Towards a Mapping of Deontic Logic onto an Abductive Framework

Marco Alberti[1], Marco Gavanelli[1], Evelina Lamma[1], Paola Mello[2], Giovanni Sartor[3], and Paolo Torroni[2]

[1] Dip. di Ingegneria - Università di Ferrara - Via Saragat, 1 - 44100 Ferrara, Italy.
`{malberti|mgavanelli|elamma}@ing.unife.it`
[2] DEIS - Università di Bologna - Viale Risorgimento, 2 - 40136 Bologna, Italy.
`{pmello,ptorroni}@deis.unibo.it`
[3] CIRSFID - Università di Bologna - Via Galliera, 2 - 40100 Bologna, Italy.
`sartor@cirfid.unibo.it`

Abstract. A number of approaches to agent society modeling can be found in the Multi-Agent Systems literature which exploit (variants of) Deontic Logic. In this paper, after briefly mentioning related approaches, we focus on the Computational Logic (CL) approach for society modeling developed within the UE IST-2001-32530 Project (named SOCS), where obligations and prohibitions are mapped into abducible predicates (respectively, positive and negative expectations), and norms ruling the behavior of members are represented as abductive integrity constraints. We discuss how this abductive framework can deal with Deontic Logic concepts, by introducing additional integrity constraints.

1 Introduction

Several researchers have studied the concepts of norms, commitments and social relations in the context of Multi-Agent Systems [1]. Furthermore, a lot of research has been devoted in proposing architectures for developing agents with social awareness (see, for instance [2]).

Several approaches to agent society modeling have been grounded on Deontic Logic, norms and institutions (e.g., [3–7]). Deontic Logic enables to address the issue of explicitly and formally defining norms and dealing with their possible violations. It represents norms, obligations, prohibitions and permissions, and enables one to deal with predicates like "p ought to be done", "p is forbidden to be done", "p is permitted to be done".

In the context of the UE IST Programme, two projects (namely ALFEBIITE [8] and SOCS [9]) have investigated the application of logic-based approaches for modeling *open*[4] societies of agents. In particular, the former focuses on the formalization of a society of agents using Deontic Logic, and the latter on a specification of an agent society which, being based on computational logic, is also executable as a verification program.

[4] For a definition of openness see [10, 11].

The ALFEBIITE approach (presented, for instance, by Artikis et al. [10]) consists of a theoretical framework for providing executable specifications of particular kinds of multi-agent systems, called open computational societies, and present a formal framework for specifying, animating and ultimately reasoning about and verifying the properties of systems where the behavior of the members and their interactions cannot be predicted in advance. Three key components of computational systems are specified, namely the social constraints, social roles and social states. The specification of these concepts is based on and motivated by the formal study of legal and social systems (a goal of the ALFEBIITE project), and therefore operators of Deontic Logic are used for expressing legal social behavior of agents [12, 13]. The ALFEBIITE logical framework comprises a set of building blocks (including doxastic, deontic and praxeologic notions) as well as composite notions (including deontic right, power, trust, role and signaling acts).

The SOCS [9] approach to society modeling can be conceived as complementary to these efforts, since it is especially oriented toward Computational Logic aspects, and it was developed with the purpose of providing a computational framework that can be directly used for automatic verification of properties such as compliance to interaction protocols. The SOCS social model represents social norms as abductive integrity constraints, where abducibles express expectations (positive and negative) on the behavior of members of the society. We map expectations into first-class abducible predicates (**E** for positive and **EN** for negative expectations). The social framework is grounded on Computational Logic (CL, for short), and a declarative abductive semantics has been defined in [14]. Operationally, the application of abductive integrity constraints (named Social Integrity Constraints) by a suitable abductive proof procedure adjusts the set of social expectations as the society acquires new knowledge from the environment in terms of happened social events. The idea of expected behavior is strictly related to *deontic operators*, and it was inspired by Deontic Logic. However, in SOCS we did not exploit the full power of the standard Deontic Logic, but only abductive integrity constraints on events that are expected to happen or not to happen, and we mapped expectations into first-class abducible predicates (**E** and **EN**, see the next section). Grounding the social framework on CL also smoothly provides an operational counterpart for it, in terms of an abductive proof procedure (named $\mathcal{S}$CIFF), which was obtained by extending the IFF proof procedure, proposed by Fung and Kowalski in [15].

Nonetheless, we believe that an approach grounded on CL, and abductive integrity constraints in particular, can be exploited in order to also deal with (fragments of) Deontic Logic. This paper is meant to present a first step towards a mapping of existsing formalizations of Deontic Logic onto an abductive computational framework such as SOCS'. This is achieved by means of additional (meta) integrity constraints. One of the main purposes of such mapping is to exploit the operational counterpart of the SOCS social framework (see, for instance, [30]) and the (modular) implementation of $\mathcal{S}$CIFF (suitably extended by the additional meta constraints) for the on-the-fly verification of conformance of agents to norms specified in the chosen Deontic Logic.

The paper is organized as follows. In Section 2, we briefly recall the SOCS social abductive model, and its abductive semantics. After briefly recalling Deontic Logic in Section 3, in Section 4 we show how two of its variants can be mapped into the SOCS social framework, by simply varying additional (meta) integrity constraints. Section 5 briefly discusses related work. Then we conclude, and mention future work.

2 The SOCS social model

Although the SOCS project also provides a logic-based model for individual agents (see, for instance, [16]), in this paper we abstract away from the internals of the individual agent and adopt an *external* perspective: we focus on the *observable* agent behavior, regardless of its motivation from an internal perspective. In this way, the model does not constrain the number and/or the type of agents that a society may be composed of.

The SOCS model describes the knowledge about an agent society in a declarative way. Such knowledge is mainly composed of two parts: a *static* part, defining the society organizational and "normative" elements (encoded in what we call *Social Integrity Constraints*, as we will show below), and a *dynamic* part, describing the "socially relevant" events, that have so far occurred (*happened* events). Depending on the context in which this model is instantiated, socially relevant events could indeed be physical actions or transactions, such as electronic payments. In addition to these two categories of knowledge, information about social *goals* is also maintained.

Based on the available history of events, on its specification of social integrity constraints and its goals, the society can define the social events that are expected to happen and those that are expected *not* to happen. We call these events *social expectations*; from a normative perspective, they reflect the "ideal" behavior of the agents.

2.1 Representation of the society knowledge

The knowledge in a society S is given by the following components:

– a (static) *Social Organization Knowledge Base*, denoted $SOKB$;
– a (static) set of *Social Integrity Constraints* (IC_S), denoted $\mathcal{IC}_S$; and
– a set of *Goals* of the society, denoted by $\mathcal{G}$.

In the following, the terms *Atom* and *Literal* have the usual Logic Programming meaning [17].

A society may evolve, as new events happen, giving rise to sequence of society instances, each one characterized by the previous knowledge components and, in addition, a (dynamic) *Social Environment Knowledge Base*, denoted $SEKB$.

In particular, $SEKB$ is composed of:

– *Happened events*: atoms indicated with functor **H**;
– *Expectations*: events that should (but might not) happen (atoms indicated with functor **E**), and events that should not (but might indeed) happen (atoms indicated with functor **EN**).

In our context, "happened" events are not all the events that have actually happened, but only those observable from the outside of agents, and relevant to the society. The collection of such events is the history, **HAP**, of a society instance. Events are represented as ground atoms of the form

$$\mathbf{H}(\mathit{Event}[, \mathit{Time}]).$$

For instance, in an electronic commerce context, the following atom:

$$\mathbf{H}(\mathit{tell}(a1, a2, \mathit{offer}(\mathit{scooter}, 1500), d1), 0)$$

could stand for an event about a communicative act *tell* made by agent $a1$, addressed to an agent $a2$, with subject $\mathit{offer}(\mathit{scooter}, 1500)$, at a time 0. $d1$ is, in this case, a dialogue identifier.

Expectations can be

$$\mathbf{E}(\mathit{Event}[, \mathit{Time}]) \qquad \mathbf{EN}(\mathit{Event}[, \mathit{Time}])$$

for, respectively, positive and negative expectations. **E** is a positive expectation about an event (the society expects the event to happen) and **EN** is a negative expectation, (the society expects the event not to happen[5]). Explicit negation ($\neg$) can be applied to expectations.

For instance, in an electronic commerce scenario, the following atom:

$$\mathbf{E}(\mathit{tell}(\mathit{Customer}, \mathit{Seller}, \mathit{accept}(\mathit{Item}, \mathit{Price}), \mathit{Dialogue}), T)$$

could stand for an expectation about a communicative act *tell* made by an agent ($\mathit{Customer}$), addressed to an agent Seller, with subject $\mathit{accept}(\mathit{Item}, \mathit{Dialogue})$, at a time T.

The SOKB is a logic program, consisting of clauses, possibly having expectations in their body. The full syntax of SOKB is reported in Appendix.

The arguments of expectation atoms can be non-ground terms (see [18] for a detailed discussion on variable quantification). Intuitively, variables occurring only in positive expectations are existentially quantified, whereas variables occurring only in negative expectations only are universally quantified.

The following is a sample SOKB clause:

$$\begin{array}{l} \mathit{on\_sale}(\mathit{Item}) \leftarrow \\ \quad \mathbf{E}(\mathit{tell}(\mathit{Seller}, \mathit{Customer}, \mathit{offer}(\mathit{Item}, \mathit{Price}), \mathit{Dialogue}), T0) \end{array} \tag{1}$$

It says that one way to fulfill the goal: "to have a certain *item* on sale," could be to have some agent acting as a seller and offering the item at a certain price to a possible buyer.

[5] **EN** is a shorthand for **E** *not*.

The goal $\mathcal{G}$ of the society has the same syntax as the *Body* of a clause in the SOKB (see Appendix), and the variables are quantified accordingly.

As an example, we can consider a society with the goal of selling items. In order to sell a scooter, the society might expect some agent to embody the role of buyer. The goal of the society could be

$$\leftarrow on\_sale(scooter)$$

and the society might have, in the $SOKB$, a rule such as Eq. 1. Indeed, there could be more clauses specifying other ways of achieving the same goal.

Social Integrity Constraints are in the form of implications. The characterizing part of their syntax is reported in Appendix. For the scope rules and quantifications, see [18]. Intuitively, $\mathcal{IC}_S$ is a set of forward rules, possibly having (a conjunction of) events and expectations in their body and (disjunction of conjunctions of) expectations in their heads. Defined predicates and CLP-like constraints can occur in body and head, as well.

The following ic_S models one (simple) electronic vending rule, stating that each time an offer event happens, the potential buyer has to answer by accepting or refusing by a certain deadline τ.

$$\begin{aligned}
&\mathbf{H}(tell(S, B, offer(Item, Price), D), T0) \rightarrow\\
&\quad \mathbf{E}(tell(B, S, accept(Item, Price), D), T1), T_1 \leq T0 + \tau \vee\\
&\quad \mathbf{E}(tell(B, S, refuse(Item, Price), D), T1), T_1 \leq T0 + \tau
\end{aligned}$$

2.2 Abductive semantics of the Society

SOCS social model has been interpreted in terms of Abductive Logic Programming [19], and an abductive semantics has been proposed for it [14]. Abduction has been widely recognized as a powerful mechanism for hypothetical reasoning in the presence of incomplete knowledge [20–23].

In SOCS social model, the idea is to exploit abduction for defining expected behavior of the agents inhabiting the society, and an abductive proof procedure (named $\mathcal{S}$CIFF, see [18]) to dynamically *generate* the expectations, and possibly perform the *compliance check*. By "compliance check" we mean the procedure of checking that the ic_S are not violated, together with the function of detecting fulfillment and violation of expectations.

Throughout this section, as usual when defining declarative semantics, we always consider the ground version of social knowledge base and integrity constraints, and we do not consider CLP-like constraints. Moreover, we omit the time argument in events and expectations.

First, we formalize the notions of *instance* of a society as an Abductive Logic Program (ALP, for short) [19], and *closure* of an instance. An ALP is a triple $\langle KB, \mathcal{A}, IC\rangle$ where KB is a logic program, (i.e., a set of clauses), $\mathcal{A}$ is a set of predicates that are not defined in KB and that are called *abducibles*, IC is a set of formulas called *Integrity Constraints*. An abductive explanation for a goal G is a set $\Delta \subseteq \mathcal{A}$ such that $KB \cup \Delta \models G$ and $KB \cup \Delta \models IC$, for some notion of entailment $\models$.

Definition 1. *An* instance $\mathcal{S}_{\mathbf{HAP}}$ *of a society* $\mathcal{S}$ *is represented as an ALP, i.e., a triple* $\langle P, \mathcal{E}, \mathcal{IC}_S \rangle$ *where:*

– P *is the SOKB of* $\mathcal{S}$ *together with the history of happened events* $\mathbf{HAP}$*;*
– $\mathcal{E}$ *is the set of* abducible predicates*, namely* $\mathbf{E}$*,* $\mathbf{EN}$*,* $\neg\mathbf{E}$*,* $\neg\mathbf{EN}$*;*
– $\mathcal{IC}_S$ *are the social integrity constraints of* $\mathcal{S}$*.*

The set $\mathbf{HAP}$ characterizes the instance of a society, and represents the set of *observable* and *relevant* events for the society which have already happened. Note that we assume that such events are always ground.

A society instance is closed, when its characterizing history has been closed under the Closed World Assumption (CWA), i.e., when it is assumed that no further event will occur. In the following, we indicate a closed history by means of an overline: $\overline{\mathbf{HAP}}$.

Semantics to a society instance is given by defining those sets of expectations which, together with the society's knowledge base and the happened events, imply an instance of the goal—if any—and *satisfy* the integrity constraints.

In our definition of integrity constraint satisfaction we will rely upon a notion of entailment in a three-valued logic, being it more general and capable of dealing with both open and closed society instances. Therefore, in the following, the symbol $\models$ has to be interpreted as a notion of entailment in a three-valued setting [24], where the history of events is open (resp. closed) for open (resp. closed) instances .

We first introduce the concept of *$\mathcal{IC}_S$-consistent set of social expectations*[6]. Intuitively, given a society instance, an $\mathcal{IC}_S$-consistent set of social expectations is a set of expectations about social events that are compatible with P (i.e., the $SOKB$ and the set $\mathbf{HAP}$), and with $\mathcal{IC}_S$.

Definition 2. ($\mathcal{IC}_S$-consistency) *Given a (closed/open) society instance* $\mathcal{S}_{\mathbf{HAP}}$*, an* $\mathcal{IC}_S$-consistent *set of social expectations* $\mathbf{EXP}$ *is a set of expectations such that:*

$$SOKB \cup \mathbf{HAP} \cup \mathbf{EXP} \models \mathcal{IC}_S \qquad (2)$$

(Notice that for closed instances $\mathbf{HAP}$ *has to be read* $\overline{\mathbf{HAP}}$*).*

$\mathcal{IC}_S$-consistent sets of expectations can be self-contradictory (e.g., both $\mathbf{E}(p)$ and $\neg\mathbf{E}(p)$ may belong to a $\mathcal{IC}_S$-consistent set). To avoid self-contradiction, a number of further *meta* integrity constraints have been taken into account [7]. We will show in Section 3 how these constraints, besides others, can express the basic axiomatizations of Deontic Logic.

Definition 3. (E-consistency) *A set of social expectations* $\mathbf{EXP}$ *is* E-consistent *if and only if for each (ground) term p:*

$$\mathbf{EXP} \cup \{\mathbf{E}(p), \mathbf{EN}(p) \rightarrow false\} \not\models false \qquad (3)$$

[6] With abuse of terminology, we call this notion $\mathcal{IC}_S$-consistency though it corresponds to the theoremhood view rather than to the consistency view defined in [15].
[7] In this notion, we adopt the *consistency view* defined in [15].

Definition 4. (¬-consistency) *A set of social expectations* **EXP** *is* ¬-consistent *if and only if for each (ground) term p:*

$$\mathbf{EXP} \cup \{\mathbf{E}(p), \neg\mathbf{E}(p) \rightarrow false\} \not\models false \tag{4}$$

and:

$$\mathbf{EXP} \cup \{\mathbf{EN}(p), \neg\mathbf{EN}(p) \rightarrow false\} \not\models false \tag{5}$$

Among sets of expectations, we are interested in those satisfying Definitions 2, 3 and 4, i.e., $\mathcal{IC}_S$-, E- and ¬-consistent (we named these sets *closed*, resp. *open*, *admissible*).

Furthermore, a notion of fulfillment (similar, for positive expectations, to the notion of regimentation in Deontic Logic) was introduced in [14], as follows.

Definition 5. (Fulfillment) *Given a (closed/open) society instance $\mathcal{S}_{\mathbf{HAP}}$, a set of social expectations* **EXP** *is* fulfilled *if and only if for all (ground) terms p:*

$$\mathbf{HAP} \cup \mathbf{EXP} \cup \{\mathbf{E}(p) \rightarrow \mathbf{H}(p)\} \cup \{\mathbf{EN}(p) \rightarrow \neg\mathbf{H}(p)\} \not\models false \tag{6}$$

Symmetrically, we define violation when the condition in Definition 5 above is not verified.

Two further notions of goal achievability and achievement were introduced in [14] to support society goal-directed modeling. We refer to [14] for details.

3 Deontic Notions

The birth of modern Deontic Logic can be traced back to the '50s. In the following, we only address the logical properties that are most useful in modeling legal reasoning, and norms, and refrain from addressing the logical background which provides a foundation for those properties.

Deontic Logic enables to address the issue of explicitly and formally defining norms and dealing with their possible violations. It represents norms, obligations, prohibitions and permissions, and enables one to deal with predicates like "p ought to be done", "p is forbidden to be done", "p is permitted to be done".

Being obligatory, being forbidden and being permitted are indeed the three fundamental *deontic statuses* of an action, upon which you can build more articulate normative conceptions. For details, refer to [25], Chapter 15 in particular.

Obligations. To say that an action is *obligatory* is to say that the action is due, has to be held, must be performed, is mandatory or compulsory. This is a very basic notion, not reducible to other, simpler or more familiar, ideas. Obligations are usually represented by formulas as:

$$\mathbf{Obl}\ A$$

where A is any (positive or negative) action description, and **Obl** is the deontic operator for obligation to be read as "it is obligatory that".

Elementary obligations can be distinguished between:

- *elementary positive obligations*, which concern positive elementary actions (e.g., “It is mandatory that John answers me”);
- *elementary negative obligations*, which concern negative elementary actions (e.g., “It is mandatory that John does not smoke”);

Prohibitions. The idea of obligation is paralleled with the idea of *prohibition*. Being forbidden or prohibited is the status of an action that should not be performed. In common language, and legal language as well, prohibitive propositions are expressed in various ways. For example, one may express the same idea by saying “It is forbidden that John smokes”, “John must not smoke”, “There is a prohibition that John smokes”, and so on.

Prohibitions are usually represented by formulas as:

$$\textbf{Forb}\ A$$

where A is any (positive or negative) action description, and **Forb** is the deontic operator for prohibition to be read as “it is forbidden that”.

The notions of obligation and prohibition are logically connected, as explained in the following. Most approaches to Deontic Logic agree in assuming that, for any action A, the prohibition of A is equivalent to the obligation of omitting A:

$$\textbf{Forb}\ A\ =\ \textbf{Obl}\ (NON\ A) \tag{7}$$

Permissions. The third basic deontic status, besides obligations and prohibitions, is *permission*. Permissive propositions are expressed in many different ways in natural language. To express permissions in a uniform way, Deontic Logic uses the operator **Perm**. Permissions are usually represented by formulas as:

$$\textbf{Perm}\ A$$

where A is any (positive or negative) action description, and **Perm** is the deontic operator for permission to be read as “it is permitted that”.

The three basic deontic notions of obligation, prohibition and permission are logically connected. First of all, intuitively when one believes that an action is obligatory, then one can conclude that the same action is permitted.

$$\textbf{Obl}\ A\ \ \textbf{entails}\ \ \textbf{Perm}\ A \tag{8}$$

Since A’s obligatoriness entails A’s permittedness, **Obl** A is incompatible with the fact that A is not permitted:

$$\textbf{Obl}\ A\ \ \textbf{incompatible}\ \ \ NON\ \textbf{Perm}\ A \tag{9}$$

The connection between the obligatoriness of A and the permittedness of A is replicated in the connection between the forbiddenness of A and the permittedness on A’s omission: an action being forbidden entails permission to omit it, i.e.:

$$\textbf{Forb}\ A\ \ \textbf{entails}\ \ \textbf{Perm}\ NON\ A \tag{10}$$

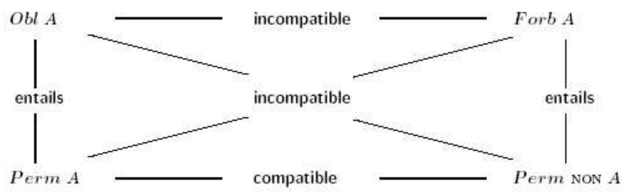

Fig. 1. The first deontic square

A being forbidden entails that the omission of A is permitted. Thus, there is a contradiction between an action being forbidden and the omission of that action not being permitted.

$$\textbf{Forb}\ A\ \ \textbf{incompatible}\quad NON\ \textbf{Perm}\ (NON\ A) \tag{11}$$

All the logical relations between deontic notions we have just described are synthesized in Figure 1. The schema shows that there is an opposition between being obliged and being prohibited: If an action A is obligatory, then its performance is permitted, which contradicts that A is forbidden.

Similarly, if an action A is forbidden, then its omission is permitted, which contradicts that A is obligatory.

It is instead compatible that both an action A is permitted and its omission $NON\ A$ also is permitted. In such a case, A would be neither obligatory nor permitted, but *facultative* (see to [25], Chapter 15).

The deontic qualifications "obligatory" and "forbidden" are complete, in the sense that they determine the deontic status of both the action they are concerned with, and the complement of that action. In fact, on the basis of the equivalence:

$$\textbf{Obl}\ \phi = \textbf{Forb}\ NON\ \phi$$

we get the following two equivalences, the first concerning the case where ϕ is a positive action A, the second concerning the case where ϕ is the omissive action $NON\ A$ (double negations get canceled):

$$\textbf{Obl}\ A = \textbf{Forb}\ NON\ A \tag{12}$$

$$\textbf{Obl}\ NON\ A = \textbf{Forb}\ A \tag{13}$$

Of course, believing that an action is permitted amounts to believing that it is not forbidden:

$$\textbf{Perm}\ A = NON\ \textbf{Forb}\ A \tag{14}$$

This means that not being permitted amounts to being forbidden (just negate both formulas, and cancel double negations):

$$NON\ \textbf{Perm}\ A = \textbf{Forb}\ A \tag{15}$$

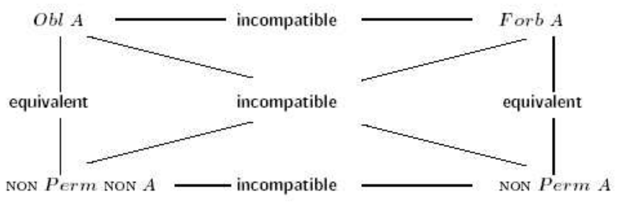

Fig. 2. The second deontic square

From this follows that an action being permitted contradicts that action being prohibited:

$$\textbf{Perm}\ A\ \ \textbf{incompatible}\ \ \textbf{Forb}\ A \tag{16}$$

Similarly, believing that an action is obligatory amounts to excluding that its omission is permitted:

$$\textbf{Obl}\ A = NON\ \textbf{Perm}\ NON\ A \tag{17}$$

Correspondingly, the obligatoriness of an action (entailing the permission to perform it) contradicts the permissiveness of its omission:

$$\textbf{Obl}\ A\ \ \textbf{incompatible}\ \ \textbf{Perm}\ NON\ A \tag{18}$$

The formulas we have just being considering are summarized in the second square of deontic notions, in Figure 2.

4 Mapping Deontic Notions onto the SOCS Social Model

This section shows how the Deontic Logic operators are mapped into SOCS social abductive model. In particular, we first show how the deontic operators can be mapped into SOCS abducible predicates standing for positive and negative expectations about social behavior (and their explicit negation). Then, we show how their logical relations can be mapped into the additional (meta) integrity constraints, considered by the (semantic and) operational machinery.

4.1 Mapping deontic operators onto expectations

There is a natural correspondence between the notion of positive expectation and obligation, and, similarly, between the notion of negative expectation and prohibition. Consequently, since a negative expectation $\mathbf{EN}(A)$ has to be read as *it is expected not A* (i.e., it is a shorthand for $\mathbf{E}(not\ A)$), its (explicit) negation, $\neg\mathbf{EN}(A)$, corresponds to permission of A.

Therefore, the three deontic notions can be mapped into expectations as summarized by the first three lines in Table 1.

Operator	**Abducibile**
Obl A	$\mathbf{E}(A)$
Forb A	$\mathbf{EN}(A)$
Perm A	$\neg\mathbf{EN}(A)$
Perm $NONA$	$\neg\mathbf{E}(A)$

Table 1. Deontic notions as expectations

Furthermore, due to the logical relations among obligation, prohibition and permission discussed in Section 3, the fourth line of Table 1 shows how to map permission of a negative action. Notice that, while both NON and $\neg$ represent the explicit negation of their argument, we keep the different symbols for uniformity with the original contexts.

It is worth noticing, however, that despite this natural mapping the deontic notions and SOCS social expectations are grounded on different semantic approaches, inherited from modal logic the former, and based on abduction the latter.

4.2 Logical relations among deontic operators as abductive integrity constraints

Let us first consider the relations summarized in the second square of deontic notions, in Figure 2. By adopting the mapping summarized in Table 1, the equivalence relations straightforwardly arose from uniform treatment of symbols NON, $\neg$ and *not*, and from their idempotency.

Incompatibility relations summarized in Figure 2 emerge between the notion of obligation and prohibition (horizontal arc), and, respectively, between obligation and permission of opposite, and prohibition and non permission of opposite (diagonal arcs). By adopting the mapping summarized in Table 1, the first incompatibility is captured by SOCS social abductive semantics into the notion of E-consistency (Definition 3), i.e., by requiring that, for each A, the addition to the expectation set of the integrity constraint:

$$\mathbf{E}(A), \mathbf{EN}(A) \rightarrow false$$

does not lead to inconsistency.

The latter two incompatibilities (corresponding to diagonal arcs in Table 1) are captured, instead, by the notion of $\neg$-consistency (Definition 4), i.e., by requiring that, for each A, the addition to the expectation set of the integrity constraints:

$$\mathbf{E}(A), \neg\mathbf{E}(A) \rightarrow false$$

and

$$\mathbf{EN}(A), \neg\mathbf{EN}(A) \rightarrow false$$

does not lead to inconsistency.

The notions of E-consistency and $\neg$-consistency (and associated integrity constraints) also correspond to incompatibility relations in the first square of deontic notions, in Figure 1.

Furthermore, the two entailment relations occurring in the first square can be captured by considering additional integrity constraints (possibly added to the set $\mathcal{IC}_S$), relating positive and negative expectations as follows:

$$\mathbf{E}(A) \rightarrow \neg\mathbf{EN}(A)$$

and

$$\mathbf{EN}(A) \rightarrow \neg\mathbf{E}(A)$$

In practice, these two constraints, when added to $\mathcal{IC}_S$ and therefore considered in $\mathcal{IC}_S$-consistency, enforce the set of expectations to be "completed", i.e., for each positive expectation $\mathbf{E}(A)$ the explicit negation of its negative counterpart, $\neg\mathbf{EN}(A)$ had to be included in the expectation set (in order to get its admissibility), and for each negative expectation $\mathbf{EN}(A)$ the explicit negation of its positive counterpart, $\neg\mathbf{E}(A)$ had to be included as well.

Finally, a notion of *regimentation* can be considered too, by enforcing obligatory actions to happen and prohibited actions not to happen. This can be easily obtained by adding to the $\mathcal{IC}_S$ the following two integrity constraints, mapping positive/negative expectations into positive/negative events:

$$\mathbf{E}(A) \rightarrow \mathbf{H}(A)$$

and

$$\mathbf{EN}(A) \rightarrow \neg\mathbf{H}(A)$$

Notice that these two conditions correspond to the (meta) integrity constraints required for fulfillment of expectation sets (see Definition 5). The adopted notion of fulfillment in the declarative semantics, however, just test that these two constraints are not violated (by adopting the consistency view discussed in [15]), whereas if we add them to the set $\mathcal{IC}_S$ the $\mathcal{IC}_S$-consistency test (by adopting the theoremhood view, also discussed in [15]) would exploit them to also make events happening or not in the social environment.

A notable difference, from the representation point of view, is that in SOCS social integrity constraints can only express disjunctions of expectations, such that $\mathbf{E}(A) \vee \mathbf{E}(B)$ (which expresses that at least one of the two between A and B events is expected). In Deontic Logic, instead, one usually expresses the obligatoriness of disjunctions, i.e., $\mathbf{Obl}(A \vee B)$. In Kripke-like semantics (adopted for Deontic Logic), however, this is not equivalent to state $\mathbf{Obl}(A) \vee \mathbf{Obl}(B)$ [8].

The SOCS formalism based on $\mathcal{IC}_S$ constraints can capture, instead, in a computational setting, the concept of (conditional) obligation with deadline presented by Dignum *et al.* in [3], with an explicit mapping of time. Dignum *et al.*

[8] The two possible worlds $(A \wedge NONB)$ e $(NONA \wedge B)$ satisfy $\mathbf{Obl}(A \vee B)$, but not $\mathbf{Obl}(A) \vee \mathbf{Obl}(B)$.

write: `Oa(r<d|p)` to state that if the precondition `p` becomes valid, the obligation becomes active. The obligation expresses the fact that `a` is expected to bring about the truth of `r` before a certain condition `d` holds.

For instance, if we have:

$$\begin{aligned}&p = \mathbf{H}(tell(S, a, request(G), D, T))\\&r = \mathbf{H}(tell(a, S, answer(G), D, T')), T' > T\\&d = T' > T + 2\end{aligned}$$

we can map `Oa(r<d|p)` into a ic_S:

$$\begin{aligned}&\mathbf{H}(tell(S, a, request(G), D), T) \rightarrow\\&\quad \mathbf{E}(tell(a, S, answer(G), D), T'), T' > T, T' \leq T + 2.\end{aligned}$$

5 Related Work

There exist a number of approaches based on Deontic Logic to formally defining norms and dealing with their possible violations.

Among the organizational models, in [3–5] the authors exploit Deontic Logic to specify the society norms and rules. Their organizational model is based on a framework which consists of three interrelated models: organizational, social and interaction. The *organizational model* defines the coordination and normative elements and describes the expected behavior of the society. Its components are roles, constraints, interaction rules, and communicative and ontology framework. The *social model* specifies the contracts that make explicit the commitments regulating the enactment of roles by individual agents. Finally, the *interaction model* describes the possible interactions between agents by specifying contracts in terms of description of agreements, rules, conditions and sanctions.

SOCS society model, instead, follows an *institutional* approach and deals with the definition of agent interactions. This approach supposes that the situations where agents interact may involve commitments (a kind of obligations), delegation, repetition of interactions and risk. A suitable model to establish and enforce conventions is that of *institutions* [6, 7]. The basic aim of an institution is to facilitate, oversee and enforce commitments among agents.

Deontic operators have been used not only at the social level, but also at the agent level. Notably, in IMPACT [26, 27], agent programs may be used to specify what an agent is obliged to do, what an agent may do, and what an agent cannot do on the basis of deontic operators of Permission, Obligation and Prohibition (whose semantics does not rely on a Deontic Logic semantics). In this respect, the IMPACT and SOCS social models have similarities even if their purpose and expressivity are different. The main difference is that the goal of agent programs in IMPACT is to express and determine by its application the behavior of a single agent, whereas the SOCS social model goal is to express rules of interaction and norms, that instead cannot really determine and constrain the behavior of the single agents participating to a society, since agents are autonomous.

Another expressive advantage of our framework is that it can express both protocols and social semantics of communicative acts with the same formalism, as discussed in [28, 29].

6 Conclusion and Future Work

In this work, we have discussed how the Computational Logic-based framework for modeling societies of agents developed within the UE IST-2001-32530 project (named SOCS) can be exploited to express different variants of Deontic Logic. SOCS approach for modeling open societies is based on an abductive framework, where obligations and prohibitions are mapped into abducible predicates (respectively, positive and negative expectations), and norms ruling the behavior of members are represented as abductive integrity constraints. The SOCS social abductive framework can easily express different Deontic Logics, by means of additional (meta) integrity constraints.

This mapping is relevant from the representation point of view, but this is even more interesting from the computational viewpoint. In fact, since SOCS abductive social model is grounded on Computational Logic, it also offers an operational counterpart as an abductive proof procedure named $\mathcal{S}$CIFF which extends the IFF proof procedure [15]. $\mathcal{S}$CIFF is based on transitions able to deal with dynamic events, propagate social integrity constraints, etc., and it was proved sound with respect to the defined abductive declarative semantics. In particular, $\mathcal{S}$CIFF is able to verify the conformance of agent interactions with respect to the specified norms as $\mathcal{IC}_S$. Its implementation (see [30]) has been obtained in SICStus Prolog [31], by exploiting the *Constraint Handling Rules* (CHR) library [32]. Both $\mathcal{S}$CIFF transitions and the meta integrity constraints (for E- and $\neg$-consistency) have been mapped into CHR rewriting rules. This modular implementation can be easily extended by considering the additional integrity constraints defined in this paper, in order to deal with the different variants of Deontic Logic discussed. This is subject for future work.

Acknowledgments

This work has been supported by the European Commission within the SOCS project (IST-2001-32530), funded within the Global Computing Programme and by the MIUR COFIN 2003 projects *La Gestione e la negoziazione automatica dei diritti sulle opere dell'ingegno digitali: aspetti giuridici e informatici* and *Sviluppo e verifica di sistemi multiagente basati sulla logica.*

References

1. Conte, R., Falcone, R., Sartor, G.: Special issue on agents and norms. Artificial Intelligence and Law **1** (1999)

2. Castelfranchi, C., Dignum, F., Jonker, C., Treur, J.: Deliberative normative agents: Principles and architecture. In Jennings, N.R., Lespérance, Y., eds.: Intelligent Agents VI, Agent Theories, Architectures, and Languages, 6th International Workshop, ATAL '99, Orlando, Florida, USA, Proceedings. Number 1757 in Lecture Notes in Computer Science, Springer-Verlag (1999) 364–378
3. Dignum, V., Meyer, J.J., Dignum, F., Weigand, H.: Formal specification of interaction in agent societies. In: Proceedings of the Second Goddard Workshop on Formal Approaches to Agent-Based Systems (FAABS), Maryland. (2002)
4. Dignum, V., Meyer, J.J., Weigand, H., Dignum, F.: An organizational-oriented model for agent societies. In: Proceedings of International Workshop on Regulated Agent-Based Social Systems: Theories and Applications. AAMAS'02, Bologna. (2002)
5. Dignum, V., Meyer, J.J., Weigand, H.: Towards an organizational model for agent societies using contracts. In Castelfranchi, C., Lewis Johnson, W., eds.: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002), Part II, Bologna, Italy, ACM Press (2002) 694–695 `http://portal.acm.org/ft_gateway.cfm?id=544909&type=pdf&dl=GUIDE&dl=ACM%&CFID=4415868&CFTOKEN=57395936`.
6. Esteva, M., de la Cruz, D., Sierra, C.: ISLANDER: an electronic institutions editor. In Castelfranchi, C., Lewis Johnson, W., eds.: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002), Part III, Bologna, Italy, ACM Press (2002) 1045–1052 `http://portal.acm.org/ft_gateway.cfm?id=545069&type=pdf&dl=GUIDE&dl=ACM%&CFID=4415868&CFTOKEN=57395936`.
7. Noriega, P., Sierra, C.: Institutions in perspective: An extended abstract. In: Sixth International Workshop CIA-2002 on Cooperative Information Agents. Volume 2446 of Lecture Notes in Artificial Intelligence., Springer-Verlag (2002)
8. : ALFEBIITE: A Logical Framework for Ethical Behaviour between Infohabitants in the Information Trading Economy of the universal information ecosystem. IST-1999-10298 (1999) Home Page: `http://www.iis.ee.ic.ac.uk/~alfebiite/ab-home.htm`.
9. : (Societies Of ComputeeS (SOCS): a computational logic model for the description, analysis and verification of global and open societies of heterogeneous computees) `http://lia.deis.unibo.it/Research/SOCS/`.
10. Artikis, A., Pitt, J., Sergot, M.: Animated specifications of computational societies. In Castelfranchi, C., Lewis Johnson, W., eds.: Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS-2002), Part III, Bologna, Italy, ACM Press (2002) 1053–1061 `http://portal.acm.org/ft_gateway.cfm?id=545070&type=pdf&dl=GUIDE&dl=ACM%&CFID=4415868&CFTOKEN=57395936`.
11. Hewitt, C.: Open information systems semantics for distributed artificial intelligence. Artificial Intelligence **47** (1991) 79–106
12. Wright, G.: Deontic logic. Mind **60** (1951) 1–15
13. van der Torre, L.: Contextual deontic logic: Normative agents, violations and independence. Annals of Mathematics and Artificial Intelligence **37** (2003) 33–63
14. Alberti, M., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: An Abductive Interpretation for Open Societies. In Cappelli, A., Turini, F., eds.: AI*IA 2003: Advances in Artificial Intelligence, Proceedings of the 8th Congress of the Italian Association for Artificial Intelligence, Pisa. Volume 2829 of Lecture Notes in Artificial Intelligence., Springer-Verlag (2003) 287–299 `http://www-aiia2003.di.unipi.it`.

15. Fung, T.H., Kowalski, R.A.: The IFF proof procedure for abductive logic programming. Journal of Logic Programming **33** (1997) 151–165
16. Bracciali, A., Demetriou, N., Endriss, U., Kakas, A., Lu, W., Mancarella, P., Sadri, F., Stathis, K., Toni, F., Terreni, G.: The KGP model of agency: Computational model and prototype implementation. In: Global Computing Workshop, Rovereto, Italy, March 2004. Lecture Notes in Artificial Intelligence. Springer-Verlag (2004) to appear.
17. Lloyd, J.W.: Foundations of Logic Programming. 2nd extended edn. Springer-Verlag (1987)
18. Alberti, M., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: Specification and verification of interaction protocols: a computational logic approach based on abduction. Technical Report CS-2003-03, Dipartimento di Ingegneria di Ferrara, Ferrara, Italy (2003) Available at `http://www.ing.unife.it/aree_ricerca/informazione/cs/technical_reports`.
19. Kakas, A.C., Kowalski, R.A., Toni, F.: The role of abduction in logic programming. In Gabbay, D.M., Hogger, C.J., Robinson, J.A., eds.: Handbook of Logic in Artificial Intelligence and Logic Programming. Volume 5., Oxford University Press (1998) 235–324
20. Cox, P.T., Pietrzykowski, T.: Causes for events: Their computation and applications. In: Proceedings CADE-86. (1986) 608–621
21. Eshghi, K., Kowalski, R.A.: Abduction compared with negation by failure. In Levi, G., Martelli, M., eds.: Proceedings of the 6th International Conference on Logic Programming, MIT Press (1989) 234–255
22. Kakas, A.C., Mancarella, P.: On the relation between Truth Maintenance and Abduction. In Fukumura, T., ed.: Proceedings of the 1st Pacific Rim International Conference on Artificial Intelligence, PRICAI-90, Nagoya, Japan, Ohmsha Ltd. (1990) 438–443
23. Poole, D.L.: A logical framework for default reasoning. Artificial Intelligence **36** (1988) 27–47
24. Kunen, K.: Negation in logic programming. In: Journal of Logic Programming. Volume 4. (1987) 289–308
25. Sartor, G.: Legal Reasoning. Volume 5 of Treatise. Kluwer, Dordrecht (2004)
26. Arisha, K.A., Ozcan, F., Ross, R., Subrahmanian, V.S., Eiter, T., Kraus, S.: IMPACT: a Platform for Collaborating Agents. IEEE Intelligent Systems **14** (1999) 64–72
27. Eiter, T., Subrahmanian, V., Pick, G.: Heterogeneous active agents, I: Semantics. Artificial Intelligence **108** (1999) 179–255
28. Alberti, M., Ciampolini, A., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: Logic Based Semantics for an Agent Communication Language. In Dunin-Keplicz, B., Verbrugge, R., eds.: Proceedings of the International Workshop on Formal Approaches to Multi-Agent Systems (FAMAS), Warsaw, Poland (2003) 21–36
29. Alberti, M., Ciampolini, A., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: A social ACL semantics by deontic constraints. In Mařík, V., Müller, J., Pěchouček, M., eds.: Multi-Agent Systems and Applications III. Proceedings of the 3rd International Central and Eastern European Conference on Multi-Agent Systems, CEEMAS 2003. Volume 2691 of Lecture Notes in Artificial Intelligence., Prague, Czech Republic, Springer-Verlag (2003) 204–213
30. Alberti, M., Chesani, F., Gavanelli, M., Lamma, E., Mello, P., Torroni, P.: Compliance verification of agent interaction: a logic-based tool. In Trappl, R., ed.: Proceedings of the 17th European Meeting on Cybernetics and Systems Research,

Vol. II, Symposium “From Agent Theory to Agent Implementation” (AT2AI-4), Vienna, Austria, Austrian Society for Cybernetic Studies (2004) 570–575
31. : SICStus prolog user manual, release 3.11.0 (2003) `http://www.sics.se/isl/sicstus/`.
32. Frühwirth, T.: Theory and practice of constraint handling rules. Journal of Logic Programming **37** (1998) 95–138

Appendix

The SOKB is a logic program, consisting of clauses, possibly having expectations in their body. The full syntax of SOKB is the following:

$$\begin{aligned} \mathit{Clause} &::= \mathit{Atom} \leftarrow \mathit{Body} \\ \mathit{Body} &::= \mathit{ExtLiteral}\ [\ \wedge \mathit{ExtLiteral}\ ]^\star \\ \mathit{ExtLiteral} &::= \mathit{Literal} \mid \mathit{Expectation} \mid \mathit{Constraint} \\ \mathit{Expectation} &::= [\neg]\mathbf{E}(\mathit{Event}\ [,T]) \mid [\neg]\mathbf{EN}(\mathit{Event}\ [,T]) \end{aligned} \tag{19}$$

Social Integrity Constraints are in the form of implications. The characterizing part of their syntax is the following:

$$\begin{aligned} ic_S &::= \chi \rightarrow \phi \\ \chi &::= (HEvent|\mathit{Expectation})\ [\wedge \mathit{BodyLiteral}]^\star \\ BodyLiteral &::= HEvent|\mathit{Expectation}|\mathit{Literal}|\mathit{Constraint} \\ \phi &::= HeadDisjunct\ [\ \vee HeadDisjunct\ ]^\star|\bot \\ HeadDisjunct &::= \mathit{Expectation}\ [\ \wedge (\mathit{Expectation}|\mathit{Constraint})]^\star \\ \mathit{Expectation} &::= [\neg]\mathbf{E}(\mathit{Event}\ [,T]) \mid [\neg]\mathbf{EN}(\mathit{Event}\ [,T]) \\ HEvent &::= [\neg]\mathbf{H}(\mathit{Event}\ [,T]) \end{aligned} \tag{20}$$

Given an ic_S $\chi \rightarrow \phi$, χ is called the *body* (or the *condition*) and ϕ is called the *head* (or the *conclusion*).

For the scope rules and quantifications, please refer to [18].