Automated Legal Assessment with OWL2

Saskia van de VEN a,1, Joost BREUKER a and Rinke HOEKSTRA a and Lars WORTEL a

a Leibniz Center for Law, University of Amsterdam, The Netherlands

Abstract. In this paper we describe the heart of HARNESS, a legal knowledge system that is developed in the ESTRELLA project. This heart consists of a reasoning engine that performs legal assessment tasks, i.e. determining whether some case violates and/or complies with legal norms. This assessment module uses three knowledge bases cast in OWL 2: a domain ontology, a set of norms and a case description. It is explained how the sound and complete reasoning provided by OWL is exploited by a careful representation of norms, using examples from law on taxation of gifts. In Section A plugin for Protégé – a workbench for OWL – is described that enables easy experimentation for this reasoning engine, powered by Pellet.

Keywords. legal assessment, OWL, law, exception hierarchy

Introduction

In this paper we describe the heart of HARNESS 2, which is the working title of a (legal) knowledge-based system that is developed within the ESTRELLA project. 3. This heart consists of a reasoning engine that performs legal assessment tasks, i.e. determining whether some case violates and/or complies with legal norms (which we describe in Section 2). In Section 1 we discuss the (dis)advantages of using OWL 2 to implement this assessment module. The assessment module uses three knowledge bases: a domain ontology, a set of norms and a case description. It is explained how the sound and complete reasoning provided by OWL is exploited by a careful representation of norms, using examples from law on taxation of gifts (Section 3). A plugin for Protégé – a workbench for OWL – is described that enables easy experimentation for this reasoning engine, powered by Pellet.

1. Using OWL 2 for reasoning with an ontology and with norms

Legal assessment concerns the problem of determining whether some case is allowed or disallowed given an appropriate body of legal norms. It involves the expression of an
individual case, an interpretation of that case in terms of world knowledge, and the application of normative knowledge to determine whether the case constitutes a norm violation. Legal assessment is the core of practical legal reasoning, i.e. all typical legal reasoning tasks, such as designing norms (legislation, contracts) or planning legal actions or constructing legal arguments have legal assessment as a subtask. Usually, in legal knowledge systems, norms are represented by rules and an explicit ontology is not available. In HARNESS we have two knowledge bases: 1) a domain ontology and 2) a set of norms representing the normative articles. Both are cast in OWL-DL 2. Assessment consists of classifying the individuals and properties making up a case description in terms of both the ontology and the norms simultaneously. The result tells whether norms are violated or not.

What are the advantages of this approach? The most important one is that the knowledge provided by the ontology is fully expanded, i.e. new properties are derived for the individuals making up the case description. For instance, it may be derived that if car A is to the left of car B, B is to the right of A, etc. By combining the reasoning with ontologies with the norms in one process no information is lost or neglected. There were more reasons why we made this combination. Originally we intended to have the reasoning with the ontology performed by a classifier (Pellet), and then submit this 'enhanced' case description to a rule engine that would apply norms in a rule format. In other words we aimed at a hybrid architecture, as the acronym HARNESS shows. However, it was difficult to obtain all inferences from Pellet as a separate file. Moreover, the rule engine to be used had have the property of being ‘DL-Safe’ [7], which means that the rule formalism should be rather restricted in expressiveness. Another advantage is that in the same process, Pellet will also detect which norms are exceptions to other norms. One gets this for free, where in (defeasible) rule formalisms one has to specify explicitly these exceptions (e.g. by an ‘unless’ construct). Finally, and probably as important was the fact that using OWL-DL 2 and Pellet would guarantee reasoning that was proven to be sound and decidable. Validity of reasoning then becomes only dependent on the quality of modeling.

This is also the price to pay: as we will show below, modeling norms in OWL 2 is less self evident than by using rules, even when an important burden of tracing exceptions is no longer required. It should be noted that this process of tracing exceptions is difficult for humans, as errors in legal drafting easily show. For that reason we can see this compact HARNESS kernel as having the same role as a (in former days) a calculator in engineering: it does not only make calculations faster but also fully consistent.

2. Modeling legal assessment

Performing legal assessment involves the following steps:

**Input of the case** Create a description of the case that needs to be assessed. Typically the facts that are provided should correspond to terms are used in the norms, i.e. both the norms and the individual case should commit to the same domain ontology.

**Matching** The individual case is matched against a set of norms. This is done by classifying parts of the case description as instances of one or more norms.
Resolving conflicts Exceptions between norms can be traced by classifying norms. A norm that is subsumed by another norm, but has a different deontic qualification (e.g. a permission vs an obligation) can be classified as an exception. This classification ‘trick’ enables us to solve this logical conflict in a monotonic way.

Conclusion Provide an outcome of the normative qualification of the case. It is desirable to present an explanation, e.g. why a certain norm applies to the given input, showing which terms in a norm match the individual facts of the case. Moreover the answer should be justified by reference to the norms and original source.

Three knowledge bases are needed for this task. A domain ontology that defines the concepts used as the basic terminology for case descriptions and norms. The norms represent the regulations. The individual case description represents the facts that describe a situation and is expressed as a set of related OWL individuals.

3. Representing and reasoning with norms

We will illustrate our approach by means of a real-world example of the Hungarian regulations dealing with duties on gifts. In modeling this domain we focused a part of section 11, which states the following:

(1) The following shall be subject to duty payment on gifts:
   a) gifts of real property,
   b) gifts of movable property,
   c) gratuitous creation of a right of pecuniary value, surrender of such right or the exercise thereof without consideration, and the waiver of such right without consideration.

(2) The gifts listed under Subsection (1) shall be subject to the payment of duty only if duly documented, or, in respect of movable property, if the market value of the movable property granted to any one donee exceeds 150 000 HUF even not documented. Such gifts shall be reported to the state tax authority within 30 days in accordance with the provisions set forth in Subsections (3)-(4) of Section 91.

(3) Rights of pecuniary value given as a gift shall not be subject to the payment of duty if retained by the donor for his own benefit, or if the real property is gifted as encumbered with some rights of pecuniary value already existing, and registered in the real estate register, prior to the transfer.

Our use case deals with a donation of a copyright. Phil donates the copyright on his book to Mary. This donation is documented, but is retained by Phil for his own benefit:

\[
\text{Donation}(\text{donation\_of\_copyright}) \land \text{Documented}(\text{donation\_of\_copyright}) \land \\
\text{donee}(\text{donation\_laptop, mary}) \land \text{Person}(\text{mary}) \land \\
\text{donor}(\text{donation\_laptop, phil}) \land \text{Person}(\text{phil}) \land \\
\text{retained\_by}(\text{donation\_laptop, phil})
\]

Representing norms We represent a norm as a deontic qualification of a generic case [8] (GC), which is a conjunction of conditions that together form a description of the situation expressed by the norm. A GC \( \Sigma \) is defined as a set of conditions \( \{\sigma_1, \ldots, \sigma_n\} \) in conjunctive normal form. Conditions are class axioms composed of classes and proper-
ties defined by the domain ontology. An individual case $C$ is a set of grounded propositions $\{c_1, ..., c_n\}$, called circumstances, that describe a certain state of affairs. The match between a GC and an individual case is through realisation, i.e. a specific form of classification.

GC themselves can also form class hierarchies. A more specific case $GC_2$ will be subsumed by $GC_1$. This hierarchy actually reflects the lex specialis exception relations between norms. Moreover we can apply the following role inclusion axioms to make these exception relations explicit:

$$\text{allows} \circ \text{disallowed_by} \rightarrow \text{exception}$$

$$\text{disallows} \circ \text{allowed_by} \rightarrow \text{exception}$$

This exception class indicates the fact that there is in principle a conflict, as the deontic qualifiers that are associated with the GCs are different ones. OWL-DL is a monotonic reasoner that cannot handle such inconsistencies directly, but that is not necessary as a more detailed semantic analysis of these qualifiers shows [5]. We make the distinction between a (deontic) qualification that qualifies a norm and the qualification a case gets – whether generic or individual. The main definitions are the following ones:

\[
\begin{align*}
\text{Norm} \sqsubseteq \text{Qualification} & \quad \sqcap \text{qualifies some} \text{Normatively_Qualified} \\
\text{Permission} \sqsubseteq \text{Norm} & \quad \equiv \text{allows some} \text{Allowed} \quad \sqcap \text{allows only} \text{Allowed} \\
\text{Obligation} \sqsubseteq \text{Permission} & \quad \equiv \text{allows some} \text{Obliged} \quad \sqcap \text{disallows some} \text{Disallowed} \\
& \quad \sqcap \text{allows only} \text{Allowed} \quad \sqcap \text{disallows only} \text{Disallowed} \\
\text{Prohibition} & \equiv \text{Obligation}
\end{align*}
\]

The Prohibition and Obligation are equivalent because they are simply two different ways to put the same thing into words: a prohibition to smoke is an obligation not to smoke. The Permission is different in that it allows something, but does not prohibit anything. For the purpose of clarity we will refer to the subclasses of Normatively_Qualified as generic cases. The properties allows and disallows are disjoint sub properties of qualifies; a norm cannot simultaneously allow and disallow the same situation. Although a GC describes an entire situation, it can be represented as a single class description because the entities in a single situation should be connected. On the other hand, this means the

<table>
<thead>
<tr>
<th>Notion</th>
<th>Match</th>
<th>Default Allowed</th>
<th>Default Disallowed</th>
</tr>
</thead>
<tbody>
<tr>
<td>Prohibition</td>
<td>no</td>
<td>no effect</td>
<td>possible effect</td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>has effect</td>
<td>no effect</td>
</tr>
<tr>
<td>Permission</td>
<td>no</td>
<td>no effect</td>
<td>has effect</td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>no effect</td>
<td>has effect</td>
</tr>
<tr>
<td>Obligation</td>
<td>no</td>
<td>possible effect</td>
<td>no effect</td>
</tr>
<tr>
<td></td>
<td>yes</td>
<td>has effect</td>
<td>has effect</td>
</tr>
</tbody>
</table>

Table 1. Behaviour of deontic notions with respect to the default
norm’s qualification is biased to the class that forms the entry point for describing the GC.

Applying norms A case is matched when the classifier finds that some part of the case description can be classified as a subclass of the GC of some norm. This classification includes all ‘expansions’ that can be applied to the case description. So in our example, ‘copyright’ is according to the domain ontology a kind of ‘right-of-some-pecuniary-value’. Norm matching and identifying case terms are part of the same classification process.

The usual normative default in law is a (weak) permission, i.e. allowed in our terms. Table 1) provides an overview of the behaviour of the deontic notions with respect to the default situation. Most matches have simple and direct outcomes: matching permissions always yield ‘allowed’, etc. However there are two problematic occurrences listed in the table as “possible effect”. If there is no match with the GC of an obligation where the default is allowed, we might have failed to detect a violation of the default. In this case, the GC is partitioned into that part which is explicitly allowed (AΩ) and that part which is explicitly disallowed (DΩ) by the obligation. Moreover a norm is typically only addressed towards a certain element of the GC, given some context. We therefore distinguish between the context of a norm and that part of the GC which the norm is directed towards. For example, when an individual case does not match this specific part, but it does match its context, an obligation is violated. The design pattern we follow in the representation is therefore as follows, where the context is defined as Σ₁: σ₁ ∩ ... ∩ σₙ, and the obliged part of the GC is Σ₂: σₙ₊₁.

Of course, where the default situation is disallowed, the partitioning of the case is only necessary where failure to detect a prohibition is concerned. We will now take a look at how the norms relevant for the use case are modeled. Section 11-1c, combined

4 However, some regulations take as their default that everything is prohibited as a starting point. This is e.g. the case where the regulation gets the character of an instruction: for instance for emergency procedures, but the classical toy domain of a ‘library regulation has the same character.
Section 11-2 is modeled as follows:

\[
\begin{align*}
A_\Sigma & \equiv \sigma_1 \land \ldots \land \sigma_n \land \sigma_{n+1} \\
\Rightarrow & \equiv \Sigma_1 \land \Sigma_2 \\
O_\Sigma & \equiv \sigma_1 \land \ldots \land \sigma_n \\
\Downarrow & \equiv \Sigma_1
\end{align*}
\]

with Section 11-3, is modeled as follows:

\[
\begin{align*}
\text{GC}_{11\_1c\_P} & \sqsubseteq \text{Generic_case} \land \text{allowed_by_value} \text{section11\_1c} \\
& \equiv \text{gratuitous\_creation\_surrender\_or\_waiver\_of\_a\_right\_of}\_\text{pecuniary\_value} \land \text{Documented} \\
& \land \text{donor some Donor} \\
& \land \text{donee some Donee} \\
& \land \text{duty\_on\_transfer some Duty\_on\_gifts}
\end{align*}
\]

\[
\begin{align*}
\text{GC}_{11\_1c\_F} & \sqsubseteq \text{Generic_case} \land \text{disallowed_by_value} \text{section11\_1c} \\
& \equiv \text{gratuitous\_creation\_surrender\_or\_waiver\_of\_a\_right\_of}\_\text{pecuniary\_value} \land \text{Documented} \\
& \land \text{donor some Donor} \\
& \land \text{donee some Donee}
\end{align*}
\]

\[
\begin{align*}
\text{11\_1c\_Obligation} & \sqsubseteq \text{Obligation} \land \text{disallows only} \text{GC}_{11\_1c\_F} \\
& \land \text{allows only} \text{GC}_{11\_1c\_P} \\
& \equiv \{ \text{section11\_1c} \}
\end{align*}
\]

Section 11-1c expresses an obligation where the default situation is allowed. Therefore we partitioned the case into something that is permitted (the obligation) and something that is disallowed (the prohibition). The other relevant part for our usecase, Section 11-3, is modeled as follows:

\[
\begin{align*}
\text{GC}_{11\_3} & \sqsubseteq \text{Generic_case} \land \text{disallowed_by_value} \text{section11\_3} \\
& \equiv \text{gratuitous\_creation\_surrender\_or\_waiver\_of\_a\_right\_of}\_\text{pecuniary\_value} \land \text{Documented} \\
& \land \text{donor some Donor} \\
& \land \text{donee some Donee} \\
& \land \text{object\_of\_transfer some right\_of\_pecuniary\_value} \\
& \land \text{retained\_by some Donor}
\end{align*}
\]

\[
\begin{align*}
\text{11\_3\_Prohibition} & \sqsubseteq \text{Prohibition} \land \text{disallows only} \text{GC}_{11\_3} \\
& \equiv \{ \text{section11\_3} \}
\end{align*}
\]
Section 11-3 demonstrates what we call the identity problem of individuals very clearly: we would like to address the fact that both the donors (the one that donates vs. the one that retains) should concern the same individual.

3.1. The HARNESS Plugin

We are currently developing a Protégé 4 plugin, that assists users in the task of legal assessment. Currently this plugin can actually be seen as the embodiment of HARNESS. HARNESS prompts the user to select an OWL file containing the norms, and a file containing a case description. Once the “check” button is pressed, Pellet is called to perform classification and realisation on the given files (see Figure 2).

First HARNESS checks all individuals that exist in the file containing the case description. Depending on their classification as Allowed or Disallowed, they are displayed in respectively the “Allowed” and/or “Disallowed” column, shown in the top left part of the screen. A user can click on the individuals in those columns and retrieve an explanation.

HARNESS also generates a final verdict, which is especially interesting in situations where multiple norms apply to the individual case (if only one norm applies it is clearly the final verdict as well). If an individual is both Allowed and Disallowed, the plugin tries to resolve this conflict by checking whether one of the norms allowing or disallowing it contains an exception relationship with the other norm. The most specific one will be displayed in the appropriate “Verdict” column, shown in the top right of the screen. The user can again select the individual to view an explanation of the verdict. The outcome of the usecase is indeed that the individual creation_of_copyright is disallowed by 11-3, but allowed by 11-1c. The explanation indicates that 11-3 is the most specific norm, which makes the final verdict “Permitted”.

4. Conclusions and future work

HARNESS as it exists now implements our basic requirements, but there are still some things we would like to address. For instance, it might be interesting to enhance the explanation by allowing users to click on a certain individual case to get more information about it. Furthermore, we would like to enhance the explanations with forms of justifi-
cation, such as a reference to the original source of law. Preferably the original source of law is stored in the MetaLex/CEN format of [10] an open XML interchange format for legal and legislative resources. This enables a direct connection between the original legal source to (specific parts of) our model. This would also form a nice way to interact with the MetaVex editor, a WYSIWYG editor used to create and edit documents in the MetaLex/CEN format [9].

The major advantage of using only OWL 2 DL to represent both conceptual and normative knowledge is that we can use an existing reasoner, Pellet, to do all reasoning. However there is a serious limitation: Description Logics are variable free. That means that it is difficult and sometimes impossible to enforce maintaining the identity of individuals. For rule formalisms this is not a problem. Alternative approaches, such as structured objects ([6]) are very promising as well, but are currently not supported by mainstream reasoners. However, the necessary effect can sometimes be achieved by approximation [4]. Also, others are working to achieve higher expressiveness by using SWRL conditions [3] or SPARQL queries. In [2] these extensions are further worked out.

References


http://www.metalex.eu/