about “all cars”. We describe this characteristic of as “restricted inheritance” – *i.e.* that these statements should be inherited as long as the result is satisfiable.

In summary our intuitions are that a construct for “p may C”, where C and D are classes and p is a property, should capture:

- Be locally satisfiability – it only makes sense to say “p may C” if “p some C” is satisfiable.
- Be strong enough to give sensible answers to queries involving the property “p” and the construct “p may”.
- Not be too strong so as to return the unrestricted class C or give other unintended entailments.
- Be reciprocal: “C → p may D” must entail that “D → inv(p) may C” and “C → p some D” must entail that “D → inv(p) may C”.
- Capture the entailment that “some implies may” but not conversely.

[2] assuming, of course, that ∀ x . Person(x) → Living_thing(x)

- Capture the notion of “restricted inheritance”.
- Allow us to represent partonomies correctly.

Finally, we would impose the additional desiderata:

- Be practical to implement – inference should be empirically scalable with existing classifiers
- Be reasonably intuitive and inspectable by users using a uniform query mechanism on the ontology

3. Some work-arounds we have seen and some we have used

3.1 Use of domain, range, and universal restrictions

One common work-around for this problem is to identify “People may own dogs” (incorrectly) with the universal constraint “People own only dogs”, usually in the disguise of making “Dog” the range of the property “owns”. This at least provides a place to put the notion in simple cases that a person is an acceptable value for owns. It is, however, obviously wrong. People own many things other than dogs. It does not achieve the basic entailment that “may” implies “some” – owning “only dogs” does not imply owning “some dog” – often as we have to emphasize this point to students.

For queries, we are concerned with finding classes more often than the individuals in those classes, which may or may not be present in the knowledge base. The situation is made more confusing to novice users of OWL because the axiom appears to behave differently for individuals and classes. From “People only own Dogs”, “Jim is a person”, “Jim owns Qxetzl”, and “Person owns SOME Zletxq”, we can conclude that “Qxetzl is a dog”, but not that “Zletxq is a kind of dog”. Furthermore, even though patently false, such statements often result in “trivially satisfiable” classes – *i.e.* classes that can only be satisfiable if the filler of some universal restriction has no members. Trivially satisfiable classes can be used for some time before an axiom is added from which it can be inferred the filler of the universal restriction is non-empty, thereby making the restricted class unsatisfiable.

Finally, this solution is, of course, completely unsuitable to the use in partonomies; it is obviously false to say that the only parts of cars are engines.

In summary of this work-around we can say:

- It does not guarantee local satisfiability – universal constraints can be trivially satisfied
- It is too strong – people own many things besides dogs.
- It is too weak – it does not say even that the class of “Dogs owned by person” is satisfiable, let alone non-empty
- It is not reciprocal – it says nothing about what can own a dog.
- It is completely unsuitable or partonomies.

A variant of this technique to address the complaint that the statement is too strong is to interpret “People may own dogs” as saying that the class “Dog” is a subclass of the range of the property “owns”, or a subclass of the filler for the restriction “Person owns only Filler”. This can be achieved in tools. Indeed the initial version of Protégé 3.x OWL allowed lists of items in a domain or range constraints to be taken as disjunctions and allowed users to add to these lists. Despite problems, this feature is extremely popular with users. Clearly, to implement this at a language level it would require a major departure, as it would require allowing axioms of a form something like “p.domain equivalentClass (p.domain OR Dog)”. This may be appropriate for a scripting language for manipulating OWL ontologies or as an operation in an API for OWL tools, but not for the language itself for at least two reasons: a) the definition is recursive, and b) it would make reasoning non-monotonic, since adding such an “axiom” would broaden an existing universal restriction and therefore could invalidate previously valid inferences.

3.2 Use of a super- and sub-property

Another option, which we have used successfully in simply applications, is to encode each property as a super-property and sub-property, *e.g.* to reinterpret “owns” as a super-property “may_own”, and a sub-property “owns”. This immediately achieves the entailments that owns implies may_own and not may_own implies not owns. This construct is not too strong – it says nothing to prevent people owning other things or dogs being owned by other entities. It is not too weak – it does say something about the class of people.

However, even if the “may_own” property is symmetric, it is not the case that “C may_own some D” implies “D imay_own some C”. Hence the reciprocal axiom must be made explicit so that each axiom is replaced by two: *e.g.* “Person subclassOf may_own some Dog” and “Dog subclassOf inv(may_own) some Person”.

The answer to the question: “What may people own?” therefore could come be answered by the reasoner just as any other question as set of classes including “Dog”. The interpretation however, is wrapped up in the meta-semantics of the property “may_own” – something like “have the potential to own” or BFO’s “dispositions. [3] Furthermore, what people do own is not “all dogs” but “some dogs” – *i.e.* some subclass of the class Dog.

Nonetheless, this comes closer to satisfying our requirements for “may,” but does not address the issue of parts of cars. To say that “Cars have engines” still says nothing about engines that would allow us to recognize them as potential car parts. To achieve this we have to say that all engines may be part of some car. Unfortunately, this is again too strong a statement. It just isn’t true. The large multi-ton marine diesel engine is not a potential car part!

The problem of being too strong in a different sense from our initial three criteria actually applies to the other cases as well, but less obviously. There is no reason to believe that all dogs are potential pets, or all bacteria potential pathogens. We may choose to ignore this fact for sake of a “working approximation,” but to do so is risky.

Finally, this encoding hits a pragmatic barrier because it requires reciprocal existential restrictions. Reasoning using most existing classifiers – the new Hermit [4] classifier may be an exception – explodes exponentially when faced with an ontology containing any systematic use of such existential constructs[3]. This is particularly serious in the case of taxonomic hierarchies such as parts explosions. Brief experiment will show that any attempt to build a large partonomy on the above principles results in an ontology that cannot be classified in practical time, at least by Pellet, FaCT++ and older versions of Racer. We understand that the problem is fundamental to this generation of algorithms.[4]

In summary, of the super-property sub-property work around we can say that it:

- Is locally satisfiable, "p some" implies "p may", since this is the definition of subproperties.
- Is appropriately strong, although some care is required in the meta-interpretation of the meaning of the properties of the form "may_p".
- Is too strong – in response to queries it gives the unrestricted class, *e.g.* "Bacteria", but not all bacteria cause pneumonia.
- Can be made reciprocal – at the cost of additional encoding
- Exhibits some features "restricted inheritance" naturally, since the existential restrictions are "inherited" but can be "cancelled" by universal restrictions or other inferences just as any other OWL restrictions, but a universal restriction making the existential restriction unsatisfiable for a subclass does not affect the super-property, *e.g.* not (cause some Pneumonia) does not imply (not may_cause some Pneumonia)
- Is likely to lead to ontologies where reasoning does not scale because of reciprocal existential restrictions, a problem that may be resolved by new hyper-tableau reasoners.

3.3 Explicit subclasses defined by existential restrictions

In this interpretation, the statement "People may own dogs" is interpreted as requiring the definition of the classes "People that own dogs" and "Dogs that are owned by people" which will be automatically checked by the reasoner for satisfiability, *e.g.*

```
DogOwner equivalentClass Person that owns some Dog
PersonOwnedDog equivalentClass Dog that inv(owns) some Person
```

Clearly, it is strong enough to get the specified answers to queries – in effect they queries have been "pre-answered" by the construct. It does not entail any extra inferences beyond those required, because we have explicitly entered precisely the information needed.

It is reciprocal if both classes are defined. However, note that existential definitions are satisfiable if, and only if, their reciprocal is satisfiable, *i.e.*

```
Person that owns some Dog".
```

is satisfiable if and only if

```
Dog that inv(owns) some Person
```

[3] This point is widely confirmed by experience and part of "OWL lore", but we have not been able to find a suitable citation except a mention in the draft part-whole W3C note on part-whole relations.

[4] Personal Communication, Ulrike Sattler and Ian Horrocks, 2005, 2006.

is satisfiable. So there is a sense in which this solution is naturally reciprocal and an opportunity for optimization.

Note that we do not require that the classes actually be non-empty in the ontology, only that they be satisfiable.

Unfortunately, this solution has two fatal defects:

- It is unintuitive and virtually unmanageable in practice in its raw form. It leads to a proliferation of classes, and users cannot tell which classes are present to convey "may" restrictions and which are present for other reasons.
- If multiple "may" conditions apply to one class or for non-functional properties it leads to an exponentially explosive number of subclasses: "C that p some X", "C that q some Y", "C that p some X and q some Q", etc.
- It does not satisfy "restricted inheritance", in fact, there is no inheritance at all. Any attempt to remedy this fault by providing the additional "inherited" subclasses would just lead to an exponential explosion of defined subclasses.

3.4 Overloading Min Cardinality 0 and implied defined classes

A third work-around is to encode "People may own dogs" as a pair of restrictions qualified by minCardinality(0), *e.g.*

```
Person owns min 0 Dog
Dog inv(owns) min 0 Person
```

Restrictions qualified by minCardinality(0) in OWL are tautologous. They place no restriction on the class in question and are, therefore, effectively ignored by reasoners. They correspond intuitively to many users' experience of expressions such as "0..*" or "*..0" in languages such as UML, and has the bonus of seeming a natural expression of the notion of "optional". They are "restrictions" and obey the usual rules of inheritance. The encoding as a pair of restrictions is required to make them reciprocal. Given that such statements are tautologies in the first order semantics, it seems not unreasonable to overload them with a limited higher-order semantic. We currently use this construct in this way in a number of ontologies.

As it stands, these statements are too weak. Nothing follows from them; they are not checked to see if the corresponding existential restriction is satisfiable, and it is only possible to answer question about them by syntactic queries.

However, these defects could be rectified if they were taking as implying the satisfiability of the relevant defined subclasses, and if query languages were extended so that the relevant defined subclasses were returned automatically. That is, if it were part of the language specification that, for any class C, class expression D, and property p, axioms of the form:

```
C subclass p min 0 D
```

were only considered "proper" if the corresponding existential restriction were satisfiable, *i.e.* if "C and p some D" were satisfiable, and "improper" axioms, or classes with improper restrictions, were automatically flagged similarly to unsatisfiable classes.

Correspondingly, query languages such as SPARQL-DL would need to be extended so that there is some in response to "owns some Dog", they return "Person that owns some Dog" (or an equivalent named class if one has been defined in the ontology). If the query returns a hierarchy, the hierarchy should include the corresponding subclasses provided they are satisfiable, *e.g.* in our earlier example "Adult that owns some Dog" but not "Child that owns some Dog" (assuming that it can be inferred from the ontology that no child can own anything). It should not, of course, expand all the combinatorially many possibilities implied by "may" restrictions not in the query, *e.g.* "Child that owns some Cat and some Dog", "Child that has some good grades and owns some dog", etc.

This is a significant extension to the language. In effect, it gives meta-semantics as well as first order semantics to some axioms. However, it does not affect existing semantics.

It meets the initial criteria: local satisfiability, sufficient strength without giving unwanted entailments, being reciprocal, and supporting restricted inheritance. It does not affect the scalability of reasoning, and is reasonably intuitive to users. It captures a useful approximation of "some implies may", in that any query that would return the implied subclasses would also return any subclasses that satisfied it explicitly. Users find it intuitive.

There are two difficulties. Firstly, as presented here it is incomplete, and needs to be extended to other constructs such as maxCardinality(n) and, arguably, allValuesFrom that also carry an implication of possibility for one side or the other. In addition, the semantics needs to be specified for those cases in which one or both sides of the axiom are expressions rather than simple classes or restrictions.

More seriously, there is concern over altering the semantics of minCardinality(0) in this way. There are also two other cases in which minCardinality(0) is often overloaded considered in the discussion.

3.5 Creating a new construct and extending standard queries

If overloading an existing construct is inappropriate, would it be appropriate to create a new construct? This would avoid any issues over changing of the semantics of existing constructs. The new construct would carry just the meta-level semantics suggested for minCardinality(0) above – *i.e.* that the corresponding existential was satisfiable and that the appropriate response to a query such as "owns some Dog" would be "Person that owns some Dog" rather than just "Dog".

A new construct in the form of a new qualifier could easily fit with the existing syntax – *e.g.* "mayValuesFrom" or "possiblyValuesFrom", or simply "may" in the Manchester syntax as used here. It would be unambiguous, and could be used immediately by developers even while arguments over fine details of the semantics were being argued out in the standards committees. It would not be confused with existing constructs.[5] It could be restricted to the case where the left hand side of the axiom was a named class and the right hand a single restriction using the new qualifier. The implication of the reciprocal axiom could be made standard, and its routine generation be made part of either the language or the standard API.

[5] A more drastic solution would be to allow the new construct to appear in a new position in the syntax, indicating that it affected both the left and right hand sides of the axiom.

Query languages could be specified so as to return results analogous to “Dog that inv(owns) some Person” in response to queries about “What owns dogs?”. It would be natural to extend this behaviour to other existential restrictions, so that the answer to the query “What parts do cars have” included “Car engines” rather than engines. This would give a clean mechanism for creating and managing partonomies of this type in OWL without requiring either axioms that are palpably false and without encountering the problem of classifying reciprocal (cyclical) restrictions.

Using such axioms to define classes using equivalentClass axioms would rarely be useful since it would not differentiate the defined class from its purported parent, just the use of minCardinality(0) in equivalentClass statements does not differentiate the defined class from its purported parent. Therefore, the new construct would normally appear only subclass axioms and queries, and this might be imposed as a restriction in the language.

The notion of a “proper”[6] ontology, as a more stringent condition than a “satisfiable” ontology would also have merit. A minimum definition would seem to be one in which there are no conditions that are only trivially satisfiable, *i.e.* in which all fillers for universal, maxCardinality, or “may” restrictions are either satisfiable or explicitly owl:Nothing. The distinction would serve as a helpful warning to most users: “This is probably wrong; don’t do it unless you are absolutely sure you know what you are doing.”

Summary of proposed behaviour

We summarise the required behaviour of the new construct briefly below. The semantics is the same whether we overload “minCardinality(0)” or introduce a new construct “mayValuesFrom” which we will abbreviate simply to “may”. The syntax for the queries is left informal, as we wish to suggest requirements for query languages rather than an extension to any specific query language.

Proper and improper “may” axioms: Consider an ontology consisting of a set of class names $C_{i,}$ class expressions using the standard constructors E_i, and properties p_i, and let $\rightarrow$ be an infix form of the OWL keyword “subclassOf”, then let “may” be a new qualifier that can only appear in expressions of the form:

$E_1 \rightarrow$ p may E_2

Where the E_i are expressions in the ontology not containing the qualifier “may”.

Such axioms are “proper” if an only if the class expression “E_1 and p some E_2” is satisfiable, and improper otherwise. The ontology is “proper” if and only if it contains no improper axioms.[7]

[Note that as of this writing, we have identified no use case for the negation “$E_1 \rightarrow$ not (p may E_2)”, and so omit it. Others might suggest such a use case and extend the semantics to accommodate it.]

[6] Others may prefer a different label for “proper”.

[7] One might wish to extend the notion of proper ontology to include the requirement that it contain no axioms that are only trivially satisfiable as discussed in 3.1 and 3.5.

Entailments for "may" axioms: If the ontology contains a proper axiom $E_1 \rightarrow$ p may E_2, then it should be inferred to contain the axiom $E_2 \rightarrow$ inv(p) may E_1.

Responses to queries: To conform to the proposed semantics, query languages, SPARQL-OWL or other, should provide a construct such that:

i) *Proper "may" axioms:* If the ontology contains the proper axiom "$E_1 \rightarrow$ p may E_2", then the returned value set for the class "?C" in queries of the form "?C $\rightarrow$ p some E_2" should include "E_2 that p some E_1" and should not include "E_1" unrestricted (unless it is otherwise implied). Reciprocally, the response to queries of the form "?C $\rightarrow$ inv(p) some E_1" should include "E_2 that inv(p) some E_1" but not "E_2". For completeness, the response to the query "?C $\rightarrow$ p may E_1" should include "E_1" unrestricted and to "?C $\rightarrow$ inv(p) may E_1" should include "E_2" unrestricted.

ii) Fillers of existential axioms: If the ontology contains the existential axiom "$E_1 \rightarrow$ p some E_2", then the query "?C $\rightarrow$ inv(p) some E_1" should include in its response set "E_2 that inv(p) some E_1" and not "E_2" unrestricted (unless it is otherwise implied). Responses to the query "?C $\rightarrow$ inv(p) may E_1" should include "E_2".

iii) "Proper" subclasses of responses: In either case, if the query language supports asking for the subclasses of the responses, then the subclasses, with or without restriction, should included if and only if the corresponding existential restriction is satisfiable, *i.e.* so long as they are "proper" or "satisfiable".

iv) Individuals in responses: If the query is for individuals, then the members of the classes in i) - iii) should form the response set.

A first implementation might restrict the E_i to named classes and still be highly useful.

4. Discussion

4.1 Related issues

The problem of "may" is closely related to the issue of optionality in representing UML and other data models and to the notion of "sanctioning" or "slot attachment" in other knowledge representation formalisms (*e.g.* Grail [5] and the original Protégé frame representations and its basis in the OKBC draft standard. [6]. It is also easily confused with the problem of "typicality".

The question of "what can be said here" or, alternatively, which properties might "sensibly" be used here is another fundamental issue for OWL ontologies. Languages such as UML or frames provide templates for data structures where as OWL provides axioms that restrict those entities. A major request from users is a construct analogous to "optional" in data modelling languages, "slot attachment" in frame based systems, or "sanctioning" in GRAIL. Whether these two issues should be treated in the same way remains open for discussion. At a minimum, sufficient vocabulary is required to distinguish the issues so as to discuss whether or not a single solution meets both.

The mechanism here is reminiscent of the canonical graphs used to establish what may be said in Sowa's Conceptual Graphs. [7] As proposed here, they overload a first order existential semantics with some with additional meta-semantics in order to designate some graphs as “canonical.”

By contrast, what is “typically” true has clearly different semantics from what may be true. What is typically true is not reciprocal. To say that “People who are bald are typically men” does not imply that “Men are typically bald”. Furthermore, “typical” statements are arguably subject to over-riding by exceptions in ways that are difficult to represent without significant extensions to the semantics. On the other hand, the issues are related, and parts of the machinery proposed here to deal with “may” might be re-used to deal with some aspects of the problem of “typically”. A full discussion is beyond the scope of this paper.

Finally, we note that the explicit use of pairs of super- and sub-properties, as in 3.2, and the use of a new construct, as in 3.5, might profitably be combined, provided that reasoners scaling problems with reciprocal relations could be overcome. The use of super-and sub-properties naturally corresponds to our intuitions about “potential.” The use of the new construct to our notion of “some” or “optionally”. In some applications, it would be useful to combine them so that we could identify, for example, the subclass of bacteria that might potentially cause pneumonia as opposed to those that actually cause pneumonia. Whether or not users, or developers, could maintain the distinctions consistently is questionable. This is a topic to be explored empirically once the basic construct is implemented.

4.2 Summary

In summary, we have presented here an argument for a standard answer to the question “how do I say ‘may’ in OWL?”. In order to analyse the required semantics, we have started from the answers expected to questions, as the best way of specifying the users’ intuitions to be modeled. We argue that, given a statement “Bacteria may cause Pneumonia”, the correct answer to the query “What causes pneumonia” is “Bacteria that cause Pneumonia”. We have further argued that the query should not return either the unrestricted class “Bacteria” – to strong – or more generic entities such as “Living thing.” We argue that any solution ought to be uniform with the means for expressing notions of “may” that arise naturally as the reciprocals of existential restrictions. One important case is in expressing partonomies, where we would like the answer to “What parts do cars have” to include “car engines” rather than just “engines”. In all cases we wish to avoid entering axioms in the ontology that are patently false, simply in order to get the required behaviour, as is sometimes done in creating partonomies.

In the extreme, all of the required responses could be entered into the ontology explicitly as defined classes, but this would overload the ontology with an exponentially explosive set of definitions for all classes corresponding to all possible combinations of restrictions that “may” apply to a given class.

Barring this, we argue that some extension to the language is required, and that this extension will inevitably go beyond strictly first order representation to provide some degree of meta-representation. Options are overloading minCardinality(0) or creating a new construct, most probably a new “qualifier” syntactically analogous to allValuesFrom and someValuesFrom. We propose an operational semantics that which gives answers that can be interpreted strictly in terms of the first order semantics. The proposed changes under the suggested semantics require no changes to reasoners, merely to the questions asked of them when an ontology contains the new construct. However, they would require modest changes to query languages. Lacking a new construct, we have experimented with the use of minCardinality(0) and explicit defined classes as suggested here to test that the results are as we believe users intend.

To those who argue that it is unreasonable to expect an OWL to answer these queries because they involve some degree of meta-reasoning, we respond that without the ability to answer such queries, the range of OWL's application will be severely restricted. The proposals would provide a standard means of expressing knowledge that is critical to many applications and for which users now find a various, often patently incorrect, work-arounds.

Acknowledgements

This work supported in part by the JISC and UK EPSRC projects CO-ODE and HyOntUse (GR/S44686/1), the EU funded Semantic Mining Network of Excellence.

References

1. Rector A, Welty C. Simple part-whole relations in OWL Ontologies: W3C Editors working draft. 2005; http://www.w3.org/2001/sw/BestPractices/OEP/SimplePartWhol.
2. Horridge M, Drummond N, Goodwin J, Rector A, Stevens R, Wang H; The Manchester OWL syntax. 2006; OWL: Experiences and Directions (OWLED 06): Athens, Georgia: CEUR; http://sunsite.informatik.rwth-aachen.de/Publications/CEUR-WS//Vol-216/submission_9.pdf.
3. Arp R, Smith B. Function, role, and disposition in Basic Formal Ontology. *nature procedings*.
4. Motik B, Shearer R, Horrocks I; Optimized reasoning in description logics using hypertableaux. 2007; 21st Conference on Automated Deduction (CADE-21): Bremen, Germany: Springer LNAI; 67-83.
5. Rector AL, Bechhofer S, Goble CA, Horrocks I, Nowlan WA, Solomon WD. The GRAIL concept modelling language for medical terminology. *Artificial Intelligence in Medicine.* 1997;9:139-171.
6. Chaudhri VK, Farquhar A, Fikes R, Karp PD, Rice a; OKBC: A programmatic foundation for knowledge base interoperability. 1998; 15th National Conference on Artificial Intelligence (AAAI 98): 600-607.
7. Sowa J. Conceptual Structures: Knowledge Representation in Mind and Machine. New York: John Wiley & Sons; 1985