Checking Licenses Compatibility between Vocabularies and Data

Guido Governatori[1], Ho-Pun Lam[1], Antonino Rotolo[2], Serena Villata[3], Ghislain Atemezing[4], and Fabien Gandon[3]

[1] NICTA Queensland[‡]
firstname.lastname@nicta.com.au
[2] University of Bologna
antonino.rotolo@unibo.it
[3] INRIA Sophia Antipolis
firstname.lastname@inria.fr
[4] Eurecom
auguste.atemezing@eurecom.fr

Abstract In the Web of Data, licenses specifying the terms of use and reuse are associated not only to datasets but also to vocabularies. However, even less support is provided for taking the licenses of vocabularies into account than for datasets, which says it all. In particular, this paper addresses the following issue: checking the compatibility among the set of licenses assigned to the vocabularies used to constitute a dataset, and the license that is intended to be associated to the dataset itself. We provide a framework called LIVE able to support data publishers in such compatibility checking step, taking into consideration both the licenses associated to the vocabularies and those assigned to the data.

1 Introduction

The license of a dataset in the Web of Data can be specified within the data, or outside of it, for example in a separate document linking the data. In line with the Web of Data philosophy [11], licenses for such datasets should be specified in RDF, for instance through the Dublin Core vocabulary[1]. In the latest years, a number of approaches have been proposed to model licenses in RDF, define licensing patterns [16,15], deal with self-referential licenses [12], or define composite licenses for query results containing heterogeneously licensed material [10]. Despite such approaches, still a lot of effort is needed to enhance the association of licenses to data on the Web, and to process licensed material possibly in an automated way. The scenario becomes even more complex when another essential component in the Web of Data is taken into account: the vocabularies. Several open issues arise concerning licensed vocabularies, and as a consequence, the (possibly licensed) data defined through such vocabularies. Consider for instance the following scenario: a data producer chooses a vocabulary V to create his RDF dataset, and such vocabulary is associated with a license l_1 prohibiting commercial reuse of the

[‡] NICTA is funded by the Australian Government and the Australian Research Council.

[1] http://purl.org/dc/terms/license

data (Non-Commercial license); then the data publisher, after creating her own dataset, assigns it to an open license l_2. What is the connection between these two licenses? Are they compatible? When only one vocabulary is used, then answering such questions may appear straightforward, but this is not the case when several licensed vocabularies are used together to build a dataset. All these issues have to be addressed in an automated way, such that the data provider is supported in assigning a license to her dataset and verifying its compatibility with the licenses associated to the adopted vocabularies. The research question we answer in this paper is *how to develop an automated framework that verifies the compatibility of the licenses associated to the exploited vocabularies with respect to the dataset license?*

We present an online framework called LIVE[2] (LIcenses VErification) that exploits and extends the formal approach to licenses composition proposed in [10,7] with the aim to verify the compatibility of a set of heterogeneous licenses. LIVE, after retrieving the licenses associated to the vocabularies used in the dataset under analysis, supports data providers in verifying whether the license assigned to the dataset is compatible with those of the vocabularies, and returns a warning when this is not the case.

The remainder of the paper is as follows. Section 2 discusses the existing literature about licenses. Section 3 summarizes the main insights of the logic for licenses compatibility we adopt, and Section 4 presents the framework whose time performances evaluation is provided in Section 5.

2 Related Work

In the Web scenario, a number of works address the problem of representing and/or reasoning over licensing information. Iannella presents the Open Digital Rights Language (ODRL) [3] for expressing rights information over content. Gangadharan et al. [3] address the issue of service license composition and compatibility analysis basing on ODRL-S. Truong et al. [18] address the issue of analyzing data contracts, based on ODRL-S again. Krotzsch and Speiser [12] present a semantic framework for evaluating ShareAlike recursive statements. Gordon [4] presents a legal prototype for analyzing open source licenses compatibility using the Carneades argumentation system. Finally, Rodiguez-Doncel et al. [15] propose the License Linked Data Resources pattern which provides a solution to describe existing licenses and rights expressions both for open and not open scenarios. All these works either propose new ways to model licenses information or new formal frameworks to deal with rights. In this paper, we do not address none of these issues, and we adopt the formal framework proposed in [10]. Also Pucella and Weissman [14] propose a logic to check whether the user's actions follow the licenses' specifications. However, as they do not deal with compatibility, do not provide a deontic account of licenses' conclusions, and their logic is not able to handle conflicting licenses, we choose and adapt the deontic logic of [10], which better suits our needs. Up to our knowledge, the issue of licensed vocabularies has never been addressed. More precisely, no available framework deals with such licenses and verifies in an automated way their potential compatibility with the license associated to datasets.

[2] The online tool is available at `http://www.eurecom.fr/~atemezin/licenseChecker/`

[3] `http://odrl.net/1.1/ODRL-11.pdf`

3 Licenses compatibility evaluation: the logic

In this section, we introduce the defeasible deontic logic we rely on to check the compatibility among licenses. The logic is basically the one developed in [5,6,17,10], i.e., it is an instance of the general multi-modal defeasible logic presented in these works. However, the formal language is different because we have here deontic modalities for each license, but the proof theory is technically the same. The logic presented in this paper is an extension of the logic proposed in [10] and later extended in [7]. The main conceptual difference between the logic presented in [10,7] and the one introduced in this paper is the following: the former is used to compute the deontic components of a so-called *composite* licenses which aggregates the components of all the licenses to be composed, ensuring the absence of possible inconsistencies; while the latter is used to verify the compatibility among different licenses, and the result is a yes/no answer, not a license, i.e., the composite one. In addition to these differences concerning the formal framework, the originality of the paper lies in the design and implementation of the online LIVE framework that verifies in an automated way the compatibility of the licenses associated to the vocabularies and the one associated to the selected dataset.

Furthermore, the reader may argue why we need deontic logic to compute the compatibility among licenses. First, deontic logic is needed because without it we cannot distinguish between `Obligatory(Action)` and `Forbidden(Action)`. Second, dealing with licenses compatibility requires reasoning about all deontic provisions, handling and solving normative conflicts arising from deontically incompatible licenses, and exceptions. A few formalisms can do that, and defeasible deontic logic is one of the best candidates, i.e., all aspects are managed in an efficient and computationally tractable way, as we discussed in [10].

Basic concepts and formal language As described in [10], reasoning about licenses compatibility requires considering two components:

Factual and ontology component: the first component is meant to describe the facts with respect to which Web of Data licenses are applied as well as the ontology of concepts involved by licenses;

Deontic component: the second component aims at capturing the deontic aspects of Web of Data licenses, thus offering mechanisms for reasoning about obligations, prohibitions, and permissions in force in each license.

Our interest mainly focuses on checking the deontic compatibility of licenses, even though, for the sake of completeness, we present the proposed method by also handling, in standard Defeasible Logic, the factual and ontology component, as done in [2,17][4]. We assume that all licenses share a same ontology, or that the ontologies are aligned.

The formal language of the logic is rule-based. Literals can be plain, such as $p, q, r \ldots$, or modal, such Op (obligatory), Pp (permitted), and Fp (forbidden/prohibited). Ontology rules work as regular Defeasible Logic rules for deriving plain literals, while the logic of deontic rules provide a constructive account of the basic deontic modalities (obligation, prohibition, and permission). However, while we assume that all licenses share a same

[4] Standard Defeasible Logic is just an option, and the factual and ontology component can be handled in any other suitable logic and by resorting to a separate reasoner.

ontology, the purpose of the formalism is mainly to establish the conditions to derive *different* deontic conclusions from *different* licenses, and check whether they are compatible. Hence, we need to keep track of how these deontic conclusions are obtained. To this purpose, deontic rules (and, as we will see, their conclusions) are parametrized by labels referring to licenses. An ontology rule such as $a_1,\ldots,a_n \Rightarrow b$ supports the conclusion of b, given $a_1,\ldots,a_n$, and so it states that, from the viewpoint of any license any instance enjoying $a_1,\ldots,a_n$ is also an instance of b. On the contrary, rules as $a,\mathrm{O}b \Rightarrow^{l_2}_{\mathrm{O}} p$ state that, if a is the case and b is obligatory, then $\mathrm{O}p$ holds in the perspective of license l_2, i.e., p is obligatory for l_2. The proof theory aims at offering an efficient method for reasoning about the deontic component of each license and, given that method, for checking their compatibility. The idea is simply to compute the set of all conclusions for each license and then check whether incompatible conclusions are obtained.

In this paper, we adopt the logical structure of the licenses proposed in the ODRL vocabulary[5]. This means that each license (i.e., `Policy`) is composed by a number of *rules* (i.e., `Duty`, `Permission`, `Prohibition`) that describe the *actions* constrained by the license (e.g., `attribute`, `sell`, etc). We have mapped the three kinds of rules with the respective deontic modalities, and each action represents a literal in the logic. The dataset of licenses we exploit in LIVE has been manually built using such vocabulary.

The basic language is as follows. Let $\mathrm{Lic} = \{l_1, l_2, \ldots, l_n\}$ be a finite set of licenses. Given a set PROP of *propositional atoms*, the set of *literals* Lit is the set of such atoms and their negation; as a convention, if q is a literal, $\sim q$ denotes the complementary literal (if q is a positive literal p then $\sim q$ is $\neg p$; and if q is $\neg p$, then $\sim q$ is p). Let us denote with $\mathrm{MOD} = \{\mathrm{O},\mathrm{P},\mathrm{F}\}$ the set of basic deontic modalities. The set ModLit of modal literals is defined as follows: i) if $X \in \mathrm{MOD}$ and $l \in \mathrm{Lit}$, then Xl and $\neg Xl$ are modal literals, ii) nothing else is a modal literal.

Every rule is of the type $r : A(r) \hookrightarrow^x_Y C(r)$, where: r is a unique identifier for the rule; $A(r) = \{a_1,\ldots,a_n\}$, the *antecedent* is a set literal if r is an ontology rule, and a set of modal literals and literals if r is a deontic rule; $C(r)$ the *consequent* is a literal; if r is a deontic rule $Y \in \mathrm{MOD}$ represents the type of conclusion obtained (We will see why we do not need rules for prohibitions and permissions) and $x \in \mathrm{Lic}$ indicates to which license the rule refers to; Y and x are not used for ontology rules.

The intuition behind the different arrows is the following. *Strict rules* have the form $a_1,\ldots,a_n \rightarrow^x_Y b$. *Defeasible rules* have the form $a_1,\ldots,a_n \Rightarrow^x_Y b$. A rule of the form $a_1,\ldots,a_n \rightsquigarrow^x_Y b$ is a *defeater*. Analogously, for ontology rules, where arrows do not have superscripts and subscripts. The three types of rules establish the strength of the relationship. Strict rules provide the strongest connection between a set of premises and their conclusion: whenever the premises are deemed as indisputable so is the conclusion. Defeasible rules allow to derive the conclusion unless there is evidence for its contrary. Finally, defeaters suggest that there is a connection between its premises and the conclusion not strong enough to warrant the conclusion on its own, but such that it can be used to defeat rules for the opposite conclusion.

A license theory is the knowledge base which is used to reason about the applicability of license rules under consideration.

[5] `http://www.w3.org/ns/odrl/2/`

Definition 1. *Let l be the name of any license. A* license theory *is a structure $D_l = (F, R^c, R^{O^l}, \prec)$, where $F \subseteq \text{Lit} \cup \text{ModLit}$ is a finite set of facts; R^c is a finite set of ontology rules; R^{O^l} is finite set of obligation rules; $\prec$ is an acyclic relation (called superiority relation) defined over $(R^c \times R^c) \cup (R^{O^l} \times R^{O^l})$.*

$R[b]$ and $R^X[b]$ with $X \in \{c, O^l\}$ denote the set of all rules whose consequent is b and of all rules (of type X). Given a set of rules R the sets R_s, R_{sd}, and R_{dft}, denote, respectively, the subsets of R of strict rules, strict and defeasible rules, and defeaters.

Proof theory A *proof* P of length n is a finite sequence $P(1), \ldots, P(n)$ of tagged literals of the type $+\Delta^X q$, $-\Delta^X q$, $+\partial^X q$ and $-\partial^X q$, where $X \in \{c, Y^l | l \in \text{Lic}, Y \in \text{MOD}\}$. The proof conditions below define the logical meaning of such tagged literals. As a conventional notation, $P(1...i)$ denotes the initial part of the sequence P of length i. Given a license theory D, $+\Delta^X q$ means that literal q is provable in D with the mode X using only facts and strict rules, $-\Delta^X q$ that it has been proved in D that q is not definitely provable in D with the mode X, $+\partial^X q$ that q is defeasibly provable in D with the mode X, and $-\partial^X q$ that it has been proved in D that q is not defeasibly provable in D with the mode X[6]. Given $\# \in \{\Delta, \partial\}$, $P = P(1), \ldots, P(n)$ *is a proof for* p in D for the license l iff $P(n) = +\#^l p$ when $p \in \text{Lit}$, $P(n) = +\#^{X^l} q$ when $p = Xq \in \text{ModLit}$, and $P(n) = -\#^{Y^l} q$ when $p = \neg Yq \in \text{ModLit}$. The proof conditions aim at determining what conclusions can be obtained within each license.

We concentrate here on deontic effects of licenses, thus working on the obligations, prohibitions, permissions entailed by any given license. Notice that the logic allows for deriving a deontic effect such as $O^l p$ by directly using obligations rules (i.e., a rule like $a_1, \ldots a_n \Rightarrow^l_O p$), or by *conversion* [5], i.e., by using an ontology rules where all antecedents are not modularized (they are plain literals), and are provable with the mode O^l. An example of conversion is the following: if *Enter_Personal_Data* $\Rightarrow$ *Identity_Disclosure* and $+\partial^{O^l}$*Enter_Personal_Data*, then we obtain $+\partial^{O^l}$*Identity_Disclosure*.

As usual with Defeasible Logic, we have proof conditions for the monotonic part of the theory (proofs for the tagged literals $\pm\Delta^Y p$) and for the non-monotonic part (proofs for the tagged literals $\pm\partial^Y p$). Since the proof theory for the ontology component ($\pm\Delta^c p$ and $\pm\partial^c p$) is the one for standard Defeasible Logic we will omit it and refer the reader to [1]. Let us first define the condition for monotonic derivations of the obligations in each license l.

$+\Delta^{O^l}$: If $P(n+1) = +\Delta^{O^l} q$ then,
1) $O^l q \in F$ or
2) $\exists r \in R_s^{O^l}[q]$:
$\forall a, Xb, \neg Yd \in A(r)$:
$+\Delta^c a$, $+\Delta^{X^l} b$, $-\Delta^{Y^l} d \in P(1..n)$, or
3) $\exists r \in R_s^c[q]$:
$A(r) \neq \emptyset$ and $\forall a \in A(r) : +\Delta^{O^l} a$.

$-\Delta^{O^l}$: If $P(n+1) = -\Delta^{O^l} q$ then,
1) $O^l q \notin F$ and
2) $\forall r \in R_s^{O^l}[q]$:
$\exists a \in A(r)$: $-\Delta^c a$ or
$\exists Xb \in A(r)$: $-\Delta^X b$ or
$\exists \neg Yd \in A(r)$: $+\Delta^Y d$, and
3) $\forall r \in R_s^c[q]$:
$A(r) = \emptyset$ or $\exists a \in A(r) : -\Delta^{O^l} a$.

[6] As we will see, we shall adopt a reading of permissions according to which they can only be defeasible. Hence, we will not define the cases $\pm\Delta^{Y^l} q$ where $Y = \text{P}$.

Definite proof conditions for prohibitions can be simply obtained (just consider the positive case):

$$+\Delta^{\mathrm{F}^l}: \text{ If } P(n+1) = +\Delta^{\mathrm{F}^l} q, \text{ then } +\Delta^{\mathrm{O}^l} {\sim} q \in P(1..n).$$

Let us now consider the defeasible derivations of obligations in any license l:

$+\partial^{\mathrm{O}^l}$: If $P(n+1) = +\partial^{\mathrm{O}^l} q$ then
(1)$+\Delta^{\mathrm{O}^l} q \in P(1..n)$ or
(2) (2.1) $-\Delta^{\mathrm{O}^l} {\sim} q \in P(1..n)$ and
(2.2) either
(2.2.1) $\exists r \in R^{\mathrm{O}^l}_{\mathrm{sd}}[q] : \forall a, Xb, \neg Yd \in A(r)$: $+\partial^c a, +\partial^{X^l} b, -\partial^{Y^l} d \in P(1..n)$, or
(2.2.2) $\exists r \in R^c_{\mathrm{sd}}[q] : A(r) \neq \emptyset$ and $\forall a \in A(r)$: $+\partial^{\mathrm{O}^l} a$, and
(2.3) $\forall s \in R^X[{\sim} q]$ either
(2.3.1) if $X = \mathrm{O}^l$ then $\exists a \in A(s)$ or $Xb \in A(s)$ or $\neg Y \in A(s)$:
$-\partial^c a \in P(1..n)$, or $-\partial^{X^l} b \in P(1..n)$, or $+\partial^{Y^l} d \in P(1..n)$; and
(2.3.2) if $X = c$ then $A(s) = \emptyset$ or $\exists a \in A(s) : -\partial^{\mathrm{O}^l} b \in P(1..n)$, or
(2.3.3) (2.3.3.1) $\exists t \in R^{\mathrm{O}^l}[q] : \forall a, Xb, \neg Yd \in A(t)$: $+\partial^c a, +\partial^{X^l} b, -\partial^{Y^l} d \in P(1..n)$, and
$t \prec s$, or
(2.3.3.2) $\exists t \in R^c[q] : A(t) \neq \emptyset$ and $\forall a \in A(t)$: $+\partial^{\mathrm{O}^l} a$, and $t \prec s$.

$-\partial^{\mathrm{O}^l}$: If $P(n+1) = -\partial^{\mathrm{O}^l} q$ then
(1)$-\Delta^{\mathrm{O}^l} q \in P(1..n)$ and
(2) (2.1) $+\Delta^{\mathrm{O}^l} {\sim} q \in P(1..n)$ or
(2.2) (2.2.1) $\forall r \in R^{\mathrm{O}^l}_{\mathrm{sd}}[q]$:
$\exists a \in A(r)$: $-\partial^c a \in P(1..n)$ or
$\exists Xb \in A(r)$: $-\partial^X b \in P(1..n)$ or
$\exists \neg Yd \in A(r)$: $+\partial^Y d \in P(1..n)$, and
(2.2.2) $\forall r \in R^c_{\mathrm{sd}}[q]$: either $A(r) = \emptyset$ or $\exists a \in A(r)$: $-\partial^{\mathrm{O}^l} a$, or
(2.3) $\exists s \in R^X[{\sim} q]$: either
(2.3.1) $X = \mathrm{O}^l$ and $\forall a, Xb, \neg Y \in A(s)$:
$+\partial^c a \in P(1..n)$, $+\partial^{X^l} b \in P(1..n)$, and $-\partial^{Y^l} d \in P(1..n)$; or
(2.3.2) $X = c$, $A(s) \neq \emptyset$ and $\forall a \in A(s) : +\partial^{\mathrm{O}^l} a \in P(1..n)$, and
(2.3.3) (2.3.3.1) $\forall t \in R^{\mathrm{O}^l}[q]$:
$\exists a \in A(t)$: $-\partial^c a \in P(1..n)$ or
$\exists Xb \in A(t)$: $-\partial^X b \in P(1..n)$ or
$\exists \neg Yd \in A(t)$: $+\partial^Y d \in P(1..n)$, or
$t \not\prec s$, and
(2.3.3.2) $\forall t \in R^c[q] : A(t) = \emptyset$ or $\exists a \in A(t)$: $-\partial^{\mathrm{O}^l} a$, or $t \not\prec s$.

As usual in standard Defeasible Logic, to show that a literal q is defeasibly provable we have two choices: (1) we show that q is already definitely provable; or (2) we need to argue using the defeasible part of a multi-license theory D. For this second case, some (sub)conditions must be satisfied. First, we need to consider possible reasoning chains in support of ${\sim} q$ with the modes l_c and X^{l^c}, and show that ${\sim} q$ is not definitely provable with that mode (2.1 below). Second, we require that there must be a strict or defeasible rule with mode at hand for q which can apply (2.2 below). Third, we must consider the

set of all rules which are not known to be inapplicable and which permit to get $\sim q$ with the mode under consideration (2.3 below). Essentially, each rule s of this kind attacks the conclusion q. To prove q, s must be counterattacked by a rule t for q with the following properties: i) t must be applicable, and ii) t must prevail over s. Thus each attack on the conclusion q must be counterattacked by a stronger rule. In other words, r and the rules t form a team (for q) that defeats the rules s.

The concept of permission is much more elusive. Here, we minimize complexities by adopting perhaps the two simplest options among those discussed in [9]. Such options model permissions either as obtained

1. when it is possible to show that the opposite obligations are not provable; or
2. from permissive norms with defeaters for obligations; a defeater like $a_1,\ldots,a_n \rightsquigarrow^{l}_{\mathrm{O}} q$ states that some q is permitted ($\mathrm{P}q$) in the license l, since it is meant to block deontic defeasible rules for $\sim q$, i.e., rules supporting $\mathrm{O}\sim q$.

The first type of permissions corresponds to the so-called weak permissions, according to which some q is permitted ($\mathrm{P}q$) because it can be obtained from the fact that $\neg q$ is not provable as mandatory [19]. The second type of permissions is just one way for modeling explicit permissive clauses for proving $\mathrm{P}q$ (strong permissions of q): for an extensive treatment of defeasible permissions, see [8]. This reading suggests that permissions are essentially defeasible. Let us only consider the positive cases:
Permission, version I (Weak Permission)

$$+\partial^{\mathrm{P}^l}\text{: If } P(n+1) = +\partial^{\mathrm{P}^l} q \text{ then } (1)\ -\Delta^{\mathrm{O}^l} \sim q \in P(1..n).$$

The first type of permission might be useful for combination for 'public domain' licenses, meaning, that unless explicitly obliged or forbidden data can be used freely.

Permission, version II (Strong Permission)

$+\partial^{\mathrm{P}^l}$: If $P(n+1) = +\partial^{\mathrm{P}^l} q$ then
(1) (1.1) $-\Delta^{\mathrm{O}^l} \sim q \in P(1..n)$ and
(2.2) either
(2.2.1) $\exists r \in R^{\mathrm{O}^l}_{\mathrm{dft}}[q] : \forall a, Xb, \neg Yd \in A(r)$: $+\partial^c a, +\partial^{X^l} b, -\partial^{Y^l} d \in P(1..n)$, or
(2.2.2) $\exists r \in R^{c}_{\mathrm{dft}}[q] : A(r) \neq \emptyset$ and $\forall a \in A(r)$: $+\partial^{\mathrm{P}^l} a$, and
(2.3) $\forall s \in R^{X}_{\mathrm{sd}}[\sim q]$ either
(2.3.1) if $X = \mathrm{O}^l$ then $\exists a \in A(s)$ or $Xb \in A(s)$ or $\neg Y \in A(s)$:
$-\partial^c a \in P(1..n)$, or $-\partial^{X^l} b \in P(1..n)$, or $+\partial^{Y^l} d \in P(1..n)$; and
(2.3.2) if $X = c$ then $A(s) = \emptyset$ or $\exists a \in A(s) : -\partial^{\mathrm{O}^l} b \in P(1..n)$, or
(2.3.3) (2.3.3.1) $\exists t \in R^{\mathrm{O}^l}_{\mathrm{dft}}[q] : \forall a, Xb, \neg Yd \in A(t)$: $+\partial^c a, +\partial^{X^l} b, -\partial^{Y^l} d \in P(1..n)$, and $t \prec s$, or
(2.3.3.2) $\exists t \in R^{c}_{\mathrm{dft}}[q] : A(t) \neq \emptyset$ and $\forall a \in A(t)$: $+\partial^{\mathrm{O}^l} a$, and $t \prec s$.

Let us define the concept of extension specifying for each license its deontic conclusions:

Definition 2. *The deontic* extension *of a License Theory D is the structure*

$$((+\Delta_{\mathrm{O}^l}, -\Delta_{\mathrm{O}^l}), (+\partial_{\mathrm{O}^l}, -\partial_{\mathrm{O}^l}), (+\partial_{\mathrm{P}^l}, -\partial_{\mathrm{P}^l})),$$

where $\pm\# = \{p : D \vdash \pm\# p\}$ *where* $\# \in \{\Delta_{\mathrm{O}^l}, \partial_{\mathrm{O}^l}, \partial_{\mathrm{P}^l}\}$.

The logic presented here is a variant of the one developed in [5,6]. Accordingly, results of soundness and linear computational complexity can be directly imported here [17,10]. Let us consider a simple example that illustrates some aspects of the proof theory:

$$
\begin{aligned}
F &= \{a, d, f\} \\
R^{\mathrm{O}^l} &= \{r_1 : a \rightsquigarrow^l_{\mathrm{O}} b, \quad r_2 : d \rightarrow^l_{\mathrm{O}} \sim e, \quad r_3 : a, f \Rightarrow^l_{\mathrm{O}} e, \quad r_4 : \sim e, f \Rightarrow^l_{\mathrm{O}} \sim d\} \\
R^c &= \{r_5 : b \rightsquigarrow d\} \\
\prec &= \{r_5 \prec r_4\}
\end{aligned}
$$

The fact d triggers r_2 thus supporting the strict conclusion $+\Delta^{\mathrm{O}^l} \sim e$, which blocks in turn the defeasible derivation of $\mathrm{O}^l e$ that could be obtained via r_3 (triggered by a, f). The conclusion $\mathrm{O}^l \sim e$ and the fact f fire rule r_4, which is however blocked by rule r_5. Notice that rule r_5 is made applicable by conversion through the conclusion $+\partial^{\mathrm{P}^l} b$, which is obtained via rule r_1.

4 The Framework

The input of the LIVE framework consists in the dataset (URI or VOiD[7]) whose license has to be verified. The framework is composed by two modules. The first module takes care of retrieving the vocabularies used in the dataset, and for each vocabulary, retrieves the associate license[8] (if any) querying the LOV repository[9]. The second module takes as input the set of licenses (i.e., the licenses of the vocabularies used in the dataset as well as the license assigned to the dataset) to verify whether they are compatible with each others. The result returned by the module is a *yes/no* answer where *no* raises a warning message to the data provider highlighting the presence of an incompatibility. Such result is returned to the data provider, which is invited to change the license associated to the dataset and check back again with the LIVE framework whether further inconsistencies arise. We now detail the two modules.

4.1 Retrieving licensing information from vocabularies and datasets

Two use-cases are taken into account: a SPARQL endpoint, or a VoID file in Turtle syntax. In the first use case, the tool retrieves the named graphs present in the repository, and then the user is asked to select the URI of the graph that needs to be checked. Having that information, a SPARQL query is triggered, looking for entities declared as `owl:Ontology`, `voaf:Vocabulary` or object of the `void:vocabulary` property. The final step is to look up the LOV catalogue to check whether they declare any license. There are two options for checking the license: *(i)* a *"strict checking"* using the filtering containing exactly the namespace of the submitted vocabulary, or *(ii)* a *"domain checking"*, where only the domain of the vocabulary is used in the filter option.

[7] `http://www.w3.org/TR/void/`

[8] Note that the LIVE framework relies on the dataset of machine-readable licenses (RDF, Turtle syntax) available at `http://purl.org/NET/rdflicense`

[9] `http://lov.okfn.org/`

This latter option is recommended in case only one vocabulary has to be checked for the license. In the second use case, the module parses a VoID file using a N3 parser for Javascript[10], and then collects the declared vocabularies in the file, querying again LOV[11] to check their licensing information. When the URIs of the licenses associated to the vocabularies and the dataset are retrieved, the module retrieves the machine-readable description of the licenses in the dataset of licenses.

4.2 Licenses compatibility verification

The logic proposed in Section 3 and the licenses compatibility verification process has been implemented using SPINdle [13] – a defeasible logic reasoner capable of inferencing defeasible theories with hundredth of thousand rules. As depicted in Figure 1,

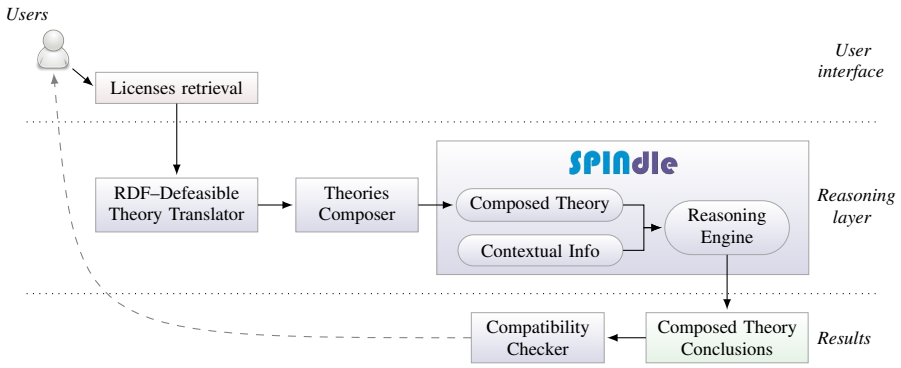

Figure 1. Licenses compatibility module.

after receiving queries from users, the selected licenses (represented using RDF) will be translated into the DFL formalism supported by SPINdle using the *RDF-Defeasible Theory Translator*. If, however, more than one license has been selected, then in order to verify the compatibility of different licensing terms, the translated defeasible theories will first be composed into a single defeasible theory based on the transformations described in Section 3. That is, each RDF-triple will be translated into a defeasible rule based on the subsumption relation between the *subject* and *object* of a RDF-triples. In our case, we can use the subject and object of the RDF-triples as the antecedent and head of a defeasible rule, respectively. Besides, the translator also supports direct import from the Web and processing of RDF data into SPINdle theories. The translated defeasible theories will then be composed into a single defeasible theory, based on the transformations described above, using the *Theories Composer*. Afterwards, the

[10] `https://github.com/RubenVerborgh/N3.js`

[11] Since LOV endpoint does not support the JSON format in the results, we have uploaded the data in `eventmedia.eurecom.fr/sparql`.

composed theory, together with other contextual information (as defined by user), will be loaded into the SPINdle reasoner to perform a compatibility check before returning the results to the users.

5 Evaluation

In this section, we address an evaluation of the time performances of the LIVE framework to retrieve the licensing information and checking the compatibility of the licenses. The LIVE framework is a Javascript application, combining HTML and Bootstrap. Hence, installation has no prerequisite. Since the tool is written in Javascript, the best way to monitor the execution time is with the `performance.now()` function. We use the 10 LOD datasets with the highest number of links towards other LOD datasets available at `http://lod-cloud.net/state/#links`. For each of the URLs in Datahub, we retrieve the VoID[12] file in Turtle format, and we use the *voidChecker* function[13] of the LIVE tool (Section 4.1).

First, we evaluate the time performances of the licenses compatibility module, we described in Section 4.2. Table 1 provides an overview of the obtained results checking of the licenses we found on LOV, considering the licenses to be verified (L_1, L_2), whether they are compatible or not, and the time performances (in ms). Time performances are about 6ms to compute the compatibility of the licenses.

Second, we evaluate time performances of the whole LIVE framework considering both the licenses retrieval module and the licenses compatibility one. Table 2 provides an overview of the time performances of the LIVE framework for the 10 LOD datasets with the highest number of links towards other LOD datasets. The first column (LicRetrieval) shows the time performances of the first module of LIVE, i.e., the performances in retrieving the vocabularies used in the dataset and their associated license, if any. The second column (vocabularies) shows the number of vocabularies declared in the VoID file for each dataset. The third column (LicCompatibility) presents the time performances of the second module of LIVE in verifying the compatibility of the retrieved licenses. Notice that the value of this column is equal to 0 if one of the following situations arise: *i)* there are no licensed vocabularies among those retrieved by the first module, and *ii)* there is only one license associated to one or more vocabularies and no license associated to the dataset. In these cases, no compatibility checking needs to be addressed. Finally, the fourth column (LIVE) shows the time performances of the whole module. The experimental evaluation has been carried out on the browser Chrome (Version 34). The results show that the LIVE framework provides the compatibility evaluation in less than 5s for 7 of the selected datasets. As it may be noticed for the last

L_1	L_2	**Compatibility**	**SPINdle module (ms)**
PDDL	OGL	Yes	6
PDDL	EUROSTAT	Yes	9
CC-BY	OGL	Yes	7
PDDL	CC0	Yes	4
CC-BY	ODBL	Yes	6
PDDL	CC-BY	Yes	8
CC-BY	CC0	Yes	3
CC-BY	CC-BY-SA	Yes	6

Table 1. Evaluation of SPINDle module.

[12] `http://www.w3.org/TR/void/`

[13] `http://www.eurecom.fr/~atemezin/licenseChecker/voidChecker.html`

two rows (i.e., for the cases where both the modules of LIVE are involved), the time performances of LIVE are mostly affected by the first module while the compatibility module does not produce a significant overhead. The last row of Table 2 is Linked Dataspaces[14], a dataset where we retrieve the licensing information in both the dataset and the vocabularies. In this case, LIVE retrieves in 13.20s 48 vocabularies, the license for the dataset is CC-BY, and the PDDL license is attached one of the vocabularies[15]. The time for verifying the compatibility is 8ms, leading to a total of 13.21s.

Dataset	LicRetrieval(ms)	vocabularies	LicCompatibility(ms)	LIVE(ms)
rkb-explorer-dblp	4 499	1	0	4499
rkb-explorer-southampton	14 693	1	0	14 693
rkb-explorer-eprints	3 220	1	0	3 220
rkb-explorer-acm	3 007	1	0	3 007
rkb-explorer-wiki	14 598	1	0	14 598
rkb-explorer-rae2001	3 343	1	0	3 343
rkb-explorer-citeseer	2 760	1	0	2 760
rkb-explorer-newcastle	3 354	1	0	3 354
rkb-explorer-kisti	4 094	5	6	4 100
270a.info	13 202	48	8	13 210

Table 2. Evaluation of the LIVE framework.

6 Conclusions

We have introduced the LIVE framework for licenses compatibility. The goal of the framework is to verify the compatibility of the licenses associated to the vocabularies exploited to create a RDF dataset and the license associated to the dataset itself. The final aim is to support data providers in assigning the "correct" license to a RDF dataset. Despite existing works about licenses in the Web of Data, and several discussions on the Web about the compatibility of different licensing terms, there is no existing framework to support in an automated way the data producer in such compatibility verification.

Several points have to be taken into account as future work. First, licensing vocabularies opens new challenges from the legal point of view. Consider for instance the CC Attribution license. This license states that "You must give appropriate credit, provide a link to the license, and indicate if changes were made". What do these terms mean concerning licensed vocabularies? Let us consider them separately: *i)* giving an appropriate credit means that a data provider using a vocabulary to construct her own RDF dataset has to give credit to such vocabulary, but it is difficult to say whether the `prefix` element may be considered an appropriate credit, *ii)* no RDF dataset at the present stage provides a link to the license assigned to the exploited vocabularies, and *iii)* indicating if changes were made concerns more the extension of existing vocabularies where, however, no link to the license is usually provided and the appropriate credit meaning is confuse as discussed in point *(i)*. This issue has to be investigated together with legal experts of the Web scenario to provide a set of good practices for data and

[14] `http://270a.info/`

[15] `http://purl.org/linked-data/cube`

vocabularies licensing on the Web of Data. Another issue concerns the "interpretation" of what a vocabulary is. More precisely, in the present paper we consider vocabularies as data, i.e., a set of triples. On the other side, this is not the only possible interpretation. For instance, we may see vocabularies as a kind of compiler, such that, after the creation of the dataset then the external vocabularies are no more used. In this case, what is a suitable way of defining a compatibility verification? We will investigate this issue as future work. Moreover, we plan to evaluate the usability of the online LIVE tool to subsequently improve the user interface.

References

1. Antoniou, G., Billington, D., Governatori, G., Maher, M.J.: Representation results for defeasible logic. ACM Trans. Comput. Logic 2, 255–287 (April 2001)
2. Boella, G., Governatori, G., Rotolo, A., van der Torre, L.: A logical understanding of legal interpretation. In: Proceedings of KR. AAAI Press (2010)
3. Gangadharan, G.R., Weiss, M., D'Andrea, V., Iannella, R.: Service license composition and compatibility analysis. In: Proceedings of ICSOC, LNCS 4749. pp. 257–269. Springer (2007)
4. Gordon, T.F.: Analyzing open source license compatibility issues with Carneades. In: Proceedings of ICAIL. pp. 51–55. ACM (2011)
5. Governatori, G., Rotolo, A.: BIO logical agents: Norms, beliefs, intentions in defeasible logic. Autonomous Agents and Multi-Agent Systems 17(1), 36–69 (2008)
6. Governatori, G., Rotolo, A.: A computational framework for institutional agency. Artif. Intell. Law 16(1), 25–52 (2008)
7. Governatori, G., Lam, H.P., Rotolo, A., Villata, S., Gandon, F.: Heuristics for licenses composition. In: Proceedings of JURIX. pp. 77–86. IOS Press (2013)
8. Governatori, G., Olivieri, F., Rotolo, A., Scannapieco, S.: Computing strong and weak permissions in defeasible logic. J. Philosophical Logic 42(6), 799–829 (2013)
9. Governatori, G., Padmanabhan, V., Rotolo, A., Sattar, A.: A defeasible logic for modelling policy-based intentions and motivational attitudes. Logic J. of IGPL 17(3), 227–265 (2009)
10. Governatori, G., Rotolo, A., Villata, S., Gandon, F.: One license to compose them all - a deontic logic approach to data licensing on the web of data. In: International Semantic Web Conference (1). Lecture Notes in Computer Science, vol. 8218, pp. 151–166. Springer (2013)
11. Heath, T., Bizer, C.: Linked Data: Evolving the Web into a Global Data Space. Morgan & Claypool (2011)
12. Krötzsch, M., Speiser, S.: ShareAlike Your Data: Self-referential Usage Policies for the Semantic Web. In: Proceedings of ISWC, LNCS 7031. pp. 354–369. Springer (2011)
13. Lam, H.P., Governatori, G.: The making of SPINdle. In: Proceedings of RuleML, LNCS 5858. pp. 315–322. Springer (2009)
14. Pucella, R., Weissman, V.: A logic for reasoning about digital rights. In: Proceedings of CSFW. pp. 282–294. IEEE (2002)
15. Rodriguez-Doncel, V., Figueroa, M.S., Gomez-Perez, A., Villalon, M.P.: License linked data resources pattern. In: Proc. of the 4th International Workshop on Ontology Patterns (2013)
16. Rodriguez-Doncel, V., Figueroa, M.S., Gomez-Perez, A., Villalon, M.P.: Licensing patterns for linked data. In: Proc. of the 4th International Workshop on Ontology Patterns (2013)
17. Rotolo, A., Villata, S., Gandon, F.: A deontic logic semantics for licenses composition in the web of data. In: Proceedings of ICAIL. pp. 111–120. ACM (2013)
18. Truong, H.L., Gangadharan, G.R., Comerio, M., Dustdar, S., Paoli, F.D.: On analyzing and developing data contracts in cloud-based data marketplaces. In: Proceedings of APSCC, IEEE. pp. 174–181 (2011)
19. von Wright, G.: Norm and action: A logical inquiry. Routledge and Kegan Paul (1963)