View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Open Repository and Bibliography - Luxembourg

Reified Input/Output logic: Combining Input/Output logic and Reification to represent norms coming from existing legislation

Livio Robaldo and Xin Sun

University of Luxembourg*

{*xin.sun, livio.robaldo*}*@uni.lu*

February 20, 2017

Abstract

In this paper, we propose to combine Input/Output logic, a well-known formalism for normative reasoning, with the reification-based approach of Jerry R. Hobbs. The latter is a wide-coverage logic for Natural Language Semantics able to handle a fairly large set of linguistic phenomena into a simple logical formalism. The result is a new framework that we will call 'reified Input/Output logic'. This paper represents the first step of a long-term research aiming at filling the gap between Input/Output logic and the richness of Natural Language Semantics. We plan in our future work to use reified Input/Output logic as the underlying formalism for applications in legal informatics to process and reason on existing legal texts, which are available in natural language only.

1 Introduction

Legal informatics is experiencing growth in activity, also at the industrial level (cf. (Boella et al., 2016), (Ajani et al., 2017)). Several research/industrial projects aimed at designing or extending platforms, via Natural Language Processing (NLP) techniques, for helping legal professionals to retrieve the information they are interested in have been funded recently by the EU commission and other institutions. Recent projects along this line are `Legivoc`[1] (Vibert,

*Livio Robaldo has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 661007 for the project "ProLeMAS: PROcessing LEgal language in normative Multi-Agent Systems". Xin Sun has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 690974 for the project "MIREL: MIning and REasoning with Legal texts".

[1]`http://www.legivoc.eu`

Jouvelot, and Pin, 2013), `Openlaws`[2] (Winkels, 2015), and `EUCases`[3] (Boella et al., 2015).

Although state-of-the-art solutions, e.g. (Boella, Di Caro, and Robaldo, 2013) and (Boella et al., 2013), help navigate legislation and retrieve information, the overall usefulness of the systems are limited due to their focus on terminological issues and information retrieval while disregarding the specific *semantic* aspects, which allow for legal reasoning. Just as standard deontic logic focused mostly on the notion of obligation, subsequent developments in deontic logic also adopt an abstract view of law, with a very loose connection with the texts of regulations. For lawyers, the meaning of laws can only be really understood in the rich expressiveness of natural language since "like language generally, legal discourse can never escape its own textuality" (Peller, 1985).

There is thus a gap between a powerful reasoning mechanism on the formalization of law and the textuality of law, which can be addressed with solutions coming from the literature on Natural Language Semantics (NLS). For this reason, two ongoing research projects in legal informatics aiming at applying past research on NLS and NLP to the legal domain have been recently approved by the EU: `ProLeMAS`[4] (PROcessing LEgal language in normative Multi-Agent Systems) and `MIREL`[5] (MIning and REasoning with Legal texts).

This paper considers Input/Output logic as a promising formalism for the legal domain. Such a direction has already been advocated in (Boella and van der Torre, 2004), where the application of the framework in legal reasoning is mentioned as possible future work, but this has been never done.

Input/Output logic takes its origin in the study of conditional norms and it has been originally introduced in (Makinson and van der Torre, 2000). As explained in (Makinson and van der Torre, 2000), (Makinson and van der Torre, 2003), and (Gabbay et al., 2013), such a kind of *operational semantics* solves the well-known (Jørgensen, 1937)'s dilemma, which roughly says that a proper truth-conditional logic of norms is impossible in that norms do not carry truth values. (Makinson and van der Torre, 2000) explicitly state: "Only declarative statements may bear truth-values, but norms are items of another kind. They may be respected (or not), and may also be assessed from the standpoint of other norms, for example when a legal norm is judged from a moral point of view (or viceversa). But it makes no sense to describe norms as true or as false.". Thanks to its operational semantics, Input/Output logic is able to properly deal with typical theoretical problems of standard deontic logic, such as contrary-to-duty reasoning (see (Makinson and Van der Torre, 2001)) and moral conflicts (see (Parent, 2011)).

The main limitation of Input/Output logic with respect to our long-term research objectives is that all Input/Output systems studied so far are strictly propositional, i.e. their basic components are whole propositions. A proposition basically refers to a whole sentence. On the other hand, NLS includes a

[2]`http://www.openlaws.eu`
[3]`http://eucases.eu`
[4]`http://www.liviorobaldo.com/ProLeMAS.html`
[5]`http://www.mirelproject.eu`

wide range of fine-grained intra-sentence linguistic phenomena: named entities, anaphora, quantifiers, etc. It is then necessary to move beyond the propositional level, in order to handle the meaning of the phrases constituting the sentences.

Being the first attempt to integrate solutions coming from the literature on NLS in Input/Output logic, this paper focuses on the basic definitions in (Makinson and van der Torre, 2000), in order to address basic kinds of norms. In particular, we will restrict our attention to the basic instances of the well-known distinction between regulative norms (obligations, permissions, and prohibitions[6]) and constitutive norms identified by (Searle, 1995) and (Sartor, 2005), among others. In our future works, we will generalize the formalization below towards more advanced Input/Output systems, in order to properly handle special subtypes of obligations and permissions, e.g., those identified in (Makinson and van der Torre, 2003), (Parent, 2011), and (Governatori et al., 2013).

Furthermore, we aim at keeping the architecture of the formulae as simple as possible, in order to increase readability and to help controlling the overall computational complexity. Since legislation is huge, we believe that simplicity is a necessary feature for a logical formalism designed for applications in legal informatics, in order to foster active collaboration of legal practitioners, usually having little expertise in logic, who can contribute to the building of large knowledge bases of formulae.

To achieve this two-fold goal, this paper proposes to wrap Input/Output logic around the approach of Jerry R. Hobbs, grounded on the notion of reification.

Reification is a concept originally introduced by the philosopher Donald Davidson in (Davidson, 1967). Modern logical approaches based on reification are known in the literature as 'neo-Davidsonian' approaches. Reification allows a wide variety of complex NL statements to be expressed in First Order Logic (FOL). NL statements are formalized such that events, states, etc., correspond to FOL terms (constants, variables, and functional terms). In other words, the states and events denoted by these terms are conceived as *things* in the world.

In line with (Bach, 1981), we use in this paper the term "eventuality" to denote both the reification of a state and the one of an event/action.

Reification allows us to move from standard notations in FOL such as the atomic formula '$(give\ a\ b\ c)$', asserting that 'a' gives 'b' to 'c', to another[7] notation in FOL '$(give'\ e\ a\ b\ c)$', where e is the *reification* of the giving action. The expression '$(give'\ e\ a\ b\ c)$' reads: 'e' is a giving event by 'a' of 'b' to 'c'. 'e' is a FOL term exactly as 'a', 'b', and 'c'. In other words, we introduced a first-order term that explicitly denotes the action.

As it will be extensively explained and exemplified below, the crucial feature of Hobbs's logic, which convinced us to use it for achieving our research goals,

[6]It is standardly assumed that prohibitions are indeed obligations; something is prohibited if and only if there is obligation to its contrary. In this paper, we assume the term "obligation" also encompasses prohibitions.

[7]Consistently with (Hobbs, 1998), we will use two related sets of predications: primed and unprimed. For instance, the 3-ary predication '$(give\ a\ b\ c)$' seen above is paired up with the 4-ary predication '$(give'\ e\ a\ b\ c)$', where 'e' is the reification of the former and refers to the "giving" action between the three individuals.

is that it allows to model complex linguistic phenomena via *flat* representations in FOL. By "flat", we mean that the resulting formulae *do not contain nestings* of subformulae within other operators. In logic, high-order operators, such as modal, temporal, etc. operators, as well as boolean connectives, usually outscope predicates and possibly other operators, thus introducing a *hierarchy* between the predications. In Hobbs's, those operators are modeled via other first-order predicates asserted on eventualities, thus allowing to: (1) handle several linguistic phenomena that can be hardly modeled via standard FOL representations, based on subformulae-embeddings, and (2) build formulae that are mere conjunctions of atomic FOL predicates.

In light of this, in our view Hobbs's logic is a promising choice to achieve our objectives, in that it allows to keep the formulae simple and readable as well as to control computational complexity.

To capture the full richness of legal language in the way it expresses norms, we started by studying a corpus of EU legislation in English on a variety of topics. The corpus includes twenty European Directives from 1998 to 2011, covering a range of subjects, e.g., the profession of lawyer, passenger ships, biotechnological inventions, electronic signatures, seafarers' hours of work, etc.

The following are three norms extracted from the corpus, that we will use below to illustrate our logical framework. We did not find relevant differences between (1.a-c) and the other norms in the corpus, thus we assume our formalization is general enough to cover a representative part of EU legislation.

(1) a. A lawyer who wishes to practise in a Member State other than that in which he obtained his professional qualification shall register with the competent authority in that State.

(From: *Directive 98/5/EC of the EU Parliament and of the Council of 16/2/1998 to facilitate practice of the profession of lawyer on a permanent basis in a Member State other than that in which the qualification was obtained.*)

b. Where baker's honey has been used as an ingredient in a compound foodstuff, the term 'honey' may be used in the product name of the compound food instead of the term 'baker's honey'.

(From: *Council Directive 2001/110/EC of 20/12/2001 relating to honey.*)

c. Member States shall take the necessary measures to allow any airport user wishing to use the tailored services or dedicated terminal or part of a terminal, to have access to these services and terminal or part of a terminal.

(From: *Directive 2009/12/EC of the EU Parliament and of the Council of 11/3/2009 on airport charges.*)

(1.a) is an *obligation* that must be respected by lawyers who wish to practise in EU but outside the State where they obtained their qualification. (1.b) is a *permission*. It states that it is possible/permitted to use the term 'honey' in place of 'baker's honey' even when the latter is used as an ingredient of a compound foodstuff. In case permissions are not respected, they of course do not result in any violation. (1.c) is the trickiest of the three examples in (1) because it contains both a permission and an obligation. Airport users are *allowed* to use the tailored services or dedicated terminal or part of a terminal. On the other hand, Member States are *obliged* to guarantee that all airports allow their users to use them. Indeed, this is a linguistic pattern quite common in law: "someone is obliged to guarantee some permissions to someone else".

The paper is organized as follows. Section 2 briefly discusses some previous similar approaches for modeling NL utterances found in legal documents, particularly the ones making use of reification. The section will show that the semantic representations employed are very reminiscent of standard approaches in NLS, such as Discourse Representation Theory (Kamp and Reyle, 1993) and Minimal Recursion Semantics (Copestake, Flickinger, and Sag, 2005), and so they suffer of the same limitations extensively discussed by both J.R. Hobbs in (Hobbs, 2001), (Montazeri and Hobbs, 2011) and the first author of this paper in (Robaldo, 2010a) and (Robaldo, 2010b), among others.

Section 3 and section 4 introduce the two formal instruments at the basis of our logical formalization: Hobbs's logic and Input/Output logic. As pointed out above, so far the two logical frameworks have been studied in isolation, so that the two sections are completely unrelated.

Section 5 shows how it is possible to wrap Input/Output logic around Hobbs's logic, by extending the basic Input/Output logic definitions to this end, while section 6 will show how the resulting logical framework can be used to model norms taken from existing legislation, such as the ones shown in (1).

Section 7 addresses future works, by focusing on how we plan to use the formalism in a specific research project we will carry out in the next years: the project proposes to formalize norms coming from the forthcoming General Data Protection Regulation (GDPR) and to correlate them with norms found in existing ISO standards. Section 8 concludes the paper.

2 Related works

Few approaches in legal informatics try to model NL sentences coming from *existing legal norms*, such as those in (1). The most representative work is perhaps (Sergot et al., 1986). Examples of real norms formalized, in deontic logic, may be also found in (Governatori et al., 2013).

Some other approaches try to formalize legal knowledge via Event Calculus (Kowalski and Sergot, 1986) (Miller and Shanahan, 1999). Event Calculus is a neo-Davidsonian logical language that extends the original account of reification by Davidson (see (Galton, 2006) for a discussion). In particular, Event Calculus introduces special terms and predicates to deal with time points and time periods, such as the predicate $HoldsAt$, used to assert that a fluent, which is basically an eventuality that can change over time, holds at a certain time.

A recent approach in the line is (Hashmi, Governatori, and Wynn, 2014), where it is argued that Event Calculus predicates for handling time cannot be also used for handling deontic meaning. Therefore, a new version of these predicates is proposed (e.g., $DHoldsAt$, where "D" stands for "deontic") to incorporate the deontic effect of norms, so that they can be used for compliance checking. Similar proposals are (Paschke and Bichler, 2005), (Evans and Eyers, 2008), and (Fornara and Colombetti, 2009). However, (Hashmi, Governatori, and Wynn, 2014) appears to be superior in that it identifies and formalizes much more fine-grained and complex obligation modalities. And, violations of the obligations and compensations of such violations are taken into great account and formalized in a time perspective.

To our knowledge, the approach that appears to be closest to the one we are going to propose below is perhaps McCarty's Language for Legal Discourse (LLD) (McCarty, 2002), (McCarty, 2007). This approach is strongly drawn on previous studies in Natural Language Semantics, it uses reification, and it has been specifically developed to model existing legal text. For this reason, we will dedicate to McCarty's approach the next subsection.

2.1 McCarty's Language for Legal Discourse (LLD)

McCarty's Language for Legal Discourse (LLD) is an intuitionistic first-order language that incorporates insights from previous literature in NLS. It has been originally defined in (McCarty, 1989), and it uses reification for handling actions/states, time, obligations and permissions in a single uniform language.

(McCarty, 2007) presents an extension of DLL and an experiment where well-formed structures in extended DLL are generated from federal civil cases in the appellate courts in USA via Collins' NLP parser (Collins, 2003).

An excerpt of legal text and its formalization, taken from (McCarty, 2007), is shown in (2). The sentence in (2) is represented via the structure below it.

In (2), `sterm`, `nterm`, `aterm`, and `pterm` are reified terms of different kind. `sterm`s are terms referring to reified relations, such as the action denoted by the main verb "contends" (cf. the first line of the formula in (2)). `nterm` refers to objects and actors such as "the petitioner" and "the jury". Finally, `aterm` and `pterm` refer to the reified relations denoted by adverbial/adjectival modifiers and prepositional modifiers respectively.

(2) "The petitioner contends that the regulatory takings claim should not have been decided by the jury"

```
sterm(contends, A,
      [nterm(petitioner, B, [])
       /det(The, nn),
       sterm(decided, C,
             [D,
              aterm(regulatory,E,[F]) &
              nterm(takings,G,[]) &
              nterm(claim,F,[])
              /det(the, nn)])
       && H^pterm(by, H,
                  [C,
                   nterm(jury,I,[])
                   /det(the, nn)])
       /[modal(should),negative,perfect,passive]])
```

We skip the technical details of the representation in (2). The crucial feature of the semantic structure in (2) is its close architectural relation with the syntactic constituency structure of the sentence (Chomsky, 1957). A rough (non-detailed) syntactic constituency structure of the sentence in (2) is shown in (3). NP, AP, PP, and VP stands for "noun phrase", "adjectival phrase", "prepositional phrase", and "verbal phrase" respectively. S marks a sentence-phrase.

(3) [[[The petitioner]$_{NP}$
  [contends that
    [[the [regulatory takings]$_{AP}$ claim]$_{NP}$
      [should not[have been decided [by [the jury]$_{NP}$]$_{PP}$]$_{VP}$]$_{VP}$]$_{S}$]$_{VP}$]$_{S}$

The syntactic structure in (3) and the semantic structure in (2) are closely related both from a lexical point of view and from a syntactic point of view:

(4) a. *Close relation with the lexicon*: The first argument of each `*term` in (2) is a lexical entry of the sentence, e.g. "contends", "petitioner", "decided", etc. `*terms` are also sub-classified with respect to the part-of-speech of the head of the constituents they refer to: `sterm` for sentence and verbal phrases, `nterm` for noun phrases, `aterm` and `pterm` for adverbial/adjectival and prepositional phrases.

b. *Close relation with the syntax*: `*terms` outscope other `*terms` by roughly mirroring the inclusion of the constituents associated with the narrow-scope `*terms` within the constituents associated with the wide-scope `*terms`. For instance, in (2), the main `sterm` outscopes the `nterm` associated with the noun phrase "the petitioner" and the `sterm` associated with the sentence-phrase "the regulatory takings claim should not have been decided by the jury". In the syntactic representation in (3), these two phrases are combined into the bigger sentence-phrase associated with the main `sterm`.

The representation in (2) is very reminiscent of formalisms standardly used in NLS, such as Montague's Universal Grammar (Montague, 1970), Discourse Representation Theory (DRT) (Kamp and Reyle, 1993), as well as underspecified logics such as Minimal Recursion Semantics (MRS) (Copestake, Flickinger, and Sag, 2005). Specifically, LLD is drawn from Quasi Logical Form (QLF) (Alshawi, 1992), which is considered one of the precursors of MRS. The reader is addressed to (Robaldo, 2007) for a full overview and comparison between QLF, MRS, and other similar formalisms.

The close relation between syntax and semantics, and in particular the two architectural choices in (4.a-b), is the main consequence of the application of the well-known Montague's *principle of compositionality*[8], a cornerstone of standard formalisms used in NLS. According to Montague, natural language is itself a logic, where "the meaning of the whole is a function of the meanings of its parts and their mode of syntactic combination".

Nevertheless, it has been shown by (Hobbs, 2008) and (Robaldo, Szymanik, and Meijering, 2014), among others, that a strict observance of the principle of compositionality prevents several readings indeed available in NL. In order to properly represent these readings, these authors propose *flat* semantic formalisms, in the sense that embeddings of terms within the scope of other terms (i.e., (4.b)) are always avoided. In other words, formulae do not establish any hierarchy among the predications occurring therein. The next subsection discusses some of these readings. Afterwards, we will present the flat reified logic by J. R. Hobbs.

2.2 The richness of Natural Language: anaphora, hidden eventualities, scopeless readings

As extensively discussed by both J.R. Hobbs in (Hobbs, 2001), (Montazeri and Hobbs, 2011) and the first author of this paper in (Robaldo, 2010a) and (Robaldo, 2010b), the two architectural choices in (4.a-b) are too rigid to properly handle NLS, in that they tend to mirror the architecture of the syntactic structures in the architecture of the semantic ones. The former are based on (recurring) embeddings of phrases within the scope of other phrases. On the other hand, at the semantic level, more flexibility is needed.

It is quite common in our everyday life to abstract complex actions and situations and treat them as atomic units while building "separate" meanings. A logical formalization based on embeddings imposes a hierarchical order between the predications. As a result, several available readings are intrinsically prevented, and more complex operators, able to connect the predications across the hierarchy, must be introduced in order to properly represent these readings.

For instance, consider the sentences in (5), which are similar to the examples considered by Hobbs and Robaldo in their past research in NLS.

[8]`http://plato.stanford.edu/entries/montague-semantics/#Com`

(5) a. Reimbursement may be obtained, but *it* could take some months.

b. The city does not have a train station, *but* it has a bus station.

c. If the parents of a student earn *less than 20k* euros per year, then the student is eligible.

In sentence (5.a), the pronoun "it" refers to *the obtaining* of the reimbursement. The interpretation of this pronoun differs from the interpretations of the pronouns "he" and "his" in (1.a) or the pronoun "it" in (5.b). The pronoun "it" in (5.a) refers to an eventuality which must be abstracted from the first sentence. On the other hand, the pronouns in (1.a) and (5.b) refer to (possibly quantified) non-eventuality individuals of the domain (lawyers and cities). Other complex cases of anaphoric references to eventualities are discussed in (Hobbs, 1998).

(5.b) is an example of concessive relation, one of the most trickiest semantic relations occurring in NL. The first clause creates the expectation that the city is unreachable by public transportation. The second clause denies that expectation. Note that the expectation is an "hidden" clause, thus it is not associated with any lexical item. For this reason, (5.b) cannot be represented in LLD via its *basic* constructs, due to (4.a). (Robaldo and Miltsakaki, 2014) proposes a solution to model concessive relations in Hobbs's logic, based on an empirical analysis of more than 1000 occurrences of concession taken from the Penn Discourse Treebank (Prasad et al., 2008), (Miltsakaki et al., 2008). Concessive relations are special cases of elliptical sentences. It is easy to understand that a close relation with the lexicon, i.e., (4.a), makes difficult the formal representation of elliptical sentences, in that they are the result of an inference process from the content of the sentence and the background knowledge. Again, we need more flexibility on the syntax-semantics interface, and, in particular, we need to neatly decouple the semantic structures from the syntactic ones, in order to avoid bringing in the former language-dependent characteristics that pertain only to the grammar.

Finally, (5.c) shows that standard approaches to NLS are indeed even unable to properly deal with non-eventuality individuals. (5.c) is an example of cumulative reading, which is a special kind of scopeless reading[9]. In the preferred reading of (5.c), if the money *cumulatively* earned by at least one of the parents, or by both together[10], is less than 20k euros, the student is eligible. Scopeless readings, which are indeed quite widespread in NL (cf. (Robaldo and Di Carlo, 2013)), have been extensively studied in a reification setting in (Robaldo, 2011) and (Robaldo, 2013). And, in (Robaldo, 2010b) it has been argued that the principle of compositionality intrinsically prevents the proper representation of these readings, unless introducing further complex operators such as polyadic quantifiers (cf. (van Benthem, 1989)). The principle of compositionality imposes to embed quantifiers within the scope of other ones, so that, in (5.c),

[9]The term "scopeless reading" comes from the fact that, as explained in the text, in order to properly interpret the sentence, the two quantifiers must be interpreted *in parallel*, i.e. neither of the two must include the other in its scope.

[10]For instance, suppose they rent an apartment they co-own.

only the interpretation where the parents of a student earn less than 20k euros *each* is possible. In (Robaldo, 2011), the quantified sets are reified into special predicates, which are all conjoined at the same level of scope. Then, it is either possible to establish Skolem-like functional dependencies between these sets (thus representing readings where some sets depend on other ones) or to evaluate the sets in parallel (thus representing scopeless readings).

To summarize, Hobbs and Robaldo propose to use reification and flat formulae in order to maximize the convenience of talking about states and events in the world. As a result, a wide set of complex linguistic phenomena, including those exemplified in (5), can be straightforwardly handled via a simple formalism. States and events may be reified at any level of abstraction. Separate predications are asserted on such reified terms and, possibly, reified again into new eventualities. This is the formal mechanism at the basis of Hobbs's logic.

Hobbs's logic will be presented in the next section. In the subsequent one, we will present Input/Output logic. Then, we will propose to use Hobbs's logic as the object logic of those Input/Output systems used for normative reasoning.

3 Hobbs' logical framework

Jerry R. Hobbs defines a wide-coverage first order logic (FOL) for NLS centered on the notion of reification. See (Hobbs, 1998) and several other earlier publications by the same author[11].

Hobbs distinguishes two parallel sets of predicates: primed and unprimed. The unprimed predicates are standard FOL predicates commonly used in logical representations. For example, (*give a b c*) asserts that *a* gives *b* to *c* in the real world. The primed predicate represents the reification of the corresponding unprimed relation. The expression (*give′ e a b c*) says that *e* is a giving event by *a* of *b* to *c*. Alternatively, we may think of *e* as "the fact that" *a* gives *b* to *c* or, in line with the terminology used in (Hobbs, 1998), as the "giving-ness" performed by *a* of *b* to *c*. Formally, *e* is a FOL term exactly as *a*, *b*, and *c*.

Eventualities may be *possible* or *actual*. In Hobbs', this distinction is represented via a unary predicate *Rexist* that holds for eventualities really existing in the world. To give an example cited in Hobbs, if I want to fly, my wanting really exists, but my flying does not. This is represented as[12]:

$$\exists_e \exists_{e1}[\ (\mathit{Rexist}\ e) \wedge (\mathit{want}'\ e\ \mathtt{I}\ e_1) \wedge (\mathit{fly}'\ e_1\ \mathtt{I})\ ]$$

Eventualities can be treated as the objects of human thoughts. Reified eventualities are inserted as parameters of such predicates as *believe*, *think*,

[11]Hobbs implements a fairly large set of linguistic and semantic concepts including sets, composite entities, scales, change, causality, time, event structure, etc., into an integrated first order logical formalism. The reader is addressed to the manuscripts at `http://www.isi.edu/~hobbs/csk.html` and `http://www.isi.edu/~hobbs/csknowledge-references/csknowledge-references.html`.

[12]'`I`' is a FOL constant deictically referring to the speaker. Analogously, in the next example, `John` and `Jack` are FOL constants respectively referring to the two boys.

want, etc. Reification can be applied recursively. The fact that "John believes that Jack wants to eat an ice cream" is represented as an eventuality e such that it holds:

$$\exists_e \exists_{e1} \exists_{e2} \exists_x [\ (Rexist\ e) \wedge (believe'\ e\ \texttt{John}\ e_1) \wedge$$
$$(want'\ e_1\ \texttt{Jack}\ e_2) \wedge (eat'\ e_2\ \texttt{Jack}\ x) \wedge (iceCream'\ e_3\ x)\ ]$$

An unprimed predicate is true if and only if the eventuality of its primed version really exists[13]; for a generic unary predicate p, this is formalized as:

(6) $\forall_e \forall_x [\ (p\ x) \leftrightarrow ((Rexist\ e) \wedge (p\text{'}\ e\ x))\ ]$

As already mentioned above, the crucial feature of Hobbs' logic, which distinguishes it from all other neo-Davidsonian approaches, such as McCarty's LLD or the other logics based on Event Calculus mentioned above, is that all formulae are "flat", in the sense that they are mere conjunctions of atomic predications asserted on FOL terms. There are no embeddings of subformulae within other operators, i.e. no hierarchies of predications.

As (Hobbs, 1998), pp.5, states: "There has been an attempt to make the notation as 'flat' as possible. All knowledge is knowledge of predications. Constants[14] are only handles. The intuition is that in natural language we cannot communicate entities directly. We can only communicate properties and hope that the listener can determine the entity we are attempting to refer to."

3.1 Modelling meaning in Hobbs's logic

Hobbs's logic is a restricted fragment of standard first order logic: in Hobbs's, well-formed formulae are conjunctions of atomic predicates instantiated on FOL terms (costants, functions, and variables). In other words, the only FOL operator used in Hobbs's logic is "$\wedge$", and so the only possible standard FOL inferences are the ones enabled by "$\wedge$", i.e. "$(A \wedge B) \vdash A, B$" and "$A, B \vdash (A \wedge B)$".

On the other hand, standard FOL, including all its connectives ($\wedge$, $\neg$, $\vee$, $\leftarrow$, $\leftrightarrow$), is used as a meta-language to build another semantics, centered on the predicate *Rexist* (cf. below in (7), (8), (9), etc.). As said above, the predicate *Rexist* is used to assert which eventualities really exist in the current world [15].

Negation, conjunction, and disjunction *on eventualities* are defined via three predicates[16] *not*, *and*, and *or*. These must not be confused with standard FOL boolean operators "$\neg$", "$\wedge$", and "$\vee$".

The assertion $(not'\ e\ e_1)$ simply states that the two eventualities e and e_1 are *related* via a particular relation: e is the negation of e_1 (and viceversa) in the sense that when one of the two really exists the other one does not.

[13]Cf. `http://www.isi.edu/~hobbs/bgt-evstruct.text`

[14]And other FOL terms (e.d.). In the formalization below, all eventualities will be quantified variables.

[15]Real existence is only one of the possible modalities of the logic. At `http://www.isi.edu/~hobbs/bgt-modality.text`, it is explained how it is possible to deal with other modalities.

[16]See `http://www.isi.edu/~hobbs/bgt-logic.text`.

The semantics of not' may be then defined by the interpretation rule in (7), expressed in standard FOL.

(7) $\forall_e \forall_{e_1}[\, (not'\ e\ e_1) \rightarrow ((Rexist\ e) \leftrightarrow \neg(Rexist\ e_1))\ ]$

In other words, if $(not'\ e\ e_1)$ is true, all what we know is that the individuals e and e_1 are related via the not' predication. But this does not tell us anything about the real existence of either e or e_1. And, the semantics of not' in (7) simply states that when the real existence (or the real non-existence) of one of the two eventualities is known, then the real existence (or the real non-existence) of the other may be inferred.

Similarly, the assertion $(and'\ e\ e_1\ e_2)$ states that e, e_1, e_2 are related via a particular "conjunctive" relation such that:

(8) $\forall_e \forall_{e_1} \forall_{e_2}[\, (and'\ e\ e_1\ e_2) \rightarrow ((Rexist\ e) \leftrightarrow ((Rexist\ e_1) \wedge (Rexist\ e_2)))\ ]$

And, the assertion $(or'\ e\ e_1\ e_2)$ states that e, e_1, e_2 are related via a particular "disjunctive" relation such that:

(9) $\forall_e \forall_{e_1} \forall_{e_2}[\, (or'\ e\ e_1\ e_2) \rightarrow ((Rexist\ e) \leftrightarrow ((Rexist\ e_1) \vee (Rexist\ e_2)))\ ]$

Finally, the assertion $(imply'\ e\ e_1\ e_2)$ states that e, e_1, e_2 are related via a particular "implicative" relation such that:

(10) $\forall_e \forall_{e_1} \forall_{e_2}[(imply'\ e\ e_1\ e_2) \rightarrow ((Rexist\ e) \leftrightarrow ((Rexist\ e_1) \rightarrow (Rexist\ e_2)))]$

From the definitions of not', and', or', and $imply'$ introduced above, we are ready to prove some theorems. For instance, in Hobbs's logic, it holds the following theorem, which parallels the well-known equivalence in standard FOL "$\forall_x[(A(x) \rightarrow B(x)) \leftrightarrow (\neg A(\mathrm{x}) \vee B(x))]$":

(11) $\forall_{e_i} \forall_{e_o} \forall_{e_1} \forall_{e_1^n} \forall_{e_2}[\, ((imply'\ e_i\ e_1\ e_2) \wedge (not'\ e_1^n\ e_1) \wedge (or'\ e_o\ e_1^n\ e_2)) \rightarrow$
$((Rexist\ e_i) \leftrightarrow (Rexist\ e_o))\ ]$

(11) may be proved in all possible directions via the interpretation rules in (7), (8), (9), and (10). For instance, if $(Rexist\ e_o)$ is true, by (9) also the disjunction $((Rexist\ e_1^n) \vee (Rexist\ e_2))$ is true, i.e. at least one of the two eventualities e_1^n and e_2 really exist. If $(Rexist\ e_1^n)$ is true, then $(Rexist\ e_1)$ is false, by (7). Then, independently of the truth value of $(Rexist\ e_2)$, $(Rexist\ e_i)$ is true by (10). On the other hand, if $(Rexist\ e_1^n)$ is false, then both $(Rexist\ e_1)$ and $(Rexist\ e_2)$ are true, and $(Rexist\ e_i)$ is again true by (10.a) $\Box$.

We stress again that, although the theorem in (11) *parallels* the well-known FOL theorem "$\forall_x[(A(x) \rightarrow B(x)) \leftrightarrow (\neg A(\mathrm{x}) \vee B(x))]$", the two theorems have nothing to do with each other, in that they belong to different logical systems. (11) simply says that if there are four eventualities e_i, e_o, e_1, e_1^n, and e_2 related via those particular predications, then that particular bi-equivalence holds between two of the four eventualities, i.e., e_i and e_o.

Another example is given in (12). The theorem in (12) parallels the well-known De Morgan rule "$\forall_x[\ \neg(A(x) \vee B(x)) \leftrightarrow (\neg A(x) \wedge \neg B(x))]$". We omit the proofs.

(12) $\forall_{e_o^n}\forall_{e_o}\forall_{e_a}\forall_{e_1}\forall_{e_2}\forall_{e_1^n}\forall_{e_2^n}[$

$$((not'\ e_o^n\ e_o) \wedge (or'\ e_o\ e_1\ e_2) \wedge (and'\ e_a\ e_1^n\ e_2^n) \wedge (not'\ e_1^n\ e_1) \wedge (not'\ e_2^n\ e_2)) \rightarrow$$
$$((Rexist\ e_o^n) \leftrightarrow (Rexist\ e_a))\ ]$$

3.2 Adding axiom schemas to the TBox

In our logical framework, we distinguish between formulae belonging to the assertive contextual statements (ABox) from formulae belonging to the terminological declarative statements (TBox), i.e., the definitions, axioms, and constraints on the predicates used in the ABox formulae.

In (7), (8), (9), and (10) we have seen examples of *definitions* in Hobbs's logic: new predicates are introduced and their semantics is defined with respect to a meta-language, which is a fragment of standard FOL in our case. On the other hand, in (11) and (12) we have seen two examples of *theorems* in Hobbs's logic: statements that may be derived from the definitions. Both are declarative statements, so they both belong to the TBox. On the other hand, the ABox will contain the formulae corresponding to the norms in legislation, e.g., the translation in logic of the sentences in (1) (see below).

It is then possible to add further *axiom schemas* to the TBox of the knowledge base. Axiom schemas, like definitions, further restrict/model the meaning of the predicates, and other theorems may be derived from them. For instance, we may add an axiom schema stating that all lawyers are humans:

(13) $\forall_{e_1}\forall_x[((lawyer'\ e_1\ x) \wedge (Rexist\ e_1)) \rightarrow \exists_{e_2}[(human'\ e_2\ x) \wedge (Rexist\ e_2)]$

According to Hobbs's terminology, in (13) e_1 and e_2 have to be thought as the "lawyer-ness" and the "human nature" of the second argument of the predicates *lawyer'* and *human'*. From (13) and (10) it may be of course derived that for every individual x who is a lawyer, i.e., for whom his "lawyer-ness" really exists, also the human nature of x really exists, i.e. x is also a human.

Hobbs and his followers defined axiom schemas to handle several complex phenomena in NLS, among which composite entities, causality, time, defeasibility, event structure, etc., thus allowing the logic to properly represent the meaning of NL sentences such as those exemplified in (5). Of course, in the present paper we will not present the solutions for dealing with each of these phenomena, but we simply address the reader to the relevant literature[17].

A major problem in the legal domain concerns the availability of different legal interpretations of the statements in the law (cf. (MacCormick and Summers, 1991) among others). For instance, consider sentence (1.a) above. To what extent should we think of a lawyer who *wishes* to practise in a Member State different from the one he obtained his professional qualification? Under a strictly *literal* interpretation, a lawyer who simply tells some friends he would

[17] We specifically suggest the reader to refer to the manuscripts at http://www.isi.edu/~hobbs/csk.html and http://www.isi.edu/~hobbs/csknowledge-references/csknowledge-references.html.

like to practice in that Member State already violates the norm, if he is not registered with the competent authority. Of course that is not the intended meaning of the norm. A reasonable interpretation of the norm is that the norm is violated only if the non-registered lawyer performs some formal action, such as defending some clients in court or requesting the commune the permission to open his law office in the city. According to the norm, that action should be blocked or, in case it is too late for that, the lawyer must pay a penalty.

Analogously, in (1.c), how do we define users *wishing* to use the mentioned services? And, what exactly are the mentioned "necessary measures"? In case the users do not experience any access problem, the Member State are not required to do anything, indeed. Thus, a reasonable legal interpretation of (1.c) is that the norm is violated only if users *try to use* the services and they do not succeed (after a reasonable number of attempts).

In real situations there are of course much more borderline cases concerning the proper interpretation of laws in given situations, which is up to the judges in courts (Liebwald, 2013). The handling of legal interpretations in a logical framework has been recently addressed by (Governatori, Rotolo, and Sartor, 2015), which introduce specific defeasible operators in order to allow certain legal interpretations to override others.

Following (Governatori, Rotolo, and Sartor, 2015), in this paper we will use the methodology[18] proposed in Hobbs's to deal with defeasibility, which is drawn from Circumscriptive Logic (McCarthy, 1980).

The idea behind Circumscriptive Logic is simple and we illustrate it with an example. The fact that every bird flies is represented in FOL as:

(14) $\forall_x[bird(x) \rightarrow fly(x)]$

In order to render the rule defeasible, we add another predicate *normalBF* stating that birds fly only if it is "normal" to assume so:

(15) $\forall_x[(bird(x) \wedge normalBF(x)) \rightarrow fly(x)]$

Adding that penguins are non-flying birds, i.e.

(16) $\forall_x[penguin(x) \rightarrow (bird(x) \wedge \neg fly(x))]$

does not entail an inconsistency. It entails that $normalBF(x)$ is false for each penguin x. In this sense, (18) is stronger than (15), i.e. it has "higher priority". Alternatively, we may simply directly assert that penguins are not "normal" with respect to the property of flying:

(17) $\forall_x[penguin(x) \rightarrow \neg normalBF(x)]$

Note that the predicate *normalBF* only refers to the property of flying. In other words, other (different) predicates must be added for all defeasible properties of birds or other classes we want to model.

[18]`http://www.isi.edu/~hobbs/bgt-defeasibility.text`

Note also that predicates *normal∗* are always assumed to be true, unless it is explicitly asserted they are not. For instance, in order to infer that birds fly, *normalBF* must be assumed to be true, expect in the case of penguins, where this assumption would lead to a contradiction. By assuming that a predicate *normal∗* is true, it may be derived that other predicates *normal∗* are false. For instance, let's (recursively) modify (18) and (17) by asserting that penguins do *not* fly only if it is "normal" to assume so:

(18) $\forall_x[(penguin(x) \wedge normalPDF(x)) \rightarrow (bird(x) \wedge \neg fly(x))] \wedge$
$\forall_x[(penguin(x) \wedge normalPDF(x)) \rightarrow \neg\, normalBF(x)]$

In (18), assuming that, for each penguin x, *normalPDF*(x) is true entails that *normalBF*(x) is false. Symmetrically, in (18) assuming that, for each penguin x, *normalBF*(x) is true entails that *normalPDF*(x) is false. Of course, in real-world contexts the first assumption must be adopted, i.e. it must be established that the priority of *normalPDF* is higher than the one of *normalBF*.

The great advantage of Circumscriptive Logic is that it does not require the definition of additional meta-operators to establish priorities among the interpretations, in order to solve conflicts. This feature makes the mechanism directly integrable in Hobbs's logic, whose main goal is to keep the formal machinery as simple as possible, i.e. not beyond standard FOL.

The different legal interpretations of "who wishes to practise ... his professional qualification" in (1.a) are similarly handled. Let us assume that if a lawyer x simply *says* he wishes to practise in a Member State y, then he really wishes to do it (literal interpretation). In Hobbs's logic, this is formalized via the following axiom schema:

(19) $\forall_x\forall_y\forall_{e_1}\forall_{e_2}\forall_{e_3}[\ ((lawyer\ x) \wedge (MS\ y) \wedge (say'\ e_1\ x\ e_2) \wedge$
$(wish'\ e_2\ x\ e_3) \wedge (practice'\ e_3\ x) \wedge (in\ e_3\ y) \wedge (Rexist\ e_1)) \rightarrow$
$(Rexist\ e_2)\ ]$

To make the axiom schema in (19) defeasible, we add a predicate *normalSP* stating that the entailment is valid only if it is "normal" to assume it:

(20) $\forall_x\forall_y\forall_{e_1}\forall_{e_2}\forall_{e_3}\forall_{e_n}\forall_{e_a}[\ ((lawyer\ x) \wedge (MS\ y) \wedge (say'\ e_1\ x\ e_2) \wedge$
$(wish'\ e_2\ x\ e_3) \wedge (practice'\ e_3\ x) \wedge (in\ e_3\ y) \wedge$
$(normalSP'\ e_n\ e_1) \wedge (and'\ e_a\ e_1\ e_n) \wedge (Rexist\ e_a)) \rightarrow$
$(Rexist\ e_2)\ ]$

The real existence of e_1 is no longer sufficient to entail the one of e_2. In order to enable the entailment, the real existence of e_n is also needed, i.e., it must be asserted that, with respect to the entailment, assuming e_1 is "normal". Now, a judge may reasonably decide that it is *not* normal assuming that a lawyer who says he will practice in a Member State entails that he really "wishes" (in the sense of (1.a)) to do so, i.e.:

(21) $\forall_x \forall_y \forall_{e_1} \forall_{e_2} \forall_{e_3}[\ ((\textit{lawyer}\ x) \wedge (\textit{MS}\ y) \wedge (\textit{say'}\ e_1\ x\ e_2) \wedge$
$(\textit{wish'}\ e_2\ x\ e_3) \wedge (\textit{practice'}\ e_3\ x) \wedge (\textit{in}\ e_3\ y) \wedge (\textit{Rexist}\ e_1)) \rightarrow$
$\exists_{e_n^n}[\ (\textit{not'}\ e_n^n\ e_n) \wedge (\textit{Rexist}\ e_n^n) \wedge (\textit{normalSP'}\ e_n\ e_1)\ ]$

From (21), in case a lawyer x simply says he wishes to practice in a Member State y, we infer that e_n does *not* really exist. Thus, it is no longer possible to infer, from (20), whether e_2 really exists or not.

To summarize, in our logical framework axiom schemas (as well as definitions and theorems) are FOL formulae intended to populate the TBox: their role is the one of restricting the meaning of the predicates in the ABox. Since the formulae in the ABox will be mere conjunctions of FOL predicates, it is clear that the complexity of the formalism is fully moved to the TBox, in that the complexity of the ABox is trivial. In our view, this feature deems pivotal from a computational point of view, in that, in practical applications, the size of the TBox is usually much lower than the size of the ABox. Thus, the complexity of the logic should be easier to control. In case axiom schemas are allowed to be implications in standard FOL, the logic possibly turns out to be semi-decidable because, as it is well-known, standard FOL is semi-decidable. However, we may require the axiom schemas to be asserted within a *fragment* of standard FOL, thus obtaining decidable and/or tractable formalisms.

In the next section, we will present Input/Output logic, the logic for normative reasoning that we chose to wrap around Hobbs's logic in order to properly represent the meaning of norms expressed in natural language. Afterwards, we will show how the integration of the two logics is possible.

4 Input/Output logic

Input/Output logic takes its origin in the study of conditional norms and it has been originally introduced in (Makinson and van der Torre, 2000). Input/Output logic is not a single logic but a family of logics, just like modal logic is a family of logics containing systems `K`, `KD`, `S4`, `S5`, etc. However, unlike modal logic, which usually uses possible world semantics, Input/Output logic mainly adopts *operational* semantics.

An Input/Output system is conceived as a deductive machine, like a "black box" which produces statements as output, when we feed it factual statements as input. Figure 1 is a brief visualization of a generic Input/Output system.

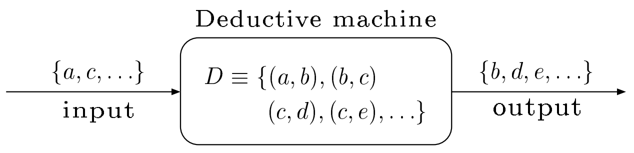

Figure 1: general Input/Output logic system

The set D represents a set of deductive rules. Formally, it is a set of pairs $(x,\ y)$, where x and y are formulae in some *object logic*. Each pair $(x,\ y)$ in D is called a "generator" and refers to a rule. The set D corresponds to the deduction machine of the Input/Output system: whenever one of the formulae in the left-hand sides of the pairs is given in input, the corresponding right-hand sides are given in output.

In order to obtain a family of Input/Output logics, we add axioms to the deductive system exemplified in Figure 1, thus constraining the set of pairs belonging to D (see examples below). But the pairs in D are never evaluated with respect to a model, i.e. associated with a truth value. The deductive machine simply matches the input with the left-hand sides of the pairs and returns the corresponding right-hand sides. In that sense the semantics is *operational*.

Input/Output logic is a *general* framework that, like modal logic, may be used to model a great variety of different meanings and forms of reasoning. For instance, (Bochman, 2003) and (Bochman, 2004) use it to model *causal* reasoning. In our research, we aim at using it for modeling legal reasoning, as an underlying formalism for applications in legal informatics.

As pointed out in the Introduction, we start from the basic definitions in (Makinson and van der Torre, 2000) and from the standard distinction between regulative norms (obligations and permissions) and constitutive norms, identified in (Searle, 1995) and (Sartor, 2005), among others.

In Input/Output logic, a solution for distinguishing between regulative and constitutive norms has been firstly proposed in (Boella and van der Torre, 2004) and then formalized in (Sun and van der Torre, 2014) in terms of *two* sequential "black boxes" (see Figure 2):

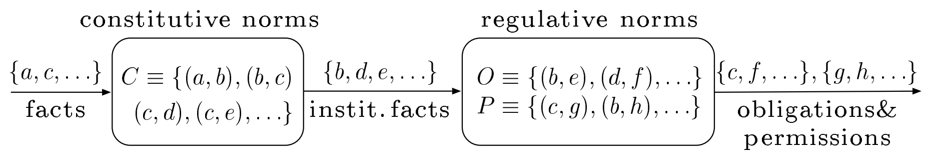

Figure 2: Input/Output logic system for the legal domain

The set C is the set of constitutive norms. It takes as input the facts of the domain and returns the *institutional* facts, i.e., it defines when something *counts as* something else in the domain (cf. (Searle, 1995)). The output of the first "black box" is then given in input to the second "black box", which implements regulative norms.

In reified Input/Output logic, the first "black box" implements the axiom schemas of the underlying object logic discussed in subsection 3.2 above, i.e., the TBox of the reference ontology. On the other hand, the set O and the set P are respectively the set of obligations and the set of permissions of the normative system. A generator $(x,\ y)$ in O reads as "given x, y is obligatory", while a generator $(z,\ w)$ in P reads as "given z, w is permitted".

(Makinson and Van der Torre, 2001) defines additional meta-structure to perform contrary-to-duty reasoning, i.e., to determine which obligations are operative in a situation that already violates some of them. On the other hand, (Parent, 2011) enrich the Input/Output generators with priorities that allow to handle moral conflicts. Finally, (Makinson and van der Torre, 2003) integrates in Input/Output logic the standard distinction between "negative permissions" and "positive permissions" originally identified by (von Wright, 1959) and further addressed in (Alchourrón and Bulygin, 1984), among others.

A proposition is negatively permitted if and only if it is not prohibited by the norms, i.e. if and only if its negation is not obligatory. On the other hand, a proposition is positively permitted by a normative code if and only if it can be derived from the code. Explicitly stated permissions, such as those asserted in P, are positive permissions in that they trivially entail themselves. Other positive permissions may be derived from P and O. However, since in logic there are basically two ways of inferring new propositions, i.e. deductively or by contradiction, and since these two ways could have different outcomes, (Makinson and van der Torre, 2003) further sub-classify positive permissions into static positive permissions and dynamic positive permissions, depending on how they are derived from P and O. Finally, they introduce additional meta-structures in the Input/Output system in order to infer the set of negative, static positive, and dynamic positive permissions from P and O.

In this paper we are not interested in theoretical issues such as contrary-to-duty reasoning and moral conflicts, nor in the distinction between negative permissions and static/dynamic positive permissions. The aim of this paper is the one of translating natural language norms in reified Input/Output logic, in order to create a version of Input/Output logic that may be used in practical (computational) applications for processing and reasoning on legal texts.

Therefore, below in this section we only give some details about the axioms needed to obtain the basic Input/Output systems introduced in (Makinson and van der Torre, 2000). In (Makinson and van der Torre, 2000), these axioms have been asserted in terms of standard propositional logic. In the next sections, they will be generalized in our logical framework.

Let L be the object logic. The set of obligatory norms O can be viewed as a function from 2^L to 2^L such that for a set A of formulas, $O(A) = \{x \in L : (a, x) \in O \text{ for some } a \in A\}$. Depending on the pairs included in O, a different output is produced against an input A. (Makinson and van der Torre, 2000) introduce four basic relevant outputs for an input A, called out_1, out_2, out_3, and out_4. These are functions taking O and A as arguments defined as follows:

(22)
- $out_1(O, A) = Cn(O(Cn(A)))$
- $out_2(O, A) = \bigcap\{Cn(O(V)) : A \subseteq V, V \text{ is complete}\}$
- $out_3(O, A) = \bigcap\{Cn(O(B)) : A \subseteq B = Cn(B) \supseteq O(B)\}$
- $out_4(O, A) = \bigcap\{Cn(O(V) : A \subseteq V \supseteq O(V)), V \text{ is complete}\}$

Here Cn is the classical consequence operator: $Cn(A) = \{a \in L : A \vdash a\}$. A set of formulas V is complete if it is either maximally consistent or equal to

L. These four sets are called 'simple-minded output', 'basic output', 'simple-minded reusable output' and 'basic reusable output' respectively.

For each of these sets, a throughput version that allows inputs to reappear as outputs is defined as $out_i^+(O, A) = out_i(O_{id}, A)$, where $O_{id} = O \cup \{(a, a) : a \in L\}$. When A is a singleton, we write $out_i(O, a)$ for $out_i(N, \{a\})$. Thus, we obtain eight basic Input/Output systems in total.

Input/Output logics are given a proof theoretic characterization. We say that an ordered pair of formulae is derivable from a set O iff (a, x) is in the least set that extends $O \cup \{(\top, \top)\}$ and is closed under a number of derivation rules (axioms). The following are the derivation rules we need for out_1 to out_4^+:

(23)
- `SI` (strengthening the input): from (a, x) to (b, x), whenever $a \in Cn(\{b\})$
- `WO` (weakening the output): from (a, x) to (a, y), whenever $y \in Cn(\{x\})$
- `AND` (conjunction of output): from (a, x) and (a, y) to $(a, x \wedge y)$
- `OR` (disjunction of input): from (a, x) and (b, x) to $(a \vee b, x)$
- `CT` (cumulative transitivity): from (a, x) and $(a \wedge x, y)$ to (a, y)
- `ID` (identity): from nothing to (a, a)

The derivation system based on the rules `SI`, `WO` and `AND` is called $deriv_1$. Adding `OR` to $deriv_1$ gives $deriv_2$. Adding `CT` to $deriv_1$ gives $deriv_3$. The five rules together give $deriv_4$. Adding `ID` to $deriv_i$ gives $deriv_i^+$ for $i \in \{1, 2, 3, 4\}$.

In (Makinson and van der Torre, 2000), it is proved that each $deriv_i$ is sound and complete with respect to $out_i^{(+)}$.

Note that the semantics in (22) and the derivations rules in (23) concern the set of obligations O only. On the other hand, the set P is a mere unconstrained list of pairs, i.e., P is not required to satisfy any axiom. P is only used to determine the set of negative, static positive, and dynamic positive permissions (cf. (Makinson and van der Torre, 2003)).

Examples of how the axioms in (23) work in practice are provided below directly on our first-order object logic. We will show that Hobbs's logic, thanks to its formal simplicity, allows us to enhance the expressivity of Input/Output systems to the first-order level via small modifications of its definitions.

On the other hand, it is important to understand that the axioms in (23) are *not* always suitable for every Input/Output system. It depends on what we want to model. For instance, the axioms `ID` and `OR` are suitable to model the Input/Output system in Figure 2 implementing constitutive rules. Whenever two facts a and b hold in the input, they also hold as institutional facts, as well as their disjunction. On the contrary, `ID` and `OR` do not appear to be suitable for handling legal reasoning, i.e., for modeling the Input/Output system in Figure 2 that implement regulative rules. It is easy to see that by imposing `ID` on that "black box", we would obtain that for every institutional fact a, a is obligatory in the context, which is clearly not the case. To see why also `OR` seems to be problematic, consider the following obligations: "If someone kills a dog, s/he has to spend two years in prison" and "If someone robs a bank s/he has to spend two years in prison". And, suppose we know that John did one of the two, but there is no way to understand *which* one, i.e. if either he killed a dog or he robbed a bank. Logically, John must spend two years in prison. But on the perspective of legal reasoning, he must not: only if concrete evidence of what he did is found, obligations apply.

On the other hand, (Parent and van der Torre, 2014) observed that although axioms `CT` and `WO` are indeed able to handle most inferences allowed in normative reasoning, some counter-examples where they lead to incorrect derivations exist. In light of this, they propose to respectively substitute them with the following (stricter) axioms, `ACT` and `OEQ`:

(24)
- `ACT`: from (a, x) and $(a \wedge x, y)$ to $(a, x \wedge y)$
- `OEQ`: from (a, x) to (a, y), whenever $y \in Cn(\{x\})$ and $x \in Cn(\{y\})$

Analogously to what it has been said above about axioms schemas, i.e. that the present paper is presenting reified Input/Output logic *in general*, while it leaves the definition of the axioms schemas needed to properly model legislation to further studies, below we will refer to the derivation rules in (23) only, without addressing whether they are truly appropriate to model norms or not. The reader is addressed to the relevant literature in Input/Output logic and normative reasoning for further detailed studies.

5 Combining Input/Output logic and Hobbs's to represent obligations and permissions

The present section shows how it is possible to combine Input/Output logic and Hobbs's logic, i.e. to use the latter as the object logic of the former. In line with the spirit and the insights at the basis of Hobbs's formalization, we aim at keeping the formalization as simple as possible.

Hobbs's formulae in the ABox are conjunctions of atomic FOL predicates instantiated on FOL terms. These are used to make assertive contextual statements corresponding to the norms found in legislation, in order to populate the set O and P of the Input/Output system implementing the regulative norms. On the other hand, formulae in the TBox are FOL implications in prenex normal form, which define the predicates used in the ABox; they are intended to be used in the generators populating the set C of the Input/Output system implementing the constitutive norms.

For the Input/Output generators in O and P (ABox), the left-hand side (LHS) and the right-hand side (RHS) are each required to assert exactly one *Rexist* predicate on the main eventuality. The reason is that we want the formulae in the ABox to be conjunctions of atomic predicates, and to clearly identify when an obligation and a permission *really exists* whenever certain facts *really exists*. On the other hand, the Input/Output generators in C (TBox) are not required to be flat formulae, i.e. conjunctions of atomic predicates.

Therefore, they may involve non-reified standard FOL predicates that do not involve any *Rexist* predicate. However, we can of course convert them into their reified counterparts via the axiom (6), shown above.

The formulae in the object logic will involve FOL variables. Some of those variables will occur both in the left-hand side (LHS) and the right-hand side (RHS) of an Input/Output generator, while the others will occur either in the LHS or in the RHS. The variables occurring in both will be universally quantified, while the ones occurring in either one of the two will be existentially quantified. Universal quantifiers will outscope the Input/Ouput generators, thus allowing to "iterate" the generators over all individuals in the domain. In other words, they act as "bridges" between the LHS and the RHS, in order to "carry" single individuals from the input to the output.

On the other hand, existential quantifiers will only outscope the LHSs or the RHSs where their corresponding variables occur.

Therefore, the single formal construct introduced in the original Input/Output definitions are universal quantifiers outscoping the generators. The LHSs and RHSs of the generators are formulae in first-order logic *within* the generators. In other words, the (propositional) (Makinson and van der Torre, 2000)'s generators in the form:

$$(LHS,\, RHS)$$

where LHS and RHS are formulae in standard propositional logic, are generalized into (first-order) generators in the form:

$$\forall_{x_1}\forall_{x_2}\dots\forall_{x_n}(LHS(x_1, x_2, \dots, x_n),\, RHS(x_1, x_2, \dots, x_n))$$

where $LHS(x_1, x_2, \dots, x_n)$ and $RHS(x_1, x_2, \dots, x_n)$ are FOL formulae. The variables $x_1, x_2, \dots, x_n$ are free in LHS and RHS but they are externally bound by universal quantifiers. LHS and RHS possibly include other existentially quantified variables. But the existential quantifiers binding them are outscoped by the generators, and so by the universal quantifiers.

This architectural choice is motivated by an empirical analysis of the obligations and permissions in our corpus of European Directives. Obligations and permissions found in legislation are typically universal assertions that hold for all members in a certain set of individuals, e.g., the set of all lawyers in the domain. For instance, (1.a) states that for each lawyer x having some characteristics, x is obliged to take some actions. But x does not refer to any specific lawyer. In case the model includes several lawyers having the required characteristics, each of them is obliged to take the actions. As said above, the universal quantifier on x acts as a "bridge" to "carry" to the output all lawyers who match the input.

On the other hand, we did not find in our corpus any obligation or permission in the form "If a lawyer exists, then he is obliged to take some actions". This sounds quite intuitive: as said above, statements in legislation are typically universal assertions, i.e., they do *not* hold for single specific individuals.

Note that, in any case, as long as quantifiers have always wide scope with respect to the other FOL items, existentials can be easily removed via skolemization. A generator in the form $\exists_x(LHS(x),\, RHS(x))$ can be substituted by

($LHS(i)$, $RHS(i)$), where i is a FOL constant skolemizing $\exists_x$. On the other hand, a generator in the form $\forall_x(\exists_y LHS(x,\, y),\, RHS(x))$ can be substituted by $\forall_x(LHS(x,\, f(x)),\, RHS(x))$, where f is a FOL function skolemizing $\exists_x$.

Similarly, it must be observed that, in finite domains, universal quantifiers are just a compact way to refer to all individuals in the universe. We obtain an equivalent set of generators by substituting the universally quantified variables with all constants referring each to an individual in the universe. For instance, assuming the universe includes the individuals a, b, c only, the generator $\forall_x(LHS(x),\, RHS(x))$ is equivalent to the set of generators ($LHS(a)$, $RHS(a)$), ($LHS(b)$, $RHS(b)$), and ($LHS(c)$, $RHS(c)$).

In light of this, it should be clear that, provided the universe is finite, our first-order generators can be easily reduced to sets of *propositional* generators, by removing existential and universal quantifiers via skolemization and enumeration respectively.

In the next two subsections, we will formally define our object logic and the generalized version of the axioms in (23). Afterwards, we will show a possible formalization of the examples in (1) in the resulting formalism.

5.1 The object logic

Definition 1 defines the syntax of our object logic. Universal quantifiers are not allowed in the object logic, but only existential ones. Variables that will be bound by (external) universal quantifiers are free in the object logic.

Since, as said above, we distinguish between the regulative norms (sets O and P, i.e., the ABox) and constitutive norms (set C, i.e., the TBox), implemented via two different Input/Ouput systems, we define two different object logics: one for the formulae in the ABox and one for the formulae in the TBox. The former are conjunctions of FOL predicates while the latter are FOL implications in prenex normal form. As said above, well-formed formulae of the object logic for the sets O and P are required to include each exactly one *Rexist* predicate, while well-formed formulae of the object logic for the set C are not.

Definition 1. Syntax of the object logic for generators in O and P
The object logic is a fragment of First Order Logic (FOL) defined as follows:

- The vocabulary includes FOL terms (constants, variables, and functions), FOL predicates, the connective $\wedge$ and the quantifier $\exists$.

- If p is an n-ary FOL predicate different from *Rexist*, the term e is an eventuality, and $a_1, \dots, a_n$ are FOL terms, then $(p\ a_1 \dots a_n)$ and $(p'\ e\ a_1 \dots a_n)$ are atomic predicates. $(p'\ e\ a_1 \dots a_n)$ is the reified version of $(p\ a_1 \dots a_n)$, i.e. e is the eventuality that reifies $(p\ a_1 \dots a_n)$.

- If $\Phi_1 \dots \Phi_n$ are atomic predicates as previously defined, the FOL formula:

$$\exists_e \exists_{x_1} \dots \exists_{x_m} [(Rexist\ e) \wedge \Phi_1 \wedge \dots \wedge \Phi_n]$$

  is a well-formed formula. The set $x_1 \dots x_m$ may be empty. $\Phi_1, \dots, \Phi_n$ possibly include free variables which are bound by external universal quantifiers.

The definition of the object logic for the generators in C is simpler, in that formulae are simply required to be in prenex normal form.

Definition 2. Syntax of the object logic for generators in C
The object logic is a fragment of First Order Logic (FOL) defined as follows:

- The vocabulary includes FOL terms (constants, variables, and functions), FOL predicates, the connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, and the quantifier $\exists$.

- If p is an n-ary FOL predicate, the term e is an eventuality, and $a_1, \dots, a_n$ are FOL terms, then $(p\ a_1 \dots a_n)$ and $(p'\ e\ a_1 \dots a_n)$ are atomic predicates. $(p'\ e\ a_1 \dots a_n)$ is the reified version of $(p\ a_1 \dots a_n)$, i.e. e is the eventuality that reifies $(p\ a_1 \dots a_n)$.

- If Φ_1 and Φ_2 are FOL formulae composed by predicates as previously defined and the connectives $\neg$, $\wedge$, $\vee$, $\rightarrow$, and $\leftrightarrow$, the FOL formula:

$$\exists_{y_1} \dots \exists_{y_m} [\Phi_1 \rightarrow \Phi_2]$$

  is a well-formed formula. The set $y_1 \dots y_m$ may be empty. Φ_1 and Φ_2 possibly include free variables which are bound by external universal quantifiers.

Note that for the LHSs and RHSs of a generator in O and P, the predicate *Rexist* must be *unique* within a well-formed formula, i.e., a well-formed formula contains only one occurrence of the *Rexist* predicate. The predicate *and* defined above in (8) is used in case multiple eventualities really exist in the current world. For instance, the sentence "John is an happy lawyer" is represented by the following well-formed formula:

$$\exists_e \exists_{e_1} \exists_{e_1} [(Rexist\ e) \wedge (and'\ e\ e_1\ e_2) \wedge (lawyer'\ e_1\ \texttt{John}) \wedge (happy'\ e_2\ \texttt{John})]$$

According to the semantics of *and* specified in (8), e really exists if and only if both e_1 and e_2 really exist, i.e. if and only if both the "lawyer-ness" and the "happy-ness" of `John` really exist.

A final clarification concerns the input formulae. In (Makinson and van der Torre, 2000), both the formulae in the generators and the ones in the input are asserted in propositional logic. On the contrary, in our framework the input formulae may contain existentially quantified variables.

Therefore, while in (Makinson and van der Torre, 2000) we must look for generators whose LHS is *equal* to some input formulae, in our framework we must, more generally, look for input formulae that *match* the LHSs of the generators. In case the input formulae contains constants, those need to be *unified* with the quantified variables. For instance, the generator $\forall_x(LHS(x),\ RHS(x))$ matches the input formula LHS(a), where a is a FOL constant. The corresponding output formula is of course $RHS(a)$. Similarly, in case the input formulae contain existentially quantified variables, those need to be unified with the quantified variables in the generators and the existential quantifiers are "carried" to the output. For instance, $\forall_x(LHS(x),\ RHS(x))$ also matches the input formula $\exists_y[LHS(y)]$ and the corresponding output formula is $\exists_y[RHS(y)]$. Since these notions are quite intuitive, we omit formal details.

5.2 Generalizing Input/Output logic axioms

We have shown above how it is possible to model complex NL statements found in existing legislation by combining Input/Ouput logic and reification. Also the axioms in (23) need to be generalized in order to make them working with reified formulae. This section only shows two examples, i.e. the generalization in this sense of the axioms `CT` and `OR`. The generalizations of the other axioms in (23) are similar so that we omit them.

The axiom `CT` (cumulative transitivity), copied again in (25) for reader's convenience, is generalized as in (26). In (26), Ψ'_1, Ψ'_2, and Ψ'_3 are conjunctions of FOL predicates, different from *Rexist*, where $x_1 \dots x_n$ occur free while all other variables are bound by existential quantifiers.

(25) `CT`: from (a, x) and $(a \wedge x, y)$ to (a, y)

(26)

$$\text{from: } \forall_{x_1}\ldots\forall_{x_n}(\exists_{e_{11}}\exists_{y_{11}}\ldots\exists_{y_{1i}}[(Rexist\ e_{11}) \wedge (\Psi'_1\ e_{11}\ y_{11}\ \ldots\ y_{1i}\ x_1\ \ldots\ x_n)],$$
$$\exists_{e_{21}}\exists_{y_{21}}\ldots\exists_{y_{2j}}[(Rexist\ e_{21}) \wedge (\Psi'_2\ e_{21}\ y_{21}\ \ldots\ y_{2j}\ x_1\ \ldots\ x_n)]\ )$$

$$\text{and: } \forall_{x_1}\ldots\forall_{x_n}(\exists_{e}\exists_{e_{11}}\exists_{y_{11}}\ldots\exists_{y_{1i}}\exists_{e_{21}}\exists_{y_{21}}\ldots\exists_{y_{2j}}[(Rexist\ e) \wedge (and'\ e\ e_{11}\ e_{21}) \wedge$$
$$(\Psi'_1\ e_{11}\ y_{11}\ \ldots\ y_{1i}\ x_1\ \ldots\ x_n) \wedge (\Psi'_2\ e_{21}\ y_{21}\ \ldots\ y_{2j}\ x_1\ \ldots\ x_n)],$$
$$\exists_{e_{31}}\exists_{y_{31}}\ldots\exists_{y_{3k}}[(Rexist\ e_{31}) \wedge (\Psi'_3\ e_{31}\ y_{31}\ \ldots\ y_{3k}\ x_1\ \ldots\ x_n)]\ )$$

$$\text{to: } \forall_{x_1}\ldots\forall_{x_n}(\exists_{e_{11}}\exists_{y_{11}}\ldots\exists_{y_{1i}}[(Rexist\ e_{11}) \wedge (\Psi'_1\ e_{11}\ y_{11}\ \ldots\ y_{1i}\ x_1\ \ldots\ x_n)],$$
$$\exists_{e_{31}}\exists_{y_{31}}\ldots\exists_{y_{3k}}[(Rexist\ e_{31}) \wedge (\Psi'_3\ e_{31}\ y_{31}\ \ldots\ y_{3k}\ x_1\ \ldots\ x_n)]\ )$$

Note that, since $x_1, \ldots, x_n$ is the set of free variables occurring in both the LHS and the RHS of the generators, the three generators in (26) must have the same number of free variables. An example of (26)'s instantiation is: given "Every lawyer is obliged to run" and "Every lawyer who runs is obliged to wear a red hat", formalized in (27) and (28) respectively:

(27) $\forall_x(\exists_{e_{11}}[(\mathit{Rexist}\ e_{11}) \wedge (\mathit{lawyer}'\ e_{11}\ x)], \exists_{e_{21}}[(\mathit{Rexist}\ e_{21}) \wedge (\mathit{run}'\ e_{21}\ x)])$

(28) $\forall_x(\exists_e\exists_{e_{11}}\exists_{e_{21}}[(\mathit{Rexist}\ e) \wedge (\mathit{and}'\ e\ e_{11}\ e_{21}) \wedge (\mathit{lawyer}'\ e_{11}\ x) \wedge (\mathit{run}'\ e_{21}\ x)],$
$\exists_{e_{31}}\exists_z[(\mathit{Rexist}\ e_{31}) \wedge (\mathit{wear}'\ e_{31}\ x\ z) \wedge (\mathit{redHat}\ z)])$

in case the Input/Output system includes the generalized `CT` axiom in (26), the set O must include (29): "Every lawyer is obliged to wear a red hat".

(29) $$\forall_x(\exists_{e_{11}}[(\mathit{Rexist}\ e_{11}) \wedge (\mathit{lawyer}'\ e_{11}\ x) \wedge (\mathit{run}'\ e_{21}\ x)], \\ \exists_{e_{31}}\exists_z[(\mathit{Rexist}\ e_{31}) \wedge (\mathit{wear}'\ e_{31}\ x\ z) \wedge (\mathit{redHat}\ z)])$$

On the other hand, the axiom `OR` (disjunction of input), copied in (30) for reader's convenience, is generalized as in (31).

(30) `OR`: from (a, x) and (b, x) to $(a \vee b, x)$

is generalized as follows:

(31)

from: $\forall_{x_1}\ldots\forall_{x_n}(\exists_{e_{11}}\exists_{y_{11}}\ldots\exists_{y_{1i}}[(\mathit{Rexist}\ e_{11}) \wedge (\Psi'_1\ e_{11}\ y_{11}\ \ldots\ y_{1i}\ x_1\ \ldots\ x_n)],$
$\exists_{e_{21}}\exists_{y_{21}}\ldots\exists_{y_{2j}}[(\mathit{Rexist}\ e_{21}) \wedge (\Psi'_2\ e_{21}\ y_{21}\ \ldots\ y_{2j}\ x_1\ \ldots\ x_n)]\ )$

and: $\forall_{x_1}\ldots\forall_{x_n}(\exists_{e_{31}}\exists_{y_{31}}\ldots\exists_{y_{3k}}[(\mathit{Rexist}\ e_{31}) \wedge (\Psi'_3\ e_{31}\ y_{31}\ \ldots\ y_{3k}\ x_1\ \ldots\ x_n)],$
$\exists_{e_{21}}\exists_{y_{21}}\ldots\exists_{y_{2j}}[(\mathit{Rexist}\ e_{21}) \wedge (\Psi'_2\ e_{21}\ y_{21}\ \ldots\ y_{2j}\ x_1\ \ldots\ x_n)]\ )$

to: $\forall_{x_1}\ldots\forall_{x_n}(\exists_e\exists_{e_{11}}\exists_{y_{11}}\ldots\exists_{y_{1i}}\exists_{e_{31}}\exists_{y_{31}}\ldots\exists_{y_{3k}}[(\mathit{Rexist}\ e) \wedge (\mathit{or}'\ e\ e_{11}\ e_{31}) \wedge$
$(\Psi'_1\ e_{11}\ y_{11}\ \ldots\ y_{1i}\ x_1\ \ldots\ x_n) \wedge (\Psi'_3\ e_{31}\ y_{31}\ \ldots\ y_{3k}\ x_1\ \ldots\ x_n)],$
$\exists_{e_{21}}\exists_{y_{21}}\ldots\exists_{y_{2j}}[(\mathit{Rexist}\ e_{21}) \wedge (\Psi'_2\ e_{21}\ y_{21}\ \ldots\ y_{2j}\ x_1\ \ldots\ x_n)]\ )$

An example of (31)'s instantiation is: given "Every lawyer is obliged to run" and "Every judge is obliged to run", formalized in (32) and (33):

(32) $\forall_x(\exists_{e_{11}}[(\mathit{Rexist}\ e_{11}) \wedge (\mathit{lawyer}'\ e_{11}\ x)], \exists_{e_{21}}[(\mathit{Rexist}\ e_{21}) \wedge (\mathit{run}'\ e_{21}\ x)])$

(33) $\forall_x(\exists_{e_{31}}[(\mathit{Rexist}\ e_{31}) \wedge (\mathit{judge}'\ e_{31}\ x)], \exists_{e_{21}}[(\mathit{Rexist}\ e_{21}) \wedge (\mathit{run}'\ e_{21}\ x)])$

in case the Input/Output system includes the generalized `OR` axiom in (31), the set O must include (34): "Every lawyer or judge is obliged to run".

(34)

$$\forall_{x_1..x_n}(\exists_e\exists_{e_{11}}\exists_{e_{31}}[(Rexist\ e)\wedge(or'\,e\ e_{11}\ e_{31})\wedge(lawyer'\,e_{11}\ x)\wedge(judge'\,e_{31}\ x)],$$
$$\exists_{e_{21}}[(Rexist\ e_{21})\ \wedge\ (run'\ e_{21}\ x)])$$

According to the semantics of the predicate *or* reported above in (9), e really exists if and only if either e_{11} or e_{31} (or both) really exist, i.e. if and only if either the "lawyer-ness" or the "judge-ness" of x really exists.

6 Formalizing natural language norms in reified Input/Output logic

We have now all the ingredients needed to build the formulae representing the norms in (1.a-c), copied again in (35) for reader's convenience. As said above, we did not find relevant differences between these norms and the other norms in our corpus, thus we assume our formalization is general enough to cover a representative part of EU legislation.

(35) a. A lawyer who wishes to practise in a Member State other than that in which he obtained his professional qualification shall register with the competent authority in that State.

b. Where baker's honey has been used as an ingredient in a compound foodstuff, the term 'honey' may be used in the product name of the compound food instead of the term 'baker's honey'.

c. Member States shall take the necessary measures to allow any airport user wishing to use the tailored services or dedicated terminal or part of a terminal, to have access to these services and terminal or part of a terminal.

The basic components of every logical framework are terms and predications, the former referring to individuals in the universe, the latter referring to the truth values of assertions made on these individuals. In standard FOL, terms are constants, variables and functions, while predications are predicates, boolean connectives, and the quantifiers $\forall$ and $\exists$. Further predications (e.g., modal, deontic, temporal, etc., operators) may be added to handle more complex meanings, but, in this way, we are no longer in standard FOL.

In Hobbs's, every predication can be reified in a predicate including at least one eventuality among its arguments, thus allowing to handle complex meanings in standard FOL. While the previous sections should have already made clear how predications are dealt with in Hobbs's, some more clarifications are needed for the identification of terms in the sentences in (35).

As it is well-known in NLS, specifically in the study of generics (Carlson, 1982), "a lawyer" and "a Member State" in (35.a) and "a compound foodstuff" in (35.b) do not refer to a *specific single* lawyer, a *specific single* Member State, and a *specific single* compound foodstuff. Rather, they refer to *every* lawyer, *every* Member State, and *every* compound foodstuff. Therefore, each of the three noun phrases must be associated with a universally quantified variable that respectively ranges over the set of lawyers, the set of Member States, and the set of compound foodstuffs.

More subtle considerations ought to be done for the expressions "that in which he obtained his professional qualification" and "the competent authority in that State" in (35.a). These expressions indeed refer to *functional terms*.

Given a lawyer x, it can be functionally determined the Member State where he obtained his professional qualification. For instance, the lawyer "John" obtained his professional qualification in France, the lawyer "Jack" in Greece, etc. Same considerations hold for "the competent authority in that State": for Italy it could be the Ministry of Justice, for Germany it could be the Court of Cassation, etc. At the level of the logical formulae, the first expression could be then represented via a function $f(x)$, that takes in input a lawyer and returns a Member State. At the implementation level, f may correspond to a query on some databases that store information about lawyers practising in Europe. The expression "the competent authority in that State" is analogously represented.

Other problems concern mass terms. A mass term is an uncountable term such that any quantity of it is treated as an undifferentiated unit. For instance, "baker's honey" in (35.b) is a mass term in that sub-quantities of baker's honey[19] are again baker's honey. A standard way to deal with mass terms in logic is to introduce a predicate *part-of* to relate (sub-)quantities of mass terms. Such a solution has been developed in (Bunt, 1985), among others. This paper neglects a proper treatment of mass terms. We will simply treat them as countable terms. Thus, baker's honey is represented via a predicate $bakersHoney(x)$, where x has to be thought as an "atomic countable unit" of baker's honey.

Finally, it seems that the sentences in (35) do not contain expressions denoting FOL constants. In legal documents, in the standard case these are denoted by specific *named entities*: institutions such as "the Court of Justice", dates such as "16/2/1998", quantities of money such as "1000 euros", references to other legal documents such as "the Directive 98/5/EC", etc.

However, a closer look reveals that legal texts contain many other named entities. For instance, in (35.b), "the term 'honey' " and "the term 'baker's honey' " do not refer to honey and baker's honey, i.e., the two *foods*, but to the two *expressions* used in English to talk about the two foods. Those may be used, for instance, to label jars containing honey and baker's honey. These two expressions are then represented via FOL constants.

6.1 Formalizing the examples

In our logical framework we represent the obligation in (35.a) as:

[19]According to the definitions in the directive, "baker's honey" is "honey which is (a) suitable for industrial uses or as an ingredient in other foodstuffs which are then processed and (b) may have a foreign taste or odour, or have begun to ferment or have fermented, or have been overheated." A specific definition may be added to the TBox, i.e., the set C, in order to represent this meaning.

(36) $\forall_x \forall_y (\; \exists_{e_1} \exists_{e_2} [(Rexist\ e_1) \wedge (lawyer\ x) \wedge (MS\ y) \wedge (wish'\ e_1\ x\ e_2) \wedge$
$(practice'\ e_2\ x) \wedge (in\ \ e_2\ y) \wedge diffFrom(y\ f_1(x))],$
$\exists_{e_3} [(Rexist\ e_3) \wedge (register'\ \ e_3\ \ x) \wedge (at\ e_3\ \ f_2(y))] \;)$

As discussed above, the noun phrases "that in which he obtained his professional qualification" and "the competent authority in that State" are formalized in terms of two functions, which we called f_1 and f_2 in (36). f_1 takes a lawyer and returns *the* country where he obtained his qualification; f_2 takes a Member State and returns *the* competent authority in that state. At the implementation level, f_1 and f_2 may correspond to queries on some databases that store information about lawyers practising in Europe and EU Member States.

On the other hand, as discussed above in section 3, the predicate *wish* may be subject to different legal interpretations, whose proper handling requires the introduction of defeasible axiom schemas in the TBox. For instance, we can take the literal interpretation of "wish" as default interpretation by converting the implication in (20) into an Input/Output generator to be added to the set C, i.e., the set of constitutive rules:

(37) $\forall_x \forall_y \forall_{e_1} \forall_{e_2} \forall_{e_3} \forall_{e_n} \forall_{e_a} (\; (lawyer\ x) \wedge (MS\ y) \wedge (say'\ e_1\ \ x\ \ e_2) \wedge$
$(wish'\ e_2\ \ x\ \ e_3) \wedge (practice'\ e_3\ \ x) \wedge (in\ e_3\ \ y) \wedge$
$(normalSP'\ \ e_n\ \ e_1) \wedge (and'\ \ e_a\ \ e_1\ \ e_n) \wedge (Rexist\ \ e_a),$
$(Rexist\ \ e_2) \;)$

The permission in (35.b) is similarly formalized in (38). Of course, since (35.a) is an obligation while (35.b) is a permission, (36) is inserted in the set O of the Input/Output system implementing regulative norms, while (38) is inserted in the set P of the same Input/Output system.

(38) $\forall_y (\exists_x \exists_{e_1} [(Rexist\ e_1) \wedge (ingrOf'\ e_1\ x\ y) \wedge (bakerHoney\ \ x) \wedge (foodStuff\ \ y)],$
$\exists_{e_2} [(Rexist\ e_2) \wedge (substitute'\ \ e_2\ \ \mathtt{T}_h\ \mathtt{T}_{bh}) \wedge (in\ \ e_2\ \ f_3(y))])$

As discussed above, the noun phrases "the term 'honey' " and "the term 'baker's honey' " correspond to FOL constants, which we called $\mathtt{T}_h$ and $\mathtt{T}_{bh}$; f_3 is a FOL function that takes a compound foodstuff and returns its product name.

Note that the verbal construction "may be used ... instead of" has been represented via the predicate *substitute*. As in the case of *wish* in (36), *substitute* may be subject to multiple legal interpretations: judges are in charge of deciding whether, in specific contexts, the way the term 'honey' has been used in the product name instead of the term 'baker's honey' is compliant with (35.b), i.e., whether that way indeed entails the predicate *substitute*.

Note also that the variable x only occurs in the LHS of the formula, thus it is existentially quantified. The formula in (38) reads as follows: for each compound foodstuff y for which it is "true" (in the sense that it really exists in the current world) the fact that one of its ingredients is baker's honey, then the norm permits the real existence of the fact that the term 'honey' is substituted by the term 'baker's honey' in the product name of y.

Finally, (35.c) is formalized via *two* reified Input/Output logic generators. As pointed out above at the end of the Introduction, (35.c) contains both a permission and an obligation. Airport users are allowed to use the tailored services or dedicated terminal or part of a terminal. On the other hand, Member States are obliged to guarantee that all airports allow their users to use them.

It seems then that the easiest way to represent (35.c) is to associate it with both an obligation and a permission, and keep them independent of one another. In other words, the permission will be *not* included in the obligation, as the wording would suggest at first glance. Rather, the obligation instantiates the same logical items used in the permission, i.e. it "copies" its formalization. Of course, for the sake of computational efficiency, at the implementation level the two formalizations may be somehow related, in order to evaluate the logical items that occur in both only once and then propagate their truth values to all copies. Nevertheless, at the level of the logical forms, such implementation details are of course immaterial.

The generator in (39) represents the permission in (35.c): airport users wishing to use the tailored services or dedicated terminal or part of a terminal are permitted to have access to these services or terminal or part of a terminal.

$$(39)\qquad \forall_x\forall_y(\ \exists_{e_1}\exists_{e_2}[(Rexist\ e_1) \wedge (airpUser\ x) \wedge (tsOrDtOrPt\ y) \wedge (wish'\ e_1\ x\ e_2) \wedge (use'\ e_2\ x\ y)],\ \exists_{e_3}[(Rexist\ e_3) \wedge (access'\ e_3\ x\ y)]\ )$$

The unary predicate *tsOrDtOrPt* is true if its argument is either a tailored service or a dedicated terminal or a part of a terminal. This predicate has been introduced for the sake of (39)'s readability only. The following axiom schema must be added to C in order to get the intended meaning. Note that (40) uses the boolean connective '$\vee$', belonging to standard FOL.

$$(40)\qquad \forall_y(\ (ts\ y) \vee (dt\ y) \vee (pt\ y),\ (tsOrDtOrPt\ y)\ )$$

The axiom schema in (40) states that a tailored service or a dedicated terminal or a part of a terminal *counts as* a "*tsOrDtOrPt*". Note that, via the axioms in (6) and (9), it is possible to build a fully-reified version of (40), which is equivalent but clearly more verbose than (40):

$$(41)\ \forall_e\forall_y(\ (Rexist\ e_{o_1}) \wedge (or'\ e_{o_1}\ e_1\ e_{o_2}) \wedge (or'\ e_{o_2}\ e_2\ e_3) \wedge (ts'e_1\ y) \vee (dt'\ e_2\ y) \vee (pt'\ e_3\ y),\ \exists_{e_5}[(Rexist\ e_5) \wedge (tsOrDtOrPt'\ e_5\ y)]\ )$$

The generator representing the obligation in (1.c) is then drawn from (39). We instantiate the same predicates and we add other ones requiring each Member State z to guarantee the permission e_3. The generator is shown in (42).

$$(42)\ \forall_x\forall_y\forall_z(\exists_{e_1}\exists_{e_2}[(\mathit{Rexist}\ e_1) \wedge (\mathit{airpUser}\ x) \wedge (\mathit{tsOrDtOrPt}\ y) \wedge (\mathit{MS}\ z) \wedge (\mathit{wish'}\ e_1\ x\ e_2) \wedge (\mathit{use'}\ e_2\ x\ y)],\ \exists_{e_4}\exists_{e_3}[(\mathit{Rexist}\ e_4) \wedge (\mathit{guarantee'}\ e_4\ z\ e_3) \wedge (\mathit{access'}\ e_3\ x\ y)]\ )$$

We stress again: (42) does not include (39): the predicates occurring in the former are *copies* of the ones occurring in the latter. And, (42) is inserted in the set O of the Input/Output system implementing regulative norms while (39) is inserted in P. Of course, this does not necessarily mean that, at the implementation level, these predicates must be evaluated twice. To enhance computational efficiency, at the implementation level the two formalizations may be somehow related, in order to evaluate the logical items only once and then propagate their truth values to all copies. Such implementation details are of course beyond the goal of the present paper.

The following axiom schema defines a default semantics for *guarantee*:

$$(43)\ \forall_{e_1}\forall_{e_2}\forall_{e_1^n}\forall_{e_2^n}\forall_x(\ (\mathit{guarantee'}\ e_1\ x\ e_2) \wedge (\mathit{not'}\ e_2^n\ e_2) \wedge (\mathit{Rexist}\ e_2^n),\ (\mathit{not'}\ e_1^n\ e_1) \wedge (\mathit{Rexist}\ e_1^n)\ )$$

In (43), e_1 is the fact that an individual x is in charge of guaranteeing another eventuality e_2. (43) states that if e_2 does not really exists then e_1 does not really exist too. For instance, in (42), if e_3 does not really exist, i.e., the user x does not have access to y, then also e_4, i.e., the fact that the Member State z guarantees e_3, does not really exist. In other words, if the user x does not have access to y, then the Member State z is *not* guaranteeing the access to y. Therefore, the RHS of the generator in (42) turns out to be false, and we can infer that the obligation in (42) has been violated.

Again, we can create a defeasible version of (43) via the methodology drawn from Circumscriptive Logic that has been explained above in section 3.2, i.e., by adding in (43) a predicate stating that *guarantee* has the (default) meaning in (43) if and only if it is "normal" to assume so. The same may be done for the predicate *access*: we may add in the TBox a defeasible axiom schema assigning the literal meaning of the verb to the predicate. According to that semantics, if a user makes an attempt to access the services and he does not manage, then the accessibility of the services does not really exist. A judge may later override that (default) semantics by stating that the user could have tried more than one time before saying that the services were not accessible.

7 Future works: the forthcoming General Data Protection Regulation (GDPR), a case study

The previous sections have proposed a new general logical framework which may be possibly used to represent the meaning of norms expressed in natural language, e.g. those found in existing legislation. Logical representations have been exemplified on some norms selected from a corpus of EU directives that we used in our past research projects.

This section discusses some future applicability of reified Input/Output logic, namely a case study in which the logical framework will be employed. In our future works we want to formalize the norms occurring in the upcoming European General Data Protection Regulation (GDPR)[20], which will enter into force from 25 May 2018, and to *correlate* them with the norms occurring in existing (industrial) ISO standards.

ISO standards do not *per se* guarantee legal compliance[21]. On the other hand, ISO standards can be certified by means of auditing procedures from qualified bodies, so that the adoption of ISO standards may provide arguments to demonstrate that adequate measures were taken to achieve legal compliance.

Therefore, the development of a knowledge base of machine-readable representations correlating the GDPR and the ISO standards could, in our view, help legal experts to assess the requirements of the GDPR that have been met by implementing certain ISO standards, as well as to distinguish them from those that still need to be fulfilled.

Note that also correlations between the GDPR and the ISO standards are subject to legal interpretation, i.e., they may be overridden by judges in court or by other legal authorities. The knowledge base must be able to keep track of the different legal interpretations over time, thus the need of constructs implementing defeasibility such as the one adopted in reified Input/Output logic.

In this section, we present an example of correlation between an obligation in the GDPR and one in the ISO/IEC 27001:2013 standard[22], in order to exemplify the use of reified Input/Output logic in our future research projects. The two obligations are shown in (44).

(44) a. GDPR, Article 46.1: "[...] a controller or processor may transfer personal data to a third country or an international organization only if the controller or processor has provided appropriate safeguards [...]"

b. ISO 27001, Article. 13.2.1: "Formal transfer policies, procedures and controls shall be in place to protect the transfer of information through the use of all types of communication facilities."

The formalization of (44.a-b) in reified Input/Output logic is rather straightforward. The predicates used in the ABox formulae will reflect the vagueness of the terms occurring in the sentences. Then, as discussed above, further axiom schemas could be added to the TBox, in order to define/constrain these predicates, as well as to attribute them default interpretations and stronger (overriding) ones. Correlations between the GDPR and the ISO standards will

[20]Regulation (EU) 2016/679 of the European Parliament and of the Council of 27 April 2016 on the protection of natural persons with regard to the processing of personal data and on the free movement of such data (General Data Protection Regulation), `http://data.europa.eu/eli/reg/2016/679/oj`.

[21]Except when the standard is endorsed by the law itself; such exceptions are however rare.

[22]`http://www.iso.org/iso/catalogue_detail?csnumber=54534`.

be achieved via additional axiom schemas connecting at least two predicates that respectively occur in a formula associated with a GDPR norm and in a formula associated with a norm of the ISO standards.

The obligations in (44.a-b) are respectively formalized in (45.a-b):

(45) a. $\forall_k(\exists_e\exists_x\exists_y\exists_z\exists_w[$

$(\mathit{dataController}\ x) \wedge (\mathit{personalData}\ y) \wedge (\mathit{dataSubject}\ z) \wedge$

$(\mathit{thirdCountryOrIntOrg}\ w\ z) \wedge (\mathit{Rexist}\ e) \wedge$

$(\mathit{transfer'}\ e\ x\ y\ w) \wedge (\mathit{instrOf}\ e\ k)\ ],$

$(\mathit{appropriate}\ k)\ )$

b. $\forall_k(\exists_e\exists_x\exists_y\exists_w[$

$(\mathit{information}\ y) \wedge (\mathit{Rexist}\ e) \wedge (\mathit{transfer'}\ e\ x\ y\ w) \wedge (\mathit{instrOf}\ e\ k)\ ],$

$(\mathit{secure}\ k)\ )$

In (45.a), e refers to the event of transfer of y performed by the agent x to w. x has to be a data controller[23] while y refers to personal data of a data subject z. k is the instrument of the action e, i.e., the communication facility through which data are transferred. The predicate *thirdCountryOrIntOrg* is a convenient predicate to state that its first argument is either an international organization or a country different from the one where the data subject lives; this is enforced by adding (46.a) to the set of constitutive rules C (TBox):

The obligation in (45.a) requires the communication facility k to be "appropriate". Of course, appropriateness (with respect to the GDPR) of a communication facility is a vague and highly debatable concept, but this has been done on purpose by the legislator(s) who wrote the GDPR, in order to encompass all scenarios where the legislative document must be applied.

Similarly, (45.b) states that, with respect to the ISO/IEC 27001:2013, every communication facility k (involved in a transfer e of information y from an agent x to a receiver z) must be "secure". Note that personal data are a particular subclass of information; this is formalized via the generator in (46.b), which must be added to the set of constitutive rules C.

(46) a. $\forall_y\forall_w\forall_c(\ (\mathit{IntOrg}\ w) \vee ((\mathit{diffFrom}\ w\ c) \wedge (\mathit{countryOf}\ c\ z)),$

$(\mathit{thirdCountryOrIntOrg}\ w\ z)\ )$

b. $\forall_y(\ (\mathit{personalData}\ y), (\mathit{information}\ y)\ )$

As said above, the predicates "*appropriate*" and "*secure*" need to be specialized via further axiom schemas defining their default interpretations. In other words, those axiom schemas will define when, *in real scenarios*, the communication

[23]Or a data processor, but in this example we assume for simplicity that the provision only refers to data controllers.

facility is assumed to be either appropriate (with respect to the GDPR) or secure (with respect to the ISO/IEC 27001).

Since ISO standards are consolidated best practices, it is easier to identify when a communication facility is assumed to be secure with respect to the ISO/IEC 27001. For instance, assuming that an undertaking certified to ISO/IEC 27001 has adopted a formal transfer policy according to which data transfers must take place via email with electronic signature or registered (hard) mail, the following two axiom schemas may be added to the C:

(47) $\forall_x(\ (\mathit{emailWithES}\ x) \wedge (\mathit{normalEeIsAp}\ x),\ (\mathit{secure}\ x)\ )$

$\forall_x(\ (\mathit{regMail}\ x) \wedge (\mathit{normalRmIsAp}\ x),\ (\mathit{secure}\ x)\ )$

On the other hand, the key idea of our future research project is the one of modeling the meaning of predicates used in the formulae representing GDPR norms, such as "*appropriate*", in terms of the predicates used in the formulae representing norms occurring in the ISO standards, thus creating correlations between the GDPR and the ISO standards. In light of this, we add to C the following constitutive rule, stating that being secure with respect to the ISO/IEC 27001 (defeasibly) entails being appropriate with respect to the GDPR.

(48) $\forall_x(\ (\mathit{secure}\ x) \wedge (\mathit{normalSecAppr}\ x),\ (\mathit{appropriate}\ x)\ )$

The entailments in (47) and (48) are defeasible, i.e., they only hold when it is "normal" to assume them. On the contrary, in case a judicial authority later decides, for example, that not all electronic emails are secure means, but only those featuring certain underspecified properties "P", the following axiom schemas may be added to the TBox in order to override (47.a):

(49) $\forall_x(\ (\mathit{emailWithES}\ x),\ \neg(\mathit{normalEeIsAp}\ x)\ )$

$\forall_x(\ (\mathit{emailWithES}\ x) \wedge (P\ x),\ (\mathit{secure}\ x)\ )$

By adding (49), from the knowledge base it cannot be derived anymore that electronic emails are secure with respect to the ISO/IEC 27001 and (in turn) appropriate with respect to the GDPR. Such derivations are allowed only for those electronic emails featuring the property "P".

Similarly, a judicial authority could decide that although some communication facilities are deemed to be secure with respect to the ISO/IEC 27001, nonetheless they are not fully appropriate with the respect to the protection of personal data. In order to be appropriate with the respect to the GDPR, they must also feature another (underspecified) characteristic "C". The following axioms are then added to the knowledge base, in order to override (48):

(50) $\forall_x(\ (\mathit{secure}\ x),\ \neg(\mathit{normalSecAppr}\ x)\ )$

$\forall_x(\ (\mathit{secure}\ x) \wedge (C\ x),\ (\mathit{appropriate}\ x)\ )$

Of course, axioms may be associated with time stamps (although this is not shown in the formulae above), so that (50) overrides (48) only for transfer actions performed after a certain date, i.e., the one from which the judicial authority has been issued. On the other hand, (48) will still assert that email with electronic signature are appropriate communication facilities for all transfer actions performed before that date.

8 Conclusions

The results presented in this paper are collocated within a long-term research aiming at devising a logical framework for modeling norms in natural language, in order to build applications in legal informatics.

We chose Input/Output logic and Hobbs's logic as the two basic frameworks, in Deontic Logic and Natural Language Semantics respectively, that need to be integrated and further developed together.

Input/Output logic is a well-studied framework in normative reasoning; it is indeed a meta-logic that may be wrapped around another logic, termed in this paper as 'the object logic'. Input/Output logic overcomes crucial theoretical limitations of standard deontic logic, above all contrary-to-duty reasoning and moral conflicts. However, it is in turn limited with respect to its expressivity: all Input/Output systems proposed so far in the literature are defined on object logics which are strictly *propositional*. These Input/Output systems are unable to handle norms coming from existing legislation: real-world norms need first-order constructs able to distinguish between terms (named entities, functional expressions, and variables ranging over all members of a certain set of individuals) and predications applied to these terms.

This paper proposes to extend the expressivity of Input/Output logic beyond the propositional level by using constructs coming from Hobbs's logic, which is grounded on the notion of reification. The result is a new logical framework that we called 'reified Input/Output logic'.

Hobbs's logic is able to represent a wide range of linguistic phenomena that can be hardly modeled via standard formalisms for Natural Language Semantics such as Discourse Representation Theory, Minimal Recursion Semantics, and McCarty's Language for Legal Discourse, which are grounded on the well-known Montague's principle of compositionality.

Furthermore, Hobbs's logic allows to distinguish between assertions in the TBox of the ontology, which correspond to constitutive norms in the legal domain, from assertions in the ABox, which correspond to regulative norms in the legal domain. The TBox will contain the axiom schemas that define the semantics of the predicates used in the formulae and that enable inferences from them. On the other hand, the Input/Output generators corresponding to obligations/prohibitions and permissions are intended to populate the ABox. While the assertions in the TBox are FOL formulae in prenex normal form, the assertions in the ABox are mere conjunctions of atomic FOL predicates.

In our view, such a separation between assertions in the TBox from those in the ABox features a twofold advantage.

First of all, the complexity of reified Input/Output logic is fully moved to the TBox of the ontology. In our view, this feature deems pivotal from a computational point of view, in that, in practical applications, the size of the TBox is usually much lower than the size of the ABox. Thus, the complexity of the logic is expected to be easier to control. While first-order logic is known to be semi-decidable, we can restrict the expressivity of the axiom schemas in the TBox so that the Input/Output systems using them turn out to be tractable, i.e. usable in practical applications. Computational complexity has been identified as a thorny issue for Input/Output logic, in that main reasoning tasks in Input/Output systems turn out to be intractable even when the object logic is taken to be propositional logic only (Sun and Robaldo, 2016), (Sun and Robaldo, 2017). In our future works, we will investigate the dependencies between the expressivity/complexity of the axiom schemas and the one of the Input/Output systems using them.

Of course, in case applications will require more expressivity, i.e. in case we are forced to sacrifice computational complexity in order to express the semantics we *do* need in our applications, the implemented Input/Output system must include a way to block long-time derivations, e.g. a timeout. This appears to be a very naïve solution, but it is indeed the only reasonable way to handle intractable tasks in practical applications. For instance, in the TACITUS system (Hobbs, 1986) (Montazeri and Hobbs, 2011), which implements Hobbs's logic, there is no control on the FOL formulae allowed in the knowledge base, so that computational complexity is constrained by positing time thresholds and setting maximum depth of inferences (cf. (Ovchinnikova et al., 2011)).

Secondly, the reified Input/Output generators in the ABox, being mere conjunctions of atomic FOL predicates, are easier to read and to edit by legal practitioners, who usually have little expertise in logic. They could then actively collaborate in the building of (large) ABoxes of formulae representing regulative norms occurring in existing legislation.

Specifically, we plan, in our future works, to develop ad-hoc semi-automatic systems such as editors to help the manual construction of ABoxes containing the assertive contextual statements, i.e., the regulative norms.

On the one hand, we need to overcome the limitation of the *manual* translation of the norms, which would be highly time-consuming and error-prone. On the other hand, we believe that current NLP technologies, even at the best of their performances, are however unable to *automatically* translate from text to generators in our framework with a reasonable level of accuracy. For this reason, we advocate a translation of the norms in a *semi-automatic* fashion.

Automatic means are intended to assist the annotation. For instance, background reasoners may be incorporated in the editor in order to check the consistency of the knowledge base as long as new formulae are added to it. It is likewise important to guarantee a uniformity of translations for excerpts of text that are similar to each other. The problem concerns translations in general, e.g., translations of documents from English to French. Here, human translators are helped by tools such a the `Trados Studio`[24], which suggest them how to translate a sentence on the basis of the translations of similar sentences that have been previously established and stored in the translation memory. By looking at previous translations, human-translators are induced to translate new sentences in a similar (uniform) way. Previously translated sentences are found via text-similarity pattern-recognition NLP techniques such as (Mihalcea, Corley, and Strapparava, 2006) and (Boella et al., 2014).

The first case study for the ad-hoc editor we are planning to implement will be the norms included in the recently-approved General Data Protection Regulation (GDPR) and in existing ISO standards, as discussed in section 7 above. The case study will possibly require to study new special sub-types of obligations and permissions such as those studied in (Governatori et al., 2013) and (Makinson and van der Torre, 2003), as well as the dependencies between deontic predications and other linguistic phenomena. For instance, (Hashmi, Governatori, and Wynn, 2014) study the connections between deontic assertions and time and model them in a framework based on Event Calculus. In our future works, we plan to properly deal with time by incorporating the insights at the basis of `OWL-Time`[25] (Hobbs and Pan, 2004) and by, of course, adapting and extending them to the legal domain in light of the work done by (Hashmi, Governatori, and Wynn, 2014), among others. Some preliminary work in that sense has been already done by the first author of this paper in (Robaldo et al., 2011).

In line with what is done in past research in both Input/Output logic and in Hobbs's logic, we plan to handle all such further refinements, stemming from assertions occurring in existing legislation, by defining new specific Input/Output axioms and/or new specific Hobbs's axioms schema, while proving they are sound and complete with respect to the desired semantics.

References

Ajani, G., G. Boella, L. Di Caro, L. Robaldo, L. Humphreys, S. Praduroux, P. Rossi, and A. Violato. 2017. The european legal taxonomy syllabus: A multi-lingual, multi-level ontology framework to untangle the web of european legal terminology. *Applied Ontology*, to appear.

Alchourrón, C.E. and E. Bulygin. 1984. Permission and permissive norms. In *Theorie der Normen*. Duncker-Humblot, Berlin.

Alshawi, H., editor. 1992. *The Core Language Engine*. Mit Press, Cambridge, MA.

[24]`http://www.translationzone.com/products/sdl-trados-studio`

[25]See `http://www.w3.org/TR/owl-time` and `http://www.isi.edu/~hobbs/owl-time.html`. See also `http://www.isi.edu/~hobbs/bgt-time.text`, where some predicates and axiom schemas to associate events with time instants or time intervals are defined.

Bach, E. 1981. On time, tense, and aspect: An essay in english metaphysics. In P. Cole, editor, *Radical Pragmatics*. Academic Press, New York, pages 63–81.

Bochman, Alexander. 2003. A logic for causal reasoning. In Georg Gottlob and Toby Walsh, editors, *IJCAI-03, Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Acapulco, Mexico, August 9-15, 2003*, pages 141–146. Morgan Kaufmann.

Bochman, Alexander. 2004. A causal approach to nonmonotonic reasoning. *Artificial Intelligence*, 160(1-2):105–143.

Boella, G., L. Di Caro, L. Humphreys, L. Robaldo, R. Rossi, and L. van der Torre. 2016. Eunomos, a legal document and knowledge management system for the web to provide relevant, reliable and up-to-date information on the law. *Artificial Intelligence and Law*, 4.

Boella, G., L. Di Caro, A. Ruggeri, and L. Robaldo. 2014. Learning from syntax generalizations for automatic semantic annotation. *Journal of Intelligent Information Systems*, 43(2):231–246.

Boella, Guido, Luigi Di Caro, Michele Graziadei, Loredana Cupi, Carlo Emilio Salaroglio, Llio Humphreys, Hristo Konstantinov, Kornel Marko, Livio Robaldo, Claudio Ruffini, Kiril Simov, Andrea Violato, and Veli Stroetmann. 2015. Linking legal open data: Breaking the accessibility and language barrier in european legislation and case law. In *Proceedings of the 15th International Conference on Artificial Intelligence and Law*, ICAIL '15, pages 171–175, New York, NY, USA. ACM.

Boella, Guido, Luigi Di Caro, Daniele Rispoli, and Livio Robaldo. 2013. A system for classifying multi-label text into eurovoc. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Law*, ICAIL '13, pages 239–240, New York, NY, USA. ACM.

Boella, Guido, Luigi Di Caro, and Livio Robaldo, 2013. *Semantic Relation Extraction from Legislative Text Using Generalized Syntactic Dependencies and Support Vector Machines*, pages 218–225. Springer Berlin Heidelberg, Berlin, Heidelberg.

Boella, Guido and Leendert W. N. van der Torre. 2004. Regulative and constitutive norms in normative multiagent systems. In *Principles of Knowledge Representation and Reasoning: Proceedings of the Ninth International Conference (KR2004)*, pages 255–266.

Bunt, H.C. 1985. *Mass Terms and Model-Theoretic Semantics*. Cambridge Studies in Linguistics. Cambridge University Press.

Cariani, Fabrizio, Davide Grossi, Joke Meheus, and Xavier Parent, editors. 2014. *Deontic Logic and Normative Systems - 12th International Conference, DEON 2014, Ghent, Belgium, July 12-15, 2014. Proceedings*, volume 8554 of *Lecture Notes in Computer Science*. Springer.

Carlson, Greg N. 1982. Generic terms and generic sentences. *Journal of Philosophical Logic*, 11(2).

Chomsky, N. 1957. *Syntactic Structures*. Mouton and Co, The Hague.

Collins, M. 2003. Head-driven statistical models for natural language parsing. *Computational Linguistics*, 29(4):589–637.

Copestake, A., D. Flickinger, and I.A. Sag. 2005. Minimal Recursion Semantics. An introduction. *Journal of Research on Language and Computation.*, 2?(3):281–?32.

Davidson, D. 1967. The logical form of action sentences. In Nicholas Rescher, editor, *The Logic of Decision and Action*. University of Pittsburgh Press.

Evans, D. and D. Eyers. 2008. Deontic logic for modelling data flow and use compliance. In *Proceedings of the 6th International Workshop on Middleware for Pervasive and Ad-hoc Computing*, pages 19–24, New York, NY, USA. ACM.

Fornara, N. and M. Colombetti. 2009. Specifying artificial institutions in the event calculus. In *V. Dignum editor, Handbook of Research on Multi-Agent Systems: Semantics and Dynamics of Organizational Models*. IGI Global, pages 335–366.

Gabbay, D., J. Horty, X. Parent, R. van der Meyden, and L. (eds.) van der Torre. 2013. *Handbook of Deontic Logic and Normative Systems*. College Publications.

Galton, Antony. 2006. Operators vs. arguments: The ins and outs of reification. *Synthese*, 150(3):415–441.

Governatori, G., F. Olivieri, A. Rotolo, and S. Scannapieco. 2013. Computing strong and weak permissions in defeasible logic. *Journal of Philosophical Logic*, 6(42):799–829.

Governatori, Guido, Antonino Rotolo, and Giovanni Sartor. 2015. Deontic defeasible reasoning in legal interpretation. In Katie Atkinson, editor, *The 15th International Conference on Artificial Intelligence & Law*, San Diego, USA.

Hashmi, M., G. Governatori, and M. Wynn. 2014. Modeling obligations with event-calculus. In Antonis Bikakis, Paul Fodor, and Dumitru Roman, editors, *Rules on the Web. From Theory to Applications*, volume 8620 of *Lecture Notes in Computer Science*. Springer International Publishing, pages 296–310.

Hobbs, J. R. 2008. Deep lexical semantics. In *Proc. of the 9th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing-2008)*, Haifa, Israel.

Hobbs, J. R. and F. Pan. 2004. An ontology of time for the semantic web. *ACM Transactions on Asian Language Processing (TALIP): Special issue on Temporal Information Processing*, 3(1):66–85.

Hobbs, J.R. 1986. Overview of the TACITUS project. *Computational Linguistics*, 12(3).

Hobbs, J.R. 1998. The logical notation: Ontological promiscuity. In *Chapter 2 of Discourse and Inference*. Available at http://www.isi.edu/∼hobbs/disinf-tc.html.

Hobbs, J.R. 2001. Syntax and metonymy. In Bouillon P. e Busa F., editor, *The Language of Word Meaning*. Cambridge University Press, pages 302–361.

Jørgensen, Jorgen. 1937. Imperatives and logic. *Erkenntnis*, 7:288–296.

Kamp, H. and U. Reyle. 1993. *From Discourse to Logic: an introduction to model-theoretic semantics, formal logic and Discourse Representation Theory*. Kluwer Academic Publishers, Dordrecht.

Kowalski, R and M Sergot. 1986. A logic-based calculus of events. *New Generation Computing*, 4(1):67–95.

Liebwald, Doris. 2013. Vagueness in law: A stimulus for 'artificial intelligence & law'. In *Proc. of the 14th International Conference on Artificial Intelligence and Law*, ICAIL '13, pages 207–211.

MacCormick, N. and R.S. Summers. 1991. *Interpreting Statutes: A Comparative Study*. Applied legal philosophy. Dartmouth.

Makinson, D. and L. Van der Torre. 2001. Constraints for input/output logics. *Journal of Philosophical Logic*, 30(2):155–185.

Makinson, David and Leendert van der Torre. 2003. Permission from an input/output perspective. *Journal of Philosophical Logic*, 32:391–416.

Makinson, David and Leendert W. N. van der Torre. 2000. Input/output logics. *Journal of Philosophical Logic*, 29(4):383–408.

McCarthy, J. 1980. Circumscription: A form of nonmonotonic reasoning. *Artificial Intelligence*, (13):27–39.

McCarty, L. T. 1989. A language for legal discourse i. basic features. In *Proc. of the 2nd International Conference on Artificial Intelligence and Law (ICAIL '89)*, ACM Press.

McCarty, L. T. 2002. Ownership: A case study in the representation of legal concepts. *Artificial Intelligence and Law*, 10(1-3):135–161.

McCarty, L. T. 2007. Deep semantic interpretations of legal texts. In *The Eleventh International Conference on Artificial Intelligence and Law, Proceedings of the Conference, June 4-8, 2007, Stanford Law School, Stanford, California, USA*, pages 217–224.

Mihalcea, Rada, Courtney Corley, and Carlo Strapparava. 2006. Corpus-based and knowledge-based measures of text semantic similarity. In *Proceedings of the 21st National Conference on Artificial Intelligence - Volume 1*, AAAI'06, pages 775–780. AAAI Press.

Miller, R. and M. Shanahan. 1999. The event calculus in classical logicalternative axiomatizations. *Electronic Transactions on Artificial Intelligence*, 16(4).

Miltsakaki, Eleni, Livio Robaldo, Alan Lee, and Aravind Joshi, 2008. *Sense Annotation in the Penn Discourse Treebank*, pages 275–286. Springer Berlin Heidelberg, Berlin, Heidelberg.

Montague, R. 1970. Universal grammar. *Theoria*, 36(3):373–398.

Montazeri, N. and J.R. Hobbs. 2011. Elaborating a knowledge base for deep lexical semantics. In J. Bos and S. Pulman, editors, *In Proc. of 9th International Workshop on Computational Semantics*, pages 195–204.

Ovchinnikova, E., N. Montazeri, T. Alexandrov, J.R. Hobbs, M.C. McCord, and R. Mulkar-Mehta. 2011. Abductive reasoning with a large knowledge base for discourse processing. In *Proc. of the Ninth International Conference on Computational Semantics (IWCS 2011)*, pages 225–234.

Parent, Xavier. 2011. Moral particularism in the light of deontic logic. *Artif. Intell. Law*, 19(2-3):75–98.

Parent, Xavier and Leendert W. N. van der Torre. 2014. "sing and dance!" - input/output logics without weakening. In Cariani et al. (Cariani et al., 2014), pages 149–165.

Paschke, A. and M. Bichler. 2005. SLA representation, management and enforcement. In *2005 IEEE International Conference on e-Technology, e-Commerce, and e-Services (EEE 2005), 29 March - 1 April 2005, Hong Kong, China*, pages 158–163.

Peller, G. 1985. *The metaphysics of American law*, volume 73. California Law Review.

Prasad, R., N. Dinesh, A. Lee, E. Miltsakaki, L. Robaldo, A. Joshi, and B. Webber. 2008. The penn discourse treebank 2.0. In *Proc. of the 6th International Conference on Language Resources and Evaluation (LREC 2008).*

Robaldo, L. 2007. *Dependency Tree Semantics.* Ph.D. thesis, University of Turin.

Robaldo, L. 2010a. Independent set readings and generalized quantifiers. *The Journal of Philosophical Logic*, 39(1):23–58.

Robaldo, L. 2010b. Interpretation and inference with maximal referential terms. *The Journal of Computer and System Sciences*, 76(5):373–388.

Robaldo, L. 2011. Distributivity, collectivity, and cumulativity in terms of (in)dependence and maximality. *The Journal of Logic, Language, and Information*, 20(2):233–271.

Robaldo, L. 2013. Conservativity: a necessary property for the maximization of witness sets. *The Logic Journal of the IGPL*, 21(5):853–878.

Robaldo, L. and J. Di Carlo. 2013. Flexible disambiguation and expressive completeness in dependency tree semantics. *The Journal of Semantics*, 30(2).

Robaldo, L. and E. Miltsakaki. 2014. Corpus-driven semantics of concession: Where do expectations come from? *Dialogue&Discourse*, 5(1).

Robaldo, L., J. Szymanik, and B. Meijering. 2014. On the identification of quantifiers' witness sets: a study of multi-quantifier sentences. *The Journal of Logic, Language, and Information.*, 23(1).

Robaldo, Livio, Tommaso Caselli, Irene Russo, and Matteo Grella. 2011. From italian text to timeml document via dependency parsing. In *Computational Linguistics and Intelligent Text Processing - 12th International Conference, CICLing 2011, Tokyo, Japan, 2011.*, pages 177–187.

Sartor, G. 2005. *Legal Reasoning: A Cognitive Approach to the Law.* Treatise of legal philosophy and general jurisprudence / ed.-in-chief Enrico Pattaro. Springer.

Searle, John R. 1995. *The construction of social reality.* The Free Press, New York.

Sergot, M. J., F. Sadri, R. A. Kowalski, F. Kriwaczek, P. Hammond, and H. T. Cory. 1986. The british nationality act as a logic program. *Commun. ACM*, 29(5):370–386.

Sun, X. and L. Robaldo, 2016. *Logic and Games for Ethical Agents in Normative Multi-agent Systems*, pages 367–375. Springer International Publishing.

Sun, X. and L. Robaldo. 2017. On the complexity of input/output logic. *Manuscript submitted to the Journal of Applied Logic.*

Sun, Xin and Leendert W. N. van der Torre. 2014. Combining constitutive and regulative norms in input/output logic. In Cariani et al. (Cariani et al., 2014), pages 241–257.

van Benthem, Johan. 1989. Polyadic quantifiers. *Linguistics and Philosophy*, 12(4):437–464.

Vibert, Hughes-Jehan, Pierre Jouvelot, and Benoît Pin. 2013. Legivoc - connectings laws in a changing world. *Journal of Open Access to Law*, 1(1).

von Wright, G.H. 1959. *On the Logic of Negation*. Societas scientiarum fennica. Commentationes physico-mathematicae, XXII, 4.

Winkels, R.G.F., editor. 2015. *The OpenLaws project: Big Open Legal Data*, Proceedings of the International Legal Informatics Symposium (IRIS 2015), Salzburg, pp. 189-196.