This paper has been presented at Int. Conference KEOD-2017 (http://www.keod.ic3k.org/)

Representing Ecological Network Specifications with Semantic Web Techniques

Gianluca Torta[1], Liliana Ardissono[1], Luigi La Riccia[2], Adriano Savoca[1] and Angioletta Voghera[2]
[1]*Dipartimento di Informatica, Università di Torino, Italy*
[2]*Dipartimento Interateneo di Scienze, Progetto e Politiche del Territorio, Torino, Italy*
{torta, ardissono}@unito.it, adrianosavoca@gmail.com, {luigi.lariccia, angioletta.voghera}@polito.it

Keywords: Geographic Knowledge, Geographical Constraints, GeoSPARQL, Ecological Networks, Urban Planning.

Abstract: Ecological Networks (ENs) are a way to describe the structures of existing real ecosystems and to plan their expansion, conservation and improvement. In this work, we present a model to represent the specifications for the local planning of ENs in a way that can support reasoning, e.g., to detect violations within new proposals of expansion, or to reason about improvements of the networks. Moreover, we describe an OWL ontology for the representation of ENs themselves. In the context of knowledge engineering, ENs provide a complex, inherently geographic domain that demands for the expressive power of a language like OWL augmented with the GeoSPARQL ontology to be conveniently represented. More importantly, the set of specification rules that we consider (taken from the project for a local EN implementation) constitute a challenging problem for representing constraints over complex geographic domains, and evaluating whether a given large knowledge base satisfies or violates them.

1 INTRODUCTION

New urbanizations, infrastructural networks and intensive agriculture have increased the fragmentation process of nature, with the consequent reduction of natural environments and the loss of habitats able to sustain the wild species and their migrations. The concept and implementation of Ecological Networks (ENs) represents a way to describe existing ecosystems and to plan their expansion, conservation and improvement with the aim of preserving biodiversity by reducing the fragmentation of the natural environment (Jongman, 1995). During the past years, the implementation of ENs at different scales (from European down to municipality) has received a lot of attention. Several documents containing guidelines have been issued for their implementation (Council of Europe, 2000; Bennett and Wit, 2001; Bennett and Mulongoy, 2006), challenging the human user with the dispersion of specifications in different textual sources. A formal approach to their representation, suitable for integrating heterogeneous information sources, is thus needed.

This paper presents an ontology for the description of ENs, and a constraint specification language aimed at supporting automatic reasoning with specifications regarding ENs. We currently focus on detecting violations within new proposals of expansion or improvement of the EN. However, our approach can be seamlessly extended to offer other reasoning features, such as the automatic proposal of changes and additions to an existing EN. Our ontology is based on the outcome of project "Experimental activity of participatory elaboration of an ecological network" conducted by the Metropolitan City of Turin (Italy) with Polytechnic of Turin and ENEA (Città Metropolitana di Torino, 2014). The project aimed at defining a proposal for the Ecological Network implementation at the local level in two pilot municipalities near Turin. The proposed approach was aimed at guiding local Public Administrations with measures to limit anthropogenic land use and, when possible, orient and qualify the conservation of ecosystem services.

In the context of knowledge engineering, ENs provide a complex, inherently geographic domain that demands an expressive language to be conveniently represented. Moreover, the EN specifications constitute a challenging problem for representing constraints over complex geographic domains, and evaluating if a given (possibly large) knowledge base satisfies or violates them. We adopted OWL (W3C, 2012), augmented by the GeoSPARQL ontology (Perry and Herring, 2012), for representing the main concepts characterizing the EN domain. However, regarding

Torta, G., Ardissono, L., Riccia, L., Savoca, A. and Voghera, A.
Representing Ecological Network Specifications with Semantic Web Techniques.
In *Proceedings of the 9th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management (IC3K 2017) - Volume 2: KEOD*, pages 86-97
ISBN: 978-989-758-272-1

the representation of EN specifications, we developed a new language guided by the following main goals:

1. The ability of expressing constraints on the EN domain in a simple and compact way.
2. The possibility of exploiting the EN specifications expressed in the language for different automatic tasks, which can't be implemented by simply querying an existing KB. For instance, we do not only consider the validation of a (possibly large) Knowledge Base that represents an EN by querying its contents, but also (i) checking the consistency of (partially defined) proposals for changes/additions to an existing EN, or (ii) automatically suggesting optimized changes or additions to an existing Ecological Network.

In line with recent research on geometric constraint reasoning, e.g., (Ajit et al., 2008; Louwsma et al., 2006), we have thus defined a high-level language, that abstracts from the details of the reasoning tasks that it will support. Our language is named *GeCoLan* after *CoLan* (Bassiliades and Gray, 1995), a language originally devised for Object-Oriented Databases on which it is strongly based, with restrictions and extensions. GeCoLan is aimed at simplifying the definition of complex arithmetic and geometric constraints, as well as supporting rich types of reasoning in addition to constraint verification. The proposed language is tailored to the needs of our domain, but it is not limited to be used for Ecological Networks.

We implemented the EN validation task by automatically mapping the GeCoLan specifications to GeoSPARQL queries, in order to exploit existing technologies for GeoSPARQL processing, rather than developing a processing engine from scratch. In particular, we developed a prototype for the automatic translation and check of EN specifications, using the efficient validity checks offered by the GeoSPARQL engine embedded within the Parliament Triple Store (Battle and Kolas, 2012).

We tested our prototype on a large knowledge base derived from project (Città Metropolitana di Torino, 2014), obtaining positive results regarding the efficiency of our approach.

The remainder of this paper is organized as follows: Section 2 provides some background on ENs and positions our work in the related literature. Section 3 describes our representation of ENs specifications and provides details about the GeCoLan language. Section 4 presents our approach to the validation of ecological networks and the tool we developed for that purpose. Section 5 discusses the broader context of our work, outlining our future research directions. Section 6 concludes the paper.

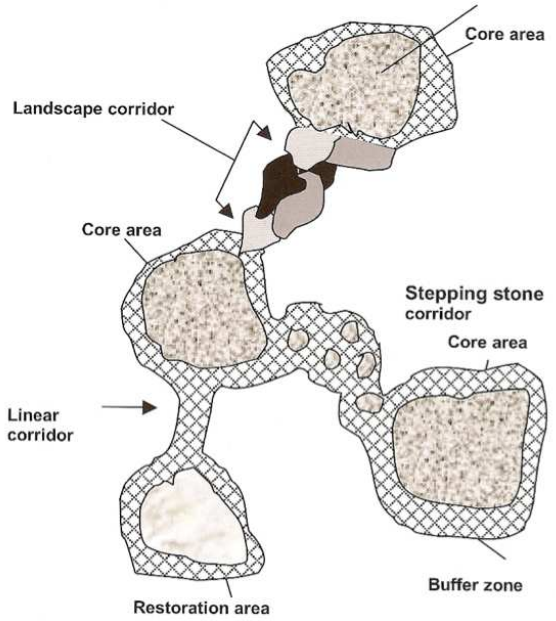

Figure 1: Diagrammatic representation of the spatial configuration of an Ecological Network, from (Bennett and Mulongoy, 2006).

2 BACKGROUND AND RELATED WORK

Ecological Networks consist of different kinds of elements: Structural Elements, or *core areas* (the nodes of the network, i.e., areas with highest "naturality"); their Adjoining Areas, or *buffers* (the areas neighboring the Structural Elements); and Connection Elements, or *corridors* (links of the network, i.e., areas with residual naturality that connect the Structural Elements). Figure 1 depicts the general design of an Ecological Network.

Several mathematical models have been developed to analyze and predict the dynamics of ENs in terms of interactions between organisms in an ecosystem, starting from observational data; e.g., (Ulanowicz, 2004; Fath et al., 2007; Lurgi and Robertson, 2011; Gobluski et al., 2016; Pilosof et al., 2017). These models are aimed at modeling and simulating the dynamics of the relations among species. Differently, to our knowledge, no existing works focus on land-usage constraints, and on how the transformations of ENs can be regulated in policy-making, taking such constraints into account.

With the aim of representing knowledge in a formal way that supports the specification of the semantics of concepts, we selected OWL (W3C, 2012) to represent the EN domain, augmented by the GeoSPARQL ontology (Perry and Herring, 2012).

These languages are well known standards supported by many tools, including highly scalable and efficient RDF (W3C, 2014) data stores, such as Triple Stores (Battle and Kolas, 2012). By being Semantic Web languages, they also seamlessly allow publishing the information on the Web for (open) access and processing, thus favoring knowledge sharing.

A way to represent constraints in the Semantic Web, is through *rule languages*, such as SWRL (Semantic Web Rule Language) (Horrocks et al., 2004) and RuleML (Boley et al., 2001). While the original expressivity of those rule languages was not general enough to support constructs such as logic quantifiers, negation, disjunction, numeric and geometric expressions (all of which typically occur in EN specifications, as we shall see), subsequent extensions empowered them with the full expressiveness of First-Order Logic (FOL) (Boley et al., 2004; W3C, 2005). Moreover, some work integrated SWRL with mathematical, geometrical, and other types of functions through built-ins; e.g., (Kessler et al., 2009).

Our work introduces the new language GeCoLan based on CoLan (Bassiliades and Gray, 1995), instead of extending an existing rule-based language, mainly because of several syntactic aspects that make it particularly succinct and convenient for our needs. We don't exclude that future evolutions of our research may make us consider translating GeCoLan to a standard rule-based language to exploit existing inference engines for those languages. See section 3.2 for more details.

Another relevant family of languages devised for manipulating the data in the Semantic Web are query languages, e.g. (W3C, 2013; Arenas et al., 2014). Specifically, SPARQL is a powerful query language which, beside allowing to access, insert, and update the triples in a Triple Store, has been used for expressing and checking constraints over OWL-based Triple Stores (Fürber and Hepp, 2010). However, the direct adoption of SPARQL (or of other query languages) for the description of EN specifications fails to support reasoning tasks that can't be implemented by simply querying an existing KB.

3 REPRESENTATION OF ECOLOGICAL NETWORKS

3.1 An OWL Ontology for Ecological Networks

In order to represent the concepts related with Ecological Networks and the associated specifications, we needed a rich language, able to deal both with the complexity of individual concepts and with their relations. The following are some key features required by any candidate language:

- ability to model *taxonomies*: e.g., hierarchies of land use types, of planned interventions, etc.;
- ability to model (and support calculations with) geographic/geometric shapes of concepts: e.g., elements of the Ecological Network and elements of land use maps;
- ability to specify restrictions on concepts and their relations: e.g., "coppicing is a maintenance intervention that only applies to wooden areas".

We selected OWL2 (Grau et al., 2008) to represent the structure of the data relevant to Ecological Networks because it satisfies all of the above requirements, and it is a very well-understood and well-supported standard for modeling data in the Semantic Web.

The central concept of the EN ontology is the *Feature*, a class defined in the GeoSPARQL ontology (Battle and Kolas, 2012; OGC, 2012). A *Feature* is a *SpatialObject* that has a *Geometry* on the 2D plane, which can be either a *Point*, a *Curve* (and, in particular, a piecewise linear curve), or a *Polygon*. Most interestingly, GeoSPARQL defines a number of topologic geometric relations between *Feature*s that correspond to the fundamental relations identified in the literature about Geographic Information Systems: namely, the Simple Features relations (Open Geospatial Consortium et al., 2011), and the equivalent RCC8 (Cohn et al., 1997), and Egenhofer relations (Egenhofer, 1989).

Figure 2 shows the Simple Features topological properties, which relate objects of the *SpatialObject* class. By specializing the *Feature* concept, we define

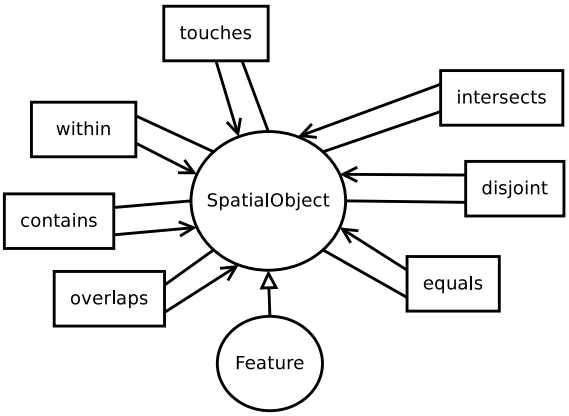

Figure 2: A fragment of the GeoSPARQL ontology with the Feature class and the Simple Features topological properties.

the four hierarchies of concepts representing the core of the Ecological Networks domain:

- *EcologicalNetworkElement* represents an element of the EN, i.e., either a Structural Element, or a Priority Expansion Element, which is further specialized to a Connection Element or Adjoining Area.
- *LandUseElement* represents a Land Use Element (LUE), i.e., a parcel of land defined in the Land Cover Piemonte (LCP) cartography (Provincia di Torino, 2014), characterized by a specific type of land use, e.g., wetland, wooden land, and similar. The LCP defines 97 different types of land use at the most detailed level (level 4) of a hierarchy which includes 45 classes at level 3, 15 classes at level 2, and 5 general classes at the most general level 1. We only model as distinct classes the values of the first level, namely *WaterBody*, *WetLand*, *WoodenLand*, *AgriculturalLand*, and *ArtificialLand*. The values of levels 2, 3, and 4 are specified as properties of *LandUseElement*: *LCPlevel2*, *LCPlevel3*, and *LCPlevel4*.
- *Intervention* represents an intervention for building, improving or conserving the EN.
- *Operation* represents a specific operation of elimination, construction or maintenance that is part of an intervention.

Figure 3 shows a portion of our EN ontology, with the *Feature* class and the first levels of the four hierarchies discussed above. Being members of *Feature*, any two individuals that belong to the ontology classes can be related with each other through the topological relations mentioned above. For example, a *ConnectionElement* may *touch* a number of *LandUseElements*, where, intuitively, the Simple Features *touches* relation between 2D areas *A*, *B* holds whenever the borders (but not the interiors) of *A*, *B* have a non-empty intersection.

In order to characterize the classes and properties used in our ontology, we exploit the powerful constructs of the OWL2 language (in particular, Class Expressions) in several places. For example:

- the *hasPlants* property applies to LUEs of types *WoodenLand* and *AgriLand*. The domain of *hasPlants* is therefore the following OWL Class Expression (in functional syntax):

```
ObjectUnionOf(WoodenLand AgriLand)
```

- the *hasArea* property applies to *Feature*s whose geometry is a *Surface*. The domain of *hasArea* is therefore the following OWL Class Expression:

```
AllValuesFrom(hasGeometry Surface)
```

- the *Coppicing* class (subclass of the *Maintenance* operation) should have a linear geometry. Class *Coppicing* is therefore also a subclass of the following Class Expression:

```
AllValuesFrom(hasGeometry LineString)
```

- the *Eradication* class (subclass of the *Elimination* operation) should only *intersect* LUEs of type *WoodenLand*s (first level of the LCP hierarchy); furthermore, such elements should have an *LCPlevel2* value of either “ShrubbyGrassyZone” or “OpenZone”. Class *Eradication* is therefore also a subclass of the following Class Expression:

```
AllValuesFrom(sfIntersects
 (ObjectIntersectionOf(WoodenLand
  (DataAllValuesFrom(LCPlevel2
   DataOneOf("ShrubbyGrassyZone"
        "OpenZone")))))
```

It is worth noting that all of the above examples are constraints on the classes and properties of the EN ontology. In particular, we regard them as *integrity constraints* on the vocabulary defined by the ontology: for example, it would be meaningless to have an element of the *Coppicing* class with a non-linear geometry, or an *Eradication* operation over an urban area. On the contrary, the EN specifications described in the next section are meant to constrain the way we are *allowed to plan* an Ecological Network, and may possibly be violated by existing ENs that have not been correctly planned. This is an important reason to encode the two types of constraints differently, besides the fact that OWL has limits on the logic constraints that it can express (Krötzsch, 2010).

3.2 A Language for Representing EN Specifications

As previously stated, the EN specification language we developed is aimed at supporting various types of reasoning about constraints on ENs. For that purpose, it must:

- use the terms defined in the EN ontology (classes and properties) as its main vocabulary;
- be able to express *constraints* about the objects of the world modeled by the ontology;
- be able to mix logic, geometric, numeric, and time requirements into the constraints.

We defined a language for the representation of EN specifications that is strongly influenced by CoLan, a language initially proposed for expressing integrity constraints in Object-Oriented Databases (Bassiliades

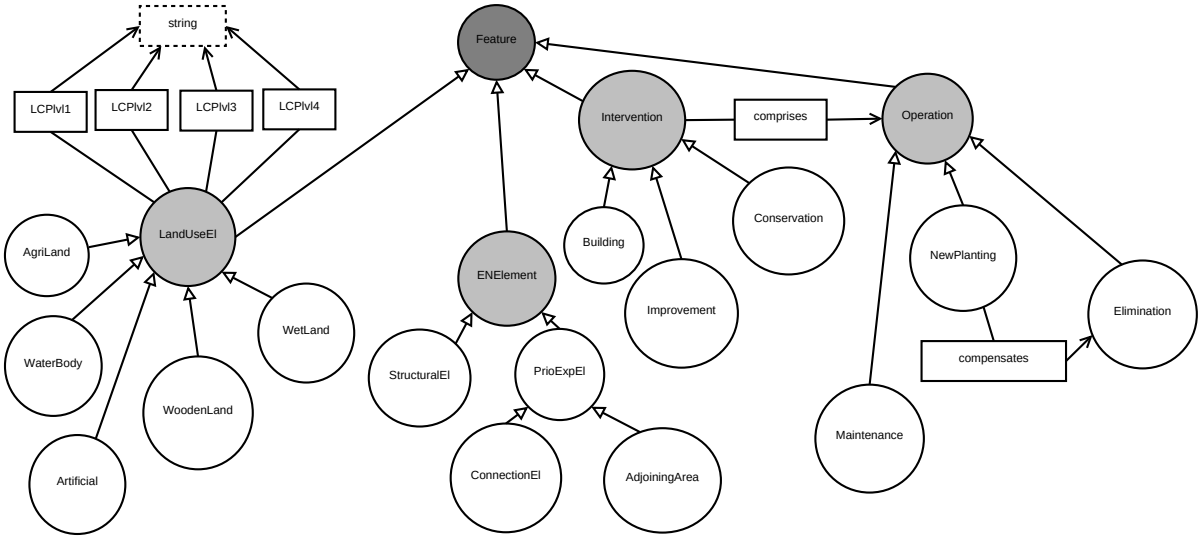

Figure 3: A fragment of the EN ontology.

and Gray, 1995), and later used as the basis for the Constraint Interchange Format (CIF), an RDF schema for exchanging constraints on the Semantic Web (Gray et al., 2001). We named our language *GeCoLan*, given that it is intended to capture geometric and geographic constraints, beside more traditional types of constraints captured by CoLan. GeCoLan inherits the expressivity and understandability of CoLan, with some changes and additions tailored to the requirements of our domain.

Compared to rule-based languages such as SWRL and RuleML (even in their extensions with First Order Logic), GeCoLan has a number of syntactic features that make it particularly convenient for our purposes:

- it allows the use of nested functions that correspond either to functions defined in (Geo)SPARQL, or to functional properties of the ontology classes. For example,

  $$\texttt{LCPlevel2}(\texttt{el})$$

  indicates the value of the (functional) property *LCPlevel2* of variable *el*, while:

  $$\texttt{distance}(\texttt{intersection}(\texttt{l1},\texttt{l2}),\ \texttt{intersection}(\texttt{l3},\texttt{l4}))$$

  indicates the distance between the intersection area of variables $l1$ and $l2$, which are *Well Known Text* (WKT) literals (i.e., serializations of geometries), and the intersection area of variables $l3$ and $l4$.

- it allows the use of infix relational and arithmetic operators in expressions over complex terms. For example:

  $$\{\texttt{distance}(\texttt{l1},\texttt{l2}) >= 2 * \texttt{distance}(\texttt{l1},\texttt{l3})\}$$

  means that the distance between the WKT literal values of variables $l1$ and $l2$ is at least twice as the distance between the values of $l1$, $l3$. Similarly,

  $$\{\texttt{irreversibility}(\texttt{el}) < 3\ ||\ \texttt{extroversion}(\texttt{el}) < 3)\}$$

  constrains the values of functional properties *irreversibility* and *extroversion* of variable *el*.

- it allows the quantifiers `forall` and `exists` with *range restricted* variables:

  $$\texttt{forall}\ \texttt{C}(\texttt{x})[\texttt{s.t.}\ \psi(\texttt{x})]\varphi(\texttt{x})$$
  $$\texttt{exists}\ \texttt{C}(\texttt{x})[\texttt{s.t.}\ \psi(\texttt{x})]\varphi(\texttt{x})$$

  where C is the name of a class of the ontology, x is a variable that represents an individual, and $\psi(x)$ is an optional formula that further restricts the individuals of C we are interested in. For example, "`forall NewPlanting(np)`" states that the value of variable *np* can be any individual belonging to class *NewPlanting*. If we add a condition "`s.t. compensates(np,el)`", only individuals *np* that compensate some given elimination *el* will be considered.

These features can be viewed as syntactic sugar over the constructs of rule based languages, and it is indeed conceivable to automatically translate a GeCoLan expression into, e.g., a less concise SWRL FOL expression. In this vein, in section 4 we discuss a translation of GeCoLan to GeoSPARQL in order to take advantage of existing SPARQL engines (which are based on closed-world semantics) for validating GeCoLan specifications by querying existing KBs. Currently,

we haven't yet found a compelling reason for translating GeCoLan to a rule language for exploiting existing inference engines for those languages (which are based on open-world semantics), but this may change in the future as we consider other reasoning tasks to tackle with GeCoLan (section 3.4).

To complete the desctiption of GeCoLan, we have to add that it allows the usual connectives `and`, `or`, `not`, and `implies` of First Order Logic (FOL), while the atoms can be in one of the following forms:

- *property* predicates $\texttt{p}(\texttt{t1},\texttt{t2})$, where p is the name of a property defined in the ontology, t_1 is a term (see below) representing an individual in the domain of p, and t_2 is a term representing an individual in the range of p, e.g., $\texttt{sfTouches}(\texttt{g1},\texttt{g2})$ states that the value of variable $g1$ (which is a *Geometry*) touches the value of variable $g2$ (another *Geometry*);
- *filter expressions* $\{\texttt{e}\}$[1]; a filter expression e (enclosed in braces $\{$, $\}$) can use the logic connectives && (and), || (or), and ! (not) between conditions built with relational operators >, <, <=, >=, =, !=. The operands of such relational operators are terms or numeric expressions obtained by combining numeric operands with operators +, -, /, *.

As mentioned above, the terms appearing in the property predicates and filter expressions are:

- literal values (numbers, strings, etc.);
- variables;
- function calls whose argoments are themselves terms.

Since we aim at expressing constraints that can be either satisfied or violated by the knowledge base, we are only interested in closed formulas, i.e. formulas where all the variables are within the scope of a corresponding quantifier.

3.3 Example Specifications

In order to illustrate the efficacy of the language for our purposes, let us consider the representation of two specifications taken from the guidelines for the Local EN implementation near Turin devised in project (Città Metropolitana di Torino, 2014). The first one is about **Connection Elements** and requires that:

> *Connection elements avoid areas with maximum irreversibility and areas with maximum extroversion.*

[1] As we shall see in section 4.1, filter expressions correspond to whole constraints to be evaluated in the *FILTER* clause of SPARQL

Using the vocabulary of the EN ontology, this constraint is satisfied if a *ConnectionElement* does not *overlap* with any *LandUseElement*s whose *irreversibility* or *extroversion* properties have the maximum value. It is worth briefly mentioning that each *LandUseElement* is characterized by five evaluation criteria: *naturality* (how close the element is to a natural environment), *relevance* (how relevant it is for the conservation of the habitat), *fragility* (how fragile it is w.r.t. the anthropogenic pressure), *extroversion* (how much pressure it can exert on the neighbouring areas), and *irreversibility* (how difficult it would be to change its use). The value for each criterion ranges from 1 to 5, with 1 representing the maximum value. One possible encoding of the specification in GeCoLan is therefore as follows:

```
forall ConnectionElement(ce)
forall LandUseElement(lue) such that
 sfOverlaps(ce,lue)
   {irreversibility(lue) > 1 &&
    extroversion(lue) > 1}
```

The formula starts with two universal quantifiers. The first one restricts the range of variable *ce*, that must be a *ConnectionElement*. The second one restricts variable *lue*, that must be a *LandUseElement*, and must have a geometry *G* that *overlaps* with the geometry of *ce*.[2] The body is a filter expression stating that both the *irreversibility* and the *extroversion* of *lue* must have a higher value than 1. Although this example is rather simple, it already makes use of several features of our ontology and language, in particular:

- the ability to automatically find the *LandUseElement*s that fall (at least partially) within a *ConnectionElement* - this is provided by the geometric capabilities of GeoSPARQL;
- the ability to perform numeric comparisons between the values of some (functional) properties of *LandUseElement*s and a numeric constant - this is provided by the filter expressions.

Let us now consider a more complex example, regarding the creation of **Buffer Zones** to protect the elements of the EN:

> *The creation of protection buffers is done, whenever possible, through interventions of restoration in areas surrounding the structural elements of the net, with the goal of enhancing and protecting them. In case a structural element is surrounded by areas with*

[2] In GeoSPARQL, the topological relations such as *sfOverlaps* are computed on the geometries associated with the individuals in class *Feature* (*lue* and *ce* in our example) through the *defaultGeometry* property.

maximum extroversion and/or maximum irreversibility, the buffers must be realized within the structural element (due to the impossibility of extending the element itself).

Using the vocabulary of the EN ontology, a *BufferCreation* is an *Intervention* of *Conservation* that *touches* a *StructuralElement* of the net, except when, in such a way, it would *overlap* with one or more *LandUseElement*s with maximum *irreversibility* or *extroversion*. In that case, the *BufferCreation* should be *within* the *StructuralElement*; see Figure 4, where the two cases are denoted as (a) and (b). One possible encoding of the specification in GeCoLan is as follows:

```
forall BufferCreation(bc)
forall StructuralElement(sel)
forall LandUseElement(lue) such that (not sfWithin(lue,sel))
  ((sfTouches(bc,sel) and sfOverlaps(bc,lue))
    implies
    {(irreversibility(lue) > 1) and (extroversion(lue) > 1)})
   and
  ((sfWithin(bc,sel) and sfTouches(bc,lue))
    implies
    {(irreversibility(lue) = 1) or (extroversion(lue) = 1)})
```

The formula starts with universal quantifiers specifying the classes of variables *bc*, *sel*, and *lue*. The quantification of *lue* further restricts the attention to *LandUseElement*s whose geometry is not *within* (i.e. part of) the geometry of *StructuralElement sel*. The body is the conjunction of two sub-specifications, corresponding to the different cases (a) and (b) of Figure 4: if (the geometry of) *bc touch*es *sel*, then each *lue* that (partially) overlaps with *bc* must have non-maximum irreversibility and extroversion. Differently, if *bc* is *within sel*, then each *lue touch*ed by *bc* must have maximum irreversibility and/or extroversion.

3.4 Reasoning with Specifications

It is worth considering at this point some reasoning tasks that may benefit from the specifications illustrated in our examples. As stated in the introduction, we currently focus on *validation* tasks aimed at checking whether a given EN satisfies the specifications or not. However, other interesting reasoning tasks can exploit GeCoLan specifications, and we want to briefly discuss some of them before diving into the details of our implementation of the validation task.

As an example, let us consider the task of checking the **Connection Elements** specification of the previous section. That means considering each *ConnectionElement ce* in the Triple Store, and verifying that all of the *LandUseElement*s overlapping with *ce* satisfy the given restrictions on the *extroversion* and *irreversibility* properties. This kind of check can be done by looking at the contents of the Triple Store and ensuring that they don't conflict with the specifications. The underlying assumption is that all of the relevant facts are present in the store (*closed world* semantics), so that, e.g., if a required fact is missing from the store, a violation is detected.

A natural extension of such a validation would be to attempt suggesting how to fix the detected violations. For example, it is quite plausible that the check of the **Connection Elements** specification may return a list of pairs (ce, lue) indicating which *LandUseElement*s *lue* that overlap with some *ConnectionElement*s *ce* have invalid *extroversion* and/or *irreversibility*.

- One way to remove the inconsistency, would be to remove the *ConnectionElement ce* from the KB. However, this might cause an even worse inconsistency in the KB, because another specification (not considered in our examples) states that any two *StructuralElement*s of the EN must be connected through elements of the EN. Thus, removing *ce* may cause two or more *StructuralElement*s to become mutually unreachable.
- Another way to remove the inconsistency of a pair (ce, lue) w.r.t. the **Connection Elements** specification would be to reduce the geometry of *ce* so that it no longer overlaps with the "bad" *LandUseElement lue*; however, this might cause the disconnection of elements of the EN, including a possible separation of *ce* itself into different parts.

An even more challenging task would be to ask an automated reasoner to suggest how to connect a new *StructuralElement sel* to the rest of the EN, by proposing a path of new *ConnectionElement*s that lead from *sel* to an existing element of the net.

The **Connection Elements** specification is relevant to all the tasks we have mentioned (and to many others), but clearly not all of them can be solved by simply querying the KB through a language like SPARQL. Some tasks may require reasoning engines such as, e.g., Prolog, Answer Set Programming, and Constraint Programming, or even specialized reasoners for 2D geometric reasoning. For example, let us assume that some EN specifications were translated to a CSP (Constraint Satisfaction Problem) for an EN containing a pair (ce, lue) which violates **Connection Elements**. If the following facts were retracted from the CSP:

```
ConnectionElement ce.
overlaps(ce,lue).
irreversibility(lue) = 1.
extroversion(lue) = 1.
```

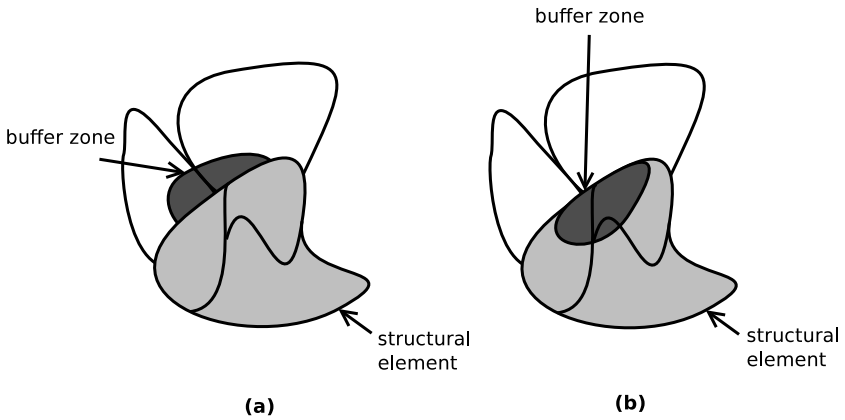

Figure 4: Graphical representation of the two cases of the specification for Buffer Zones.

we may expect a CSP solver to be able to suggest a change to the type of *ce*, and/or to its overlapping with *lue*, and/or to the attributes of *lue*, that satisfies the **Connection Elements** specification while preserving the connection of the EN elements (if this latter constraint is encoded in the CSP).

Of course, this is just an illustrative example, and a systemathic proposal for exploiting the EN specifications for the kinds of reasoning we mentioned will require further research work. The point that we want to stress here, is that by adopting a higher-level language such as GeCoLan for encoding the specifications, we can then (re-)use such specifications as inputs to an appropriate combination of reasoners for solving different and possibly more complex tasks.

4 VALIDATION OF ECOLOGICAL NETWORKS

4.1 Translating Specifications to GeoSPARQL Queries

In order to check whether the specifications discussed in the previous section are satisfied by an existing knowledge base, we translate them to equivalent GeoSPARQL queries, that can be executed by any engine that supports the GeoSPARQL standard; e.g., Parliament (Battle and Kolas, 2012).

Given a closed formula ϕ in the GeCoLan language, we want to translate it to a GeoSPARQL query q_ϕ s.t.:

$$KB \models \phi \leftrightarrow \begin{cases} [\![q_\phi]\!]^{KB} \neq \emptyset & \phi = \texttt{exists C(x)}\varphi\texttt{(x)} \\ [\![q_\phi]\!]^{KB} = \emptyset & \phi = \texttt{not exists C(x)}\varphi\texttt{(x)} \end{cases} \quad (1)$$

where ϕ is in either one of the above forms (as we shall shortly see all formulas of GeCoLan are converted in one of these forms before translating them to SPARQL), *KB* is the Triple Store against which we want to evaluate ϕ, and $[\![q_\phi]\!]^{KB}$ denotes the result of applying q_ϕ to *KB*.

Similarly to (Angles and Gutierrez, 2008; Kostylev et al., 2015), we define the translation inductively. First of all, we consider some rewritings of ϕ into equivalent GeCoLan formulas in order to reduce the number of cases we have to explicitly translate:

`forall C(x)` φ	$\mathcal{T}$(`not exists C(x) (not` φ`)`)
`exists C(x) s.t.`ψ (φ)	$\mathcal{T}$(`exists x (C(x) and (`ψ `implies` φ`))`)
φ `implies` ψ	$\mathcal{T}$(`not` φ `or` ψ)

After these rewritings, formula ϕ starts either with "`not exists C(x)`", or "`exists C(x)`". Before starting the translation, we remove such a prefix from ϕ so that the values returned by the query are not completely projected out; see the example below. Whether ϕ is satisfied or not by *KB* depends on the result of the application of the translation of ϕ to *KB*, according to equation (1).

Having got rid of the `forall` quantifier and the `implies` connective, let us now consider the translation of the remaining quantifiers and connectives. For finite domain classes $C_1, \ldots, C_k$, we define the cross product of their domains as:

$$ADom(x_1, \ldots, x_k) = \{\texttt{x}_1\ \texttt{rdf:type C}_1. \ \ldots \ \texttt{x}_\texttt{k}\ \texttt{rdf:type C}_\texttt{k}.\}$$

We then define the following translations:

`exists x` φ	`SELECT` ($free(\varphi) - \{x\}$) `WHERE` $\{\mathcal{T}(\varphi)\}$
φ `or` ψ	$\mathcal{T}(\varphi)$ `UNION` $\mathcal{T}(\psi)$
φ `and` ψ	$\{\mathcal{T}(\varphi)\ \mathcal{T}(\psi)\}$
$\neg\varphi$	$\{ADom(free(\varphi))$ `FILTER NOT EXISTS` $\mathcal{T}(\varphi)\}$

The translation of `exists` involves the projection of variable *x* through a subquery, since *x* should no longer appear in outer formulas. The translations

of `or` and `and` connectors are mapped, respectively, to the `UNION` clause of SPARQL and to a sequence of graph patterns. Finally, negation of φ maps to a `FILTER NOT EXISTS` clause between the domains of the free variables and their assignments satisfying φ.

We still need to describe the translation of the atoms of GeConLan:

C(x)	{x rdf : type C.}
p(t1,t2)	{$\mathcal{T}$(t1) $\mathcal{T}$(t2) x1 p x2.}
{e(t1,...,tn)}	$\mathcal{T}$(t1) ...$\mathcal{T}$(tn) FILTER e(x1,...,xn)

The class C of a variable x is translated as a graph pattern with predicate `rdf : type`. Let us consider the translation of $p(t_1,t_2)$. First of all, we need to translate the terms t_1, t_2 themselves. For example, let the property atom be:

$$\texttt{member}(\texttt{hasPlants}(\texttt{lue}), \texttt{"Rudbekia"})$$

where t_1 is *hasPlants(lue)*, i.e., the application of a functional property *hasPlants* to a *LandUseElement lue*; the *hasPlants* property maps *lue* to a *Collection* element *col*. We need to introduce a new variable x_1 for representing the value of such property, and add the pattern {lue hasPlants x1} to the translation. Then, to complete the translation, we add the pattern {x1 member "Rudbekia"} to state that Rudbekia is a member of the $x1$ collection. The translation of filter expressions requires a similar approach to generate variables corresponding to applications of functional properties.

Let us now illustrate the translation method described above by applying it to the **Connection Elements** example introduced in section 3.3. First, the GeCoLan formula is transformed in order to eliminate the `forall` quantifiers:

```
not exists ConnectionElement(ce)
exists LandUseElement(lue)
   not (sfOverlaps(ce,lue) implies
          {irreversibility(lue) > 1 &&
           extroversion(lue) > 1})
```

and then to eliminate the `implies` connective:

```
not exists ConnectionElement(ce)
exists LandUseElement(lue)
  (sfOverlaps(ce,lue) and
   not{irreversibility(lue) > 1 &&
         extroversion(lue) > 1})
```

Let us consider the translation of the filter expression:

$$\begin{aligned}\tau_0 = \quad & \{\{\texttt{lue irreversibility ir.}\} \\ & \{\texttt{lue extroversion ex.}\} \\ & \texttt{FILTER } ((\texttt{ir} > 1) \;\&\&\; (\texttt{ex} > 1))\}\end{aligned}$$

Two new variables (that we have named *ir* and *ex* for readability) have been automatically added to hold the values of the *irreversibility* and *extroversion* properties of *lue*. Then, the filter expression (where the property applications have been replaced with *ir*, *ex*) becomes the argument of a `FILTER` clause.

As the range expression for *ce*, *lue* is:

$$\begin{aligned}\tau_R = \quad & \{\texttt{ce rdf : type ConnectionElement.} \\ & \texttt{lue rdf : type LandUseElement.}\}\end{aligned}$$

the translation of the negation (`not`) of the filter expression is:

$$\tau_1 = \quad \{\tau_R \texttt{ FILTER NOT EXISTS } \tau_0\}$$

By applying the translation rules for the property predicates and conjunctions, we obtain the following translation for the body of the formula:

$$\tau_2 = \quad \{\{\texttt{ce sfOverlaps lue.}\}\ \tau_1\}$$

Finally, we translate the existential quantifier on *lue* and class restrictions on *lue*, *ce* using the associated rules:

$$\tau_3 = \quad \texttt{SELECT (ce) WHERE } \{\tau_R\ \tau_2\}$$

As already mentioned, we do not translate the first existential quantifier on *ce*, because this would leave no variables. According to equation 1, given that the original formula started with `not exists`, it is satisfied if query τ_3 returns $\emptyset$.

4.2 A Tool for Automated Translation and Query Execution

We developed a parser to convert GeCoLan formulas into GeoSPARQL queries with the ANTLR parser generator (http://www.antlr.org/), which was chosen due to its power and relative simplicity of use. Based on the formal specification of the grammar of the GeCoLan language, ANTLR generated a generic parser based on the Visitor design pattern (Gamma et al., 1994). We then extended this parser to produce the correct GeoSPARQL fragments according to the grammar rules. After translating a formula, the tool submits it to the Parliament Triple Store through the Jena (https://jena.apache.org/) library for execution. The result returned by the Triple Store is then interpreted in order to determine whether the data in the Triple Store satisfies the EN specification or not.

The geospatial data we used was obtained from a series of ESRI shapefiles produced in project (Città Metropolitana di Torino, 2014), which we converted to RDF using GeoTriples (Kyzirakos et al., 2014). That tool allows the user to define an appropriate mapping in order to associate shapefiles attributes to RDF properties. Figure 5 shows part of the map we used, where land use elements are colored according

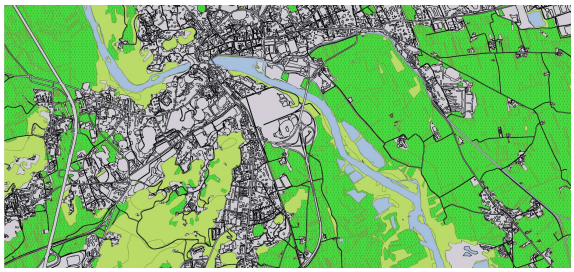

Figure 5: A portion of the map of the area covered by the Local EN proposal.

to their first level LCP type, e.g., artificial land is gray, water bodies are light blue, and so forth.

We uploaded to Parliament all the project data located within a circle of 10km diameter centered at a given point on the map of municipalities near Turin. Overall, the data consisted of 183,752 triples. More specifically, within the considered circle there are 5162 *LandUseElements*, over which lay the elements of the EN: i.e., 579 *StructuralElements* and 1054 *ConnectionElements*.

We first tested the **Connection Elements** example from section 3.3, whose translation has just been illustrated. The query is challenging because it could require to consider all the pairs formed by a *LandUseElement* and a *ConnectionElement*. Overall, our Triple Store contains almost 5.5M such pairs, and complex operations had to be performed on each pair, such as checking whether the *LandUseElement* and *ConnectionElement* geometries overlapped.

The execution of the query took about 5.5 minutes on a medium-end laptop, pointing out that (according to the data in our Triple Store), 595 *ConnectionElements* of the EN intersected at least a *LandUseElement* of maximum irreversibility or extroversion. This means that, in the project data, more than 55% of the *ConnectionElements* (595 out of 1054) violate the specification. The evaluation of whether these are severe violations of the considered specification, or acceptable small overlaps with forbidden areas, is left to the human user.

We also tested the **Buffer Zones** example from section 3.3. The translation of the specification is complex and yields a GeoSPARQL query with four (partially nested) `SELECT`s and 44 lines. However, the execution took only a few seconds before answering that there are no violating *Buffer Creation*s in our data. The impressive efficiency of this query is explained by the fact that the number of pending *Intervention*s of type *Buffer Creation* is usually small at a given time; e.g., in the dataset we considered, there was only one such a pending intervention. The number of *LandUseElement*s and *StructuralElement*s that must be considered by the query is therefore strongly limited by the fact that they must be "close" to the *Buffer Creation*s; i.e., they touch, or overlap with them.

5 FUTURE WORK

The study and implementation of additional reasoning tasks based on GeCoLan represents one of the most important directions of our future work.

Whereas, at the current stage, we implemented the validation as a stand-alone prototype, the main motivation and application of our work lies in its possible integration within Participatory Geographical Information Systems (PGIS), in order to support online interaction with stakeholders in inclusive processes aimed at collecting feedback and project proposals from stakeholders. This would be a novel feature of PGIS (e.g., see Ushahidi (ushahidi.com), PlanYourPlace (planyourplace.ca) and OnToMap (ontomap.ontomap.eu)), which only support feedback collection, and fail to provide validation functions to check the feasibility of the proposed actions. Specifically, in order to facilitate the convergence towards mutually agreed and feasible plans, it might be interesting to know how far the constraints on land usage are satisfied in a certain area, or whether any hypothetical actions, e.g., related to redevelopment project plans, are compatible with them. The present work is suitable for answering this kind of question by proposing a model for verifying the compliance of a set of geo-data with specifications concerning the related geographical area. Although in this paper we focus on the preservation and reconstruction of the Ecological Networks, this type of model can be applied in rather different contexts, including the management of city plans, in which detailed constraints have to be satisfied when building new elements or remodeling existing ones.

Another line of research we will pursue concerns the improvement of the efficiency of validity checks. While our prototype implementation has shown the feasibility of the approach in a domain of realistic size, we believe there are many different ways to improve its scalability: firstly, we would like to consider if and how the GeoSPARQL queries produced by our translator can be automatically optimized in order to better exploit the operational semantics of GeoSPARQL and of its implementation(s); secondly, it would be worth to explore whether there exist any Triple Store that supports GeoSPARQL queries more efficiently than Parliament; finally, pre-computing and storing certain information (e.g., which elements intersect with which) may turn out to be advantageous

for the subsequent check of specifications.

6 CONCLUSIONS

Ecological Networks (ENs) describe the structure of existing real ecosystems and to plan their expansion, conservation and improvement. In this paper we proposed an OWL ontology for the representation of ENs. Moreover, we presented the GeCoLan language for expressing specifications about Ecological Networks planning, which allows for the automatic validation of OWL-based representations of networks against existing land-usage restrictions. GeCoLan can be automatically translated to GeoSPARQL queries for implementing the validation checks in an efficient way. Moreover, the language can support not only the verification of constraints in a geographical area, but also other reasoning tasks, such as making constructive suggestions, possibly optimizing some desired measure.

ACKNOWLEDGEMENTS

This work is partially funded by project MIMOSA (MultIModal Ontology-driven query system for the heterogeneous data of a SmArtcity, “Progetto di Ateneo Torino_call2014_L2_157”, 2015-17) and by “Ricerca Locale” of the University of Torino.

REFERENCES

Ajit, S., Sleeman, D., Fowler, D. W., and Knott, D. (2008). Constraint capture and maintenance in engineering design. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 22(04):325–343.

Angles, R. and Gutierrez, C. (2008). The expressive power of sparql. In *Proc. International Semantic Web Conference*, pages 114–129.

Arenas, M., Gottlob, G., and Pieris, A. (2014). Expressive languages for querying the semantic web. In *Proceedings of the 33rd ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 14–26. ACM.

Bassiliades, N. and Gray, P. (1995). Colan: A functional constraint language and its implementation. *Data & Knowledge Engineering*, 14(3):203–249.

Battle, R. and Kolas, D. (2012). Enabling the geospatial Semantic Web with Parliament and GeoSPARQL. *Semantic Web*, 3(4):355–370.

Bennett, G. and Mulongoy, K. (2006). Review of experience with ecological networks, corridors and buffer zones. *Technical Series*, 23.

Bennett, G. and Wit, P. (2001). The development and application of ecological networks: a review of proposals, plans and programmes. *AIDEnvironment*.

Boley, H., Dean, M., Grosof, B., Sintek, M., Spencer, B., Tabet, S., and Wagner, G. (2004). FOL RuleML: The first-order logic web language. *Document located online at http://www. ruleml. org/fol*.

Boley, H., Tabet, S., and Wagner, G. (2001). Design rationale of ruleml: A markup language for semantic web rules. In *Proceedings of the First International Conference on Semantic Web Working*, pages 381–401. CEUR-WS. org.

Città Metropolitana di Torino (2014). Misura 323 del psr 2007-2013 (in italian). http://www.cittametropolitana.torino.it/cms/territorio-urbanistica/misura-323/misura-323-sperimentale.

Cohn, A., Bennett, B., Gooday, J., and N., G. (1997). Qualitative spatial representation and reasoning with the region connection calculus. *GeoInformatica*, 1(3):275–316.

Council of Europe (2000). General guidelines for the development of the pan-european ecological network. *Nature and environment*, 107.

Egenhofer, M. (1989). A formal definition of binary topological relationships. In *Proc. Int. Conf. on Foundations of Data Organization and Algorithms*, pages 457–472.

Fath, B., Sharler, U., Ulanowicz, R., and Hannon, B. (2007). Ecological network analysis: network construction. *Trends in Ecology & Evolution*, 208:49–55.

Fürber, C. and Hepp, M. (2010). Using sparql and spin for data quality management on the semantic web. In *Proc. Business Information Systems*, pages 35–46.

Gamma, E., Helm, R., Johnson, R., and Vlissides, J. (1994). *Design patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley.

Gobluski, A., Westlund, E., Vandermeer, J., and Pascual, M. (2016). Ecological networks over the edge: Hypergraph trait-mediated indirect interaction (TMII) structure. *Trends in Ecology & Evolution*, 31(5):344–354.

Grau, B., Horrocks, I., Motik, B., Parsia, B., Patel-Schneider, P., and Sattler, U. (2008). OWL 2: The next step for OWL. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(4):309–322.

Gray, P., Hui, K., and Preece, A. (2001). An expressive constraint language for semantic web applications. In *E-Business and the Intelligent Web: Papers from the IJCAI-01 Workshop*, pages 46–53.

Horrocks, I., Patel-Schneider, P., Boley, H., Tabet, S., Grosof, B., and Dean, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member submission*, 21.

Jongman, R. (1995). Nature conservation planning in europe: developing ecological networks. *Landscape and urban planning*, 32(3):169–183.

Kessler, C., Raubal, M., and Wosniok, C. (2009). Semantic rules for context-aware geographical information retrieval. In *Proceedings of the 4th European Conference on Smart Sensing and Context*, EuroSSC’09, pages 77–92. Springer-Verlag.

Kostylev, E., Reutter, J., and Ugarte, M. (2015). Construct queries in sparql. *LIPIcs-Leibniz International Proceedings in Informatics*, 31.

Krötzsch, M. (2010). *Description Logic Rules*, volume 8 of *Studies on the Semantic Web*. IOS Press.

Kyzirakos, K., Vlachopoulos, I., Savva, D., Manegold, S., and Koubarakis, M. (2014). Geotriples: A tool for publishing geospatial data as RDF graphs using R2RML mappings. In *Proceedings of ISWC 2014 - Posters and Demonstrations Track - Volume 1272*, ISWC-PD'14, pages 393–396, Aachen, Germany, Germany. CEUR-WS.org.

Louwsma, J., Zlatanova, S., van Lammeren, R., and van Oosterom, P. (2006). Specifying and implementing constraints in GIS - with examples from a geo-virtual reality system. *GeoInformatica*, 10(4):531–550.

Lurgi, M. and Robertson, D. (2011). Automated experimentation in ecological networks. *Automated Experimentation*, 3(1).

OGC (2012). Geosparql vocabulary. http://schemas.opengis.net/geosparql/1.0/geosparql_vocab_all.rdf.

Open Geospatial Consortium et al. (2011). *OpenGIS Implementation Standard for Geographic information-Simple feature access-Part 1: Common architecture*.

Perry, M. and Herring, J. (2012). GeoSPARQL - a geographic query language for RDF data. *OGC Implementation Standard. Sept*.

Pilosof, S., Porter, M., Pascual, M., and Kefi, S. (2017). The mulutilayer nature of ecological networks. *Nature ecology&evolution*, 1:article 101.

Provincia di Torino (2014). Linee guida per le reti ecologiche (in italian). http://www.provincia.torino.gov.it/territorio/file-storage/download/pdf/pian_ territoriale/rete_ecologica/ lgsv_lgre.pdf.

Ulanowicz, R. (2004). Quantitative methods for ecological network analysis. *Computational biology and chemistry*, 28:321–339.

W3C (2005). A Proposal for a SWRL Extension towards First-Order Logic. https://www.w3.org/Submission/SWRL-FOL/.

W3C (2012). Web ontology language (OWL). https://www.w3.org/TR/owl2-overview/.

W3C (2013). SPARQL query language for RDF. https://www.w3.org/TR/rdf-sparql-query/.

W3C (2014). Resource description framework (RDF). https://www.w3.org/RDF/.