R. Wieringa, F. Dignum, J.-J.Ch. Meyer and R. Kuiper. A modal approach to intentions, commitments and obligations: Intention plus commitment yields obligation. In M.A. Brown and J. Carmo, editors, *Deontic Logic, Agency and Normative Systems (Workshops in Computing)*, pages 80--97. Springer-Verlag, 1996.

A Modal Approach to Intentions, Commitments and Obligations:

Intention plus Commitment yields Obligation.

F.Dignum [*] J.-J.Ch.Meyer [†] R.J.Wieringa [‡] R. Kuiper [§]

Content areas: formal systems of deontic logic and actions, formal specification of normative systems.

[*] Eindhoven University of Technology, Dept. of Mathematics and Computer Science, P.O.box 513, 5600 MB Eindhoven, The Netherlands, tel.+31-40-473705, fax. +31-40-463992, e-mail: dignum@win.tue.nl

[†] Utrecht University, Dept. of Computer Science, P.O.box 80085, 3508 TB Utrecht, The Netherlands, e-mail: jj@cs.ruu.nl

[‡] Free University of Amsterdam, Faculty of Mathematics and Computer Science, De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands, e-mail:roelw@cs.vu.nl The research of J.-J.Ch.Meyer and R.J.Wieringa is partially supported by ESPRIT BRA 8319 ModelAge.

[§] Eindhoven University of Technology, Dept. of Mathematics and Computer Science, P.O.box 513, 5600 MB Eindhoven, The Netherlands, tel.+31-40-474122, fax. +31-40-463992, e-mail: wsinruur@win.tue.nl

A Modal Approach to Intentions, Commitments and Obligations: Intention plus Commitment yields Obligation.

EXTENDED ABSTRACT

Content areas: formal systems of deontic logic and actions, formal specification of normative systems.

Abstract

In this paper we introduce some new operators that make it possible to reason about decisions and commitments to do actions. In our framework, a decision leads to an intention to do an action. The decision in itself does not change the state of the world; a commitment to actually perform the intended action changes the deontic state of the world such that the intended action becomes obligated. Of course, the obligated action may never actually occur. In our semantic structure, we use static (ough-to-be) and dynamic (ought-to-do) obligation operators. The static operator resembles the classical conception of obligation as truth in ideal worlds, except that it takes the current state as well as the past history of the world into account. This is necessary because it allows us to compare the way a state is actually reached with the way we committed ourselves to reach it. We show that some situations that could formerly not be expressed easily in deontic logic can be described in a natural way using the extended logic described in this paper.

1 Introduction

In the fields of law as well of computer science there are frequently occurring situations in which intentions and obligations to perform actions arise depending on past events. In law, the main purpose of a court trial is to discover what was the *actual* sequence of events that led to a situation in which a crime had been committed. Usually there are many ways that the situation could have been reached. The task of the judge (and the jury) is to find out what the actual way was. On the basis of the findings a judgement is formed, which leads to an intended (i.e. preferred) action for the future. The intended action is either some punishment or a dismissal of the defendant. Sometimes the decision can still change a few times either through a ruling of the judge or through a ruling by a higher court. When a final judgement has been given and thus a commitment has been made there arises an obligation (for the public prosecutor) to fulfil the judgement.

In database systems the *transaction* concept is used to maintain the integrity of the database [Date95]. A transaction is a conceptually atomic state transition that, at any moment, either has occurred or has not yet occurred. A transaction is implemented as a process that is not atomic but guarantees atomicity by a "rollback/commit" protocol. (The database "commit" is a different concept from the commitment studied in this paper.) Before a transaction attempt is "committed", it can be rolled back to restore the state as if no transaction has occurred. After the transaction is "committed", it cannot be rolled back; however, a *compensating transaction* can be performed to undo some of the effects of the transaction. For example, a money transfer is a transaction, and it can be rolled back if it is not yet committed. The history of the bank account will then not contain this money transfer. If after the transaction is committed, it turns out that it should be undone, then the obligation arises to inform the account owner of this and to perform a compensating transfer, which returns the money (and lost interest). In this case, however, the account has both transactions in its history.

The purpose of this paper is to show how these examples can be modelled in a simple and elegant way, using deontic logic with actions, intentions and (a very limited set of) temporal operators. In the next section we will introduce the logic that we will use. In section 3 we will extend the logic with two types of meta-actions that express decisions and commitments. In section 4 we will show how the above examples can be modelled in our logic. Section 5 is used to draw some conclusions.

2 A logic of actions and norms

We now proceed with the definition of a set of *formulas* with which we can describe the behaviour of (interpreted) actions. This language is a variant of *dynamic logic* ([Har79]), and was first used for this purpose in [Mey88]. In the present paper we add a few "new" formulas to this language. They are the ones defined in points (5) and (6) below. The formulas defined in (5) all involve some type of temporal operations on actions. The formulas defined in (6) define the "classical" deontic formulas as introduced by v.Wright in [Wri51, Aqv84].

We assume a fixed set *Prop* of atomic propositions and a set *Act* of action expressions. The set *Form* of formulas is then the smallest set closed under:

(1). $Prop \subseteq Form$

(2). $\phi_1, \phi_2 \in Form \Longrightarrow \phi_1 \wedge \phi_2 \in Form$

(3). $\phi \in Form \Longrightarrow \neg\phi \in Form$

(4). $\alpha \in Act, \phi \in Form \Longrightarrow [\alpha]\phi \in Form$

(5). $\alpha \in Act \Longrightarrow PAST(\alpha), PREV(\alpha), INT(\alpha) \in Form$

(6). $\phi \in Form \Longrightarrow O(\phi) \in Form$

Note: Other propositional connectives such as $\vee$ and $\rightarrow$ are assumed to be introduced as the usual abbreviations. Also the special proposition $false$ is introduced as the abbreviation of $p \wedge \neg p$ for some $p \in Prop$. The informal meaning of $[\alpha]\phi$ is "doing α necessarily leads to a state where ϕ obtains". The meaning of $PAST(\alpha)$ is that α has actually been performed in the past. The meaning of $PREV(\alpha)$ is "the present state is actually reached by performing α". Note that we make a difference between the possible ways that the state can be reached and the way it actually *is* reached. In our semantics, we will therefore consider state-history pairs. The formula $INT(\alpha)$ intuitively means that we prefer (or intend) to perform the action expressed by α next. The informal meaning of $O(\phi)$ is that ϕ should be the case in the present state. Because we include histories in our semantic structure, we can express that the current state should, ideally, have been reached by another history. We introduce the other deontic operators using the usual abbreviations:

- $F(\phi)$ abbreviates $O(\neg\phi)$
- $P(\phi)$ abbreviates $\neg F(\phi)$

The semantics of the formulas in *Form* is given in two stages. First we will give the syntax and semantics of the action expressions, which we will use in section 2.2 to define the semantics of the formulas.

2.1 Action expressions and their semantics

First we give a definition of *action expressions*, which we shall typically denote α, possibly with subscripts. To this end we assume a set At of *atomic action expressions* that are typically denoted by $\underline{a}, \underline{b}, \ldots$. Finally, we assume special action expressions $\mathbf{any}$ and $\mathbf{fail}$ denoting "don't care what happens" and "failure", respectively.

Definition 1 The set Act of action expressions is given as the smallest set closed under:

(i). $At \cup \{\mathbf{any}, \mathbf{fail}\} \subseteq Act$

(ii). $\alpha_1, \alpha_2 \in Act \Longrightarrow \alpha_1 + \alpha_2 \in Act$

(iii). $\alpha_1, \alpha_2 \in Act \Longrightarrow \alpha_1 \& \alpha_2 \in Act$

(iv). $\alpha \in Act \Longrightarrow \overline{\alpha} \in Act$

We further use the following terminology: atomic action expressions $\underline{a} \in Act$, as well as their negations $\overline{\underline{a}}$ ($\underline{a} \in At$) are called *action literals*. An action expression that is the conjunction (using the &-connective) of one or more action literals is called an *action term*. So, for example, $\underline{a} \& \overline{\underline{b}} \& \underline{c}$, for $\underline{a}, \underline{b}, \underline{c} \in At$, is an action term.

For simplicity, we do *not* have a notion of sequences of actions in the action expressions in this paper. We discuss the action sequence operator elsewhere [DM90, Dig92].

The semantics of action expressions is given in two stages. First we define an algebra of uninterpreted actions (called a *uniform* semantics elsewhere [dBKM+86]), which allow us to interpret equalities between between action expressions without taking their effect into account. In the algebraic semantics, each action expression will be interpreted as a choice over possible steps. Next, we give a state-transition semantics of action expressions in which we define the effect of steps on the state of the world.

Algebraic action semantics. With every atomic action expression $\underline{a} \in Act$, we associate an event a in a given class $\mathcal{A}$ of events, with typical elements a, b, c.... Events are the semantical entities on which we shall base our interpretation of action expressions. We further assume a special event δ, which is not an element of $\mathcal{A}$, called failure (comparable to deadlock in process algebra ([BW90]). The relation between an action expression $\underline{a} \in Act$ and the associated event $a \in \mathcal{A}$ is more involved than just interpreting $\underline{a}$ as a. We shall interpret atomic action expressions $\underline{a} \in Act$ in a more sophisticated way, which we call "open": the meaning of an atomic action expression $\underline{a} \in Act$ will be the event $a \in \mathcal{A}$ corresponding with it, in combination with any other subset of the events in $\mathcal{A}$. Thus $\underline{a}$ expresses that a occurs, but it leaves open which other events occur simultaneously (in the same step) with a. The intuitive motivation for this is that if we say that an event a occurs, we do not means that nothing else occurs in the world.

Definition 2

1. The set $\{\delta\}$ is a step.

2. Every non-empty finite subset of $\mathcal{A}$ is a step. The powerset of non-empty finite subsets of $\mathcal{A}$ will be denoted by $\mathcal{F}^+(\mathcal{A})$.

3. The domain $\mathcal{D}$ for our model for action expressions from *Act* is the collection of sets of steps, $\mathcal{D}$=$\wp(\wp^+(\mathcal{A}) \cup \{[\delta]\})$. An element of $\mathcal{D}$ is called a *choice set*.

The above definition prevents the simultaneous execution of the special event δ with other events, because it is not in $\mathcal{A}$. This is necessary, because it is not possible to perform an event and at the same time have a deadlock. Below, we interpret action expressions as choice sets.

Definition 3 Let T be a set of steps then

$$T^\delta = \begin{cases} T \backslash \{[\delta]\} & \text{if } \exists S \in T : S \neq [\delta] \\ \{[\delta]\} & \text{otherwise} \end{cases}$$

The operator T^δ is closely related to what is called "failure removal" in [dBKM+86]. The idea is that failure is avoided when possible, i.e. when there is a non-failing alternative. In [Bro86], this is called *angelic* nondeterminism. We can define the semantic operators on $\mathcal{D}$. For the parallel operator $\&$ we use a set-intersection $\owedge$, which is almost the same as the normal set-intersection.

Definition 4 For $T, T' \in \mathcal{D}$:

$$T \owedge T' = \begin{cases} T \cap T' & \text{if } T \cap T' \neq \emptyset \\ \{[\delta]\} & \text{otherwise} \end{cases}$$

The semantical counterpart of the choice operator is defined as follows:

Definition 5 For $T, T' \in \mathcal{D}$:

$$T \owedge T' = (T \cup T')^\delta$$

The above definition states that the choice between two sets of steps is the union of those two sets minus $[\delta]$, unless the union does not contain anything else. Finally, we define the semantic counterpart of the negation operator.

Definition 6 The definition of "$\sim$" is given as follows:

1. For a step S,

$$S^{\sim} = \wp^{+}(\mathcal{A}) \backslash \{S\}$$

2. For a non-empty set $T \in \mathcal{D}$

$$T^{\sim} = \mathbin{⫛}_{S \in T} S^{\sim}$$

That is, for a step ($S \neq [\delta]$) the negation just yields the set-theoretic complement of $\{S\}$ with respect to $\wp^{+}(\mathcal{A})$. The negation of a set of steps T is the set-theoretic complement of T with respect to $\wp(\wp^{+}(\mathcal{A}))$ if $T \neq \wp^{+}(\mathcal{A})$. Otherwise it is $\{[\delta]\}$. We can now give the algebraic semantics of action expressions:

Definition 7 The semantic function $[\![]\!] \in Act \rightarrow \mathcal{D}$ is given by:

$$[\![\underline{a}]\!] = \{S \in \wp^{+}(\mathcal{A}) | a \in S\}$$
$$[\![\alpha_1 + \alpha_2]\!] = [\![\alpha_1]\!] \mathbin{⫏} [\![\alpha_2]\!]$$
$$[\![\alpha_1 \& \alpha_2]\!] = [\![\alpha_1]\!] \mathbin{⫛} [\![\alpha_2]\!]$$
$$[\![\overline{\alpha}]\!] = [\![\alpha]\!]^{\sim}$$
$$[\![\mathbf{fail}]\!] = \{[\delta]\}$$
$$[\![\mathbf{any}]\!] = \wp^{+}(\mathcal{A})$$

The first clause of the above definition expresses that the meaning of the action expression $\underline{a}$ is exactly as we have described informally before: it is the set of steps that contain the event a, representing a choice between all (simultaneous) performances of sets of events which at least contain the event a, so that the performance of a is guaranteed but also other events may happen simultaneously. The meaning of the action expression **fail** is comparable to a deadlock. The only event that can be performed is δ. The action expression **any** is the complement of **fail**. It stands for a choice of any possible combination of events.

The following properties can be easily proven and will be used later on:

Proposition 1

1. $[\![\alpha]\!] \cap [\![\overline{\alpha}]\!] = \emptyset$
2. $[\![\overline{\overline{\alpha}}]\!] = [\![\alpha]\!]$
3. $[\![\overline{\mathbf{fail}}]\!] = [\![\mathbf{any}]\!]$

Finally we define equality and implication between action expressions.

Definition 8 Action expressions α_1 and α_2 are *equal*, written $\alpha_1 =_{\mathcal{D}} \alpha_2$, iff [[$\alpha_1$]]= [[$\alpha_2$]]. α_1 *involves* or *implies* α_2, written $\alpha_1 > \alpha_2$, iff [[α_1]]$\subseteq$ [[α_2]].

State-transition action semantics. To get a state-transition semantics, we postulate what effects events have in terms of state transformations (we do this relative to a set Σ of states). We assume that there is a function $eff_\Sigma : \mathcal{A} \to (\Sigma \to \Sigma)$, such that $eff_\Sigma(a)$ is a function from states to states. (For simplicity, we assume events to be deterministic. Elsewhere, we show how nondeterministic events can be incorporated [Mey88].) Two actions are called *compatible* if their joint effect is independent from the order in which they occur.

Definition 9 Let $S = [a_1, ..., a_n] \subseteq \mathcal{A}$ be a step consisting of pairwise compatible events. The accessibility relation $R_S \subseteq \Sigma \times \Sigma$ is defined as follows:

$$R_S(\sigma, \sigma') \Longleftrightarrow_{def} (eff_\Sigma(a_1) \circ ... \circ eff_\Sigma(a_n))(\sigma) = \sigma'$$

We require that in every state, we can do at least one step: $\forall \sigma \in \Sigma \exists S \subseteq \mathcal{A} \exists \sigma' \in \Sigma : R_S(\sigma, \sigma')$

2.2 Semantics of formulas

Having defined the semantics of the action expressions within the formulas, we can now give the semantics of formulas in *Form* by means of the notion of a Kripke structure $\mathcal{M} = (\Sigma, \mathcal{A}, \pi, R_\mathcal{A}, R_O, I)$.
Σ is a set of states (worlds).
$\mathcal{A}$ is a finite set of events.
π is a truth assignment function to the atomic propositions relative to a state: π is a function $\Sigma \to (Prop \to \{tt, ff\})$, where tt and ff denote truth and falsehood, respectively. Thus, for $p \in Prop$, $\pi(\sigma)(p) = tt$ means that the atomic proposition p is true in state σ.

The accessibility relation $R_\mathcal{A}$ specifies how actions can change states. The relation $R_\mathcal{A}$ is defined as follows: $R_\mathcal{A} = \{R_S | S \subseteq \mathcal{A}\}$. Thus we have

$$R_\mathcal{A}(\sigma, \sigma') \Longleftrightarrow \exists S \subseteq \mathcal{A} : R_S(\sigma, \sigma')$$

The relation R_O relates pairs of states σ and histories γ. The history of a state is expressed by a *trace* of actions. In process algebra, temporal logic and semantics of parallel programs traces are widely used to record what actions have taken place. Traces are defined inductively as follows:

Definition 10 1. ϵ is a (empty) trace.

2. if γ is a trace and $S \subseteq \mathcal{A}$ then $\gamma \circ S$ is a trace.

Furthermore $\gamma \circ \epsilon = \epsilon \circ \gamma = \gamma$. We use $\mathcal{A}^*$ to denote the set of all traces.

We would like to ensure that it is possible for a state to be reached by its history. Therefore we introduce the following relation between states and traces relative to a model $\mathcal{M}$:

Definition 11 For each model $\mathcal{M} = (\mathcal{A}, \Sigma, \pi, R_{\mathcal{A}}, R_O, I)$ and pair (σ, γ) with $\sigma \in \Sigma$ and $\gamma \in \mathcal{A}^*$ $(\sigma, \gamma) \in Comp(\mathcal{M})$ iff

1. $\gamma = \epsilon$ or

2. $\gamma = \gamma' \circ S \wedge \exists \sigma' \in \Sigma[R_S(\sigma', \sigma) \wedge (\sigma', \gamma') \in Comp(\mathcal{M})]$

From now on we assume that all pairs $(\sigma, \gamma) \in Comp(\mathcal{M})$ unless it is stated otherwise.

R_O is the deontic relation that with respect to a state σ reached by history γ, the ideal situation is state σ' reached by history γ'. R_O resembles the classical deontic relation in modal interpretations, except that we do not consider only states, but pairs of states and traces. We assume the relation R_O to be serial. I.e. for every world σ and trace γ there exists at least one pair (σ', γ') such that $R_O((\sigma, \gamma)(\sigma', \gamma'))$ holds.

The last type of accessibility relation I specifies which of the actions that are possible in a certain state are "intended" to be performed. I.e. it indicates the actions that are not only possible but also intended to be done. Like $R_{\mathcal{A}}$ it is defined as $I = \{I_S | S \subseteq \mathcal{A}\}$. Thus we have

$$I(\sigma, \sigma') \Longleftrightarrow \exists S \subseteq \mathcal{A} : I_S(\sigma, \sigma')$$

Because we intend this relation to be a subset of the accessibility relation with actions, we require that $\forall S \subseteq \mathcal{A}, \sigma, \sigma' \in \Sigma : I_S(\sigma, \sigma') \Longrightarrow R_S(\sigma, \sigma')$. The relation I can be given as primitive, but can also be given a deontic interpretation by relating it to deontically preferable actions. For now we will assume this relation to be primitive. In section 5 we will show how it can be related to a deontic ordering on states as defined in [DMW94].

We now give the interpretation of formulas in *Form* in Kripke structures. We interpret formulas with respect to a structure $\mathcal{M}$ and a pair $(\sigma, \gamma) \in Comp(\mathcal{M})$

Definition 12 Given $\mathcal{M} = (\mathcal{A}, \Sigma, \pi, R_{\mathcal{A}}, R_O, I)$ as above and $(\sigma, \gamma) \in Comp(\mathcal{M})$, we define:

1. $(\mathcal{M},(\sigma,\gamma)) \models p \iff \pi(\sigma)(p) = tt$ (for $p \in Prop$)

2. $(\mathcal{M},(\sigma,\gamma)) \models \phi_1 \wedge \phi_2 \iff (\mathcal{M},(\sigma,\gamma)) \models \phi_1$ and $(\mathcal{M},(\sigma,\gamma)) \models \phi_2$

3. $(\mathcal{M},(\sigma,\gamma)) \models \neg\phi \iff$ not $(\mathcal{M},(\sigma,\gamma)) \models \phi$

4. $(\mathcal{M},(\sigma,\gamma)) \models [\alpha]\phi \iff \forall S \in [\![\alpha]\!] \forall \sigma' \in \Sigma[R_S(\sigma,\sigma') \Rightarrow (\mathcal{M},(\sigma',\gamma \circ S)) \models \phi]$

5. $(\mathcal{M},(\sigma,\gamma)) \models O(\phi) \iff \forall (\sigma',\gamma') \in Comp(\mathcal{M})[R_O((\sigma,\gamma),(\sigma',\gamma')) \Rightarrow (\mathcal{M},(\sigma',\gamma')) \models \phi]$

6. $(\mathcal{M},(\sigma,\gamma)) \models PREV(\alpha) \iff \exists S \in [\![\alpha]\!], \gamma' \in \mathcal{A}^*[\gamma = \gamma' \circ S]$

7. $(\mathcal{M},(\sigma,\gamma)) \models PAST(\alpha) \iff \exists \sigma' \in \Sigma \exists \gamma', \gamma'' \in \mathcal{A}^*[\gamma = \gamma' \circ \gamma'' \wedge (\mathcal{M},(\sigma',\gamma')) \models PREV(\alpha)]$

8. $(\mathcal{M},(\sigma,\gamma)) \models INT(\alpha) \iff \exists S \in [\![\alpha]\!], \sigma' \in \Sigma[I_S(\sigma,\sigma')]$

9. ϕ is *valid* w.r.t. model $\mathcal{M} = (\mathcal{A}, \Sigma, \pi, R_{\mathcal{A}}, R_O, I)$, notation $\mathcal{M} \models \phi$, if $(\mathcal{M},(\sigma,\gamma)) \models \phi$ for all $\sigma \in \Sigma$ and $\gamma \in \mathcal{A}^*$.

10. ϕ is *valid*, notation $\models \phi$, if ϕ is valid w.r.t. all models $\mathcal{M}$ of the form considered above.

The first four definitions are quite standard and we will not explain them any further at this place. The definition of the static obligation involves both the state and the trace (and *not* just the state). In this way, we can express that a formula like $PREV(\alpha)$ is obligated. For example, it might be obligated to have just done the action indicated by α. This means that in an ideal world the history (i.e. the trace) might differ from the history of the present world. We will use this feature to define obligations on actions shortly.

It should be noted that using the semantic definition of $[\mathbf{any}]\phi$ we can express the usual temporal operators over static formulas as given in e.g. [Eme89]. Points (6), (7) and (8) define extra temporal operators reaching over action expressions! (6) and (7) are quite obvious. The definition of the semantics of INT indicates that an action α is intended to be performed (or preferred) whenever there is *some* way to perform α. Intuitively this seems evident. For example, if I intend to fly to Barcelona, then I intend to go to Barcelona. Conversely, if I intend to go to Barcelona, I do not intend to do this in every possible way (e.g. by walking) but just that there is one way in which I intend to do this.

Using the above definitions we can now introduce the deontic operators over actions as follows:

- $O(\alpha) \equiv [\mathbf{any}]O(PREV(\alpha))$
- $F(\alpha) \equiv O(\overline{\alpha})$
- $P(\alpha) \equiv \neg F(\alpha)$

So, α is obligated if, whatever I do, I should have done the action expressed by α afterwards. This means that if I do $\overline{\alpha}$, I reach a state where a violation occurs, indicated by the fact that in that state both $O(PREV(\alpha))$ and $PREV(\overline{\alpha})$ hold true.
Note that by definition the following formula is valid for all actions $\alpha \in Act$:

$$[\alpha]PREV(\alpha)$$

Therefore

$$O(\alpha) \Longrightarrow [\overline{\alpha}](PREV(\overline{\alpha}) \wedge O(PREV(\alpha)))$$

In earlier accounts this implication was used to define the obligation as:

$$O(\alpha) \equiv [\overline{\alpha}]Violation$$

using a kind of Anderson's reduction. For an extensive account of the adequacy of these abbreviations we refer to [Mey88, Mey87, WWMD91]; here it suffices to note that e.g. the claim that it is forbidden to do the action denoted by α is equated with the fact that the resulting state is not ideal (i.e. is in violation), because $PREV(\alpha)$ holds in that state while $PREV(\overline{\alpha})$ holds in the ideal states reachable from that state.

An important difference with this earlier account is that now we do *not* have:

$$O(\alpha) \Longleftarrow [\overline{\alpha}](PREV(\overline{\alpha}) \wedge O(PREV(\alpha)))$$

The consequence of this difference is that whereas in the earlier account we did not have the D-axiom

$$\models \neg(O(\alpha) \wedge O(\overline{\alpha}))$$

for obligations (automatically), we do inherit this property now from the static deontic operator.

The following validities follow (easily) from the above definition of the semantics of formulas in *Form*:

Proposition 2

1. $\models [\alpha](\phi \rightarrow \psi) \rightarrow ([\alpha]\phi \rightarrow [\alpha]\psi)$
2. $\models \phi \Longrightarrow \models [\alpha]\phi$
3. $\models \neg[\mathbf{any}]false$
4. $\models O(\phi \rightarrow \psi) \rightarrow (O(\phi) \rightarrow O(\psi))$
5. $\models \phi \Longrightarrow \models O(\phi)$
6. $\models \neg O(false)$
7. $\models \neg O(\mathbf{fail})$
8. $\models O(\mathbf{any})$
9. $\models O(\overline{\alpha}) \rightarrow \neg O(\alpha)$
10. $\models O(\alpha_1) \vee O(\alpha_2) \rightarrow O(\alpha_1 + \alpha_2)$
11. $\models (O(\alpha_1) \wedge O(\alpha_2)) \leftrightarrow O(\alpha_1 \& \alpha_2)$
12. $\models [\alpha]PREV(\alpha)$
13. $\models PREV(\overline{\alpha}) \leftrightarrow \neg PREV(\alpha)$
14. $\models PREV(\alpha_1) \vee PREV(\alpha_2) \leftrightarrow PREV(\alpha_1 + \alpha_2)$
15. $\models (PREV(\alpha_1) \wedge PREV(\alpha_2)) \leftrightarrow PREV(\alpha_1 \& \alpha_2)$
16. $\models \neg PREV(\mathbf{fail})$
17. $\models PREV(\mathbf{any})$
18. $\models PREV(\alpha) \rightarrow PREV(\beta)$ *if* $\alpha > \beta$
19. $\models PAST(\alpha) \rightarrow PAST(\beta)$ *if* $\alpha > \beta$
20. $\models PREV(\alpha) \rightarrow PAST(\alpha)$

21. $\models [\alpha]\phi \Longleftrightarrow \models PREV(\alpha) \rightarrow \phi$

22. $\models INT(\alpha) \rightarrow INT(\beta)$ *if* $\alpha > \beta$

23. $\models INT(\alpha_1 \& \alpha_2) \rightarrow INT(\alpha_1) \wedge INT(\alpha_2)$

24. $\models INT(\alpha_1 + \alpha_2) \leftrightarrow (INT(\alpha_1) \vee INT(\alpha_2))$

25. $\models \neg INT(\mathbf{fail})$

Note that we do *not* have $\models INT(\alpha_1 \& \alpha_2) \leftarrow INT(\alpha_1) \wedge INT(\alpha_2)$. For example, if I intend to drink a beer and have a brandy, I do not intend to gulp them down simultaneously. Note that this implies

$$\not\models INT(\alpha) \wedge INT(\overline{\alpha}) \rightarrow INT(\alpha \& \overline{\alpha}).$$

3 Decisions and Commitments

In the previous section we introduced a deontic logic that allows for the expression of both static as well as dynamic deontic formulas. It also contains several "temporal" operators that make it possible to state that an action has actually been performed or is intended to be performed next.

In the present section we will extend the logic with two types of meta-actions, the $DECIDE$ and the $COMMIT$ actions. The $DECIDE$ actions establish which actions are intended to be performed in a certain state. Thus, it is possible to express in the logic, the decision which action is intended to be performed next. It may be clear that these type of actions have to be distinguished from the other actions, because they change the accessibility relations for intentions and obligations. We first define the syntax of the meta-actions:

Definition 13

1. $\alpha \in Act \Longrightarrow DECIDE(\alpha) \in MAct$ and $COMMIT(\alpha) \in MAct$

2. $\mu \in MAct, \phi \in Form \Longrightarrow [\mu]\phi \in Form$

Next we define a function on the semantic domain that will serve as the semantics of the $DECIDE$ functions. On the basis of a state σ and an action expression α it changes the relation I such that from

state σ only the actions that can be done in parallel with the action expressed by α are intended to be performed.

Definition 14

$$\begin{aligned} dec(\alpha, \sigma, I) = I' \Longleftrightarrow \quad & \forall \sigma' \neq \sigma [I(\sigma', \sigma'') \Longleftrightarrow I'(\sigma', \sigma'')] \wedge \\ & \forall S \in [\![\alpha]\!] \exists \sigma' I'_S(\sigma, \sigma') \wedge \\ & \forall S \notin [\![\alpha]\!] \neg \exists \sigma' I'_S(\sigma, \sigma') \end{aligned}$$

So, the relation I does not change for any state except σ. In σ it is changed in a way that the action expressed by α is intended to be performed, while the action expressed by $\overline{\alpha}$ cannot be intended to be performed. The decision to (intend) to perform the action expressed by α does not fix the way the action will be performed. Because the function dec establishes a relation I_S for all $S \in [\![\alpha]\!]$ from the state σ to some state σ' means that every possible way to perform the action expresses by α is intended. This conforms to the idea that I may e.g. decide to go on holiday next week, leaving open where I will go to, how I will go there, etc. But if I intend to go on holiday next week I cannot, of course, also intend to stay at home and work next week. The semantics of $DECIDE$ is now defined as follows:

Definition 15

$$((\mathcal{A}, \Sigma, \pi, R_{\mathcal{A}}, R_O, I), (\sigma, \gamma)) \models [DECIDE(\alpha)]\phi \Longleftrightarrow ((\mathcal{A}, \Sigma, \pi, R_{\mathcal{A}}, R_O, dec(\alpha, \sigma, I)), (\sigma, \gamma)) \models \phi$$

The fact that a certain action α is intended does not have any influence on the course of events. To influence (in a deontic way) the course of events, α must be *committed* to. The act of commitment has been discussed in the literature at various places already. Notably, in the theory of speech acts it is one of the basic communication types (see [Sea69, SV85]). In [LHM95] the commitment is related to a goal of an autonomous agent. In this paper we give the commitment a deontic connotation by defining the commitment to perform α as a (meta-)action with the result that it is obligated to perform α. Moreover we restrict the commitment to only those actions that are already intended. The following definition gives a function that defines the exact changes of the model brought about by a commitment. It will serve as the semantics of the $COMMIT$ actions.

Definition 16

$com(\alpha, (\sigma, \gamma), I, R_O) = R'_O \Longleftrightarrow$

(1) $[\exists S \in [\![\alpha]\!], \sigma' : I_S(\sigma, \sigma')] \Longrightarrow$

(1a) $[\forall\sigma'\forall S \subseteq \mathcal{A} : [R_S(\sigma,\sigma') \Longrightarrow$
$\forall(\sigma'',\gamma'')(R'_O((\sigma',\gamma \circ S),(\sigma'',\gamma'')) \Longleftrightarrow (R_O((\sigma',\gamma \circ S),(\sigma'',\gamma'')) \wedge \exists T \in [[\alpha]]\exists\gamma' \in \mathcal{A}^* :$
$\gamma'' = \gamma' \circ T)]\wedge$

(1b) $\forall\sigma'[(\neg\exists S \subseteq \mathcal{A} : R_S(\sigma,\sigma') \Longrightarrow \forall(\sigma'',\gamma'')R'_O((\sigma',\gamma'),(\sigma''\gamma'')) = R_O((\sigma'\gamma'),(\sigma'',\gamma''))]]$
$\wedge$

(2) $[\neg\exists S \in [[\alpha]], \sigma' : I_S(\sigma,\sigma')] \Longrightarrow$
$\forall(\sigma',\gamma')(\sigma'',\gamma'')R'_O((\sigma',\gamma'),(\sigma'',\gamma'')) = R_O((\sigma',\gamma'),(\sigma'',\gamma''))$

Part (2) states that the relation R_O does not change if α does not denote an intended action. I.e. the commitment only has effect if the action that is committed to is already intended.

Part (1) of the definition covers the case that α denotes an intended action. Even if α is intended only a limited part of the relation R_O changes. In part (1b) it is stated that the relation R_O does not change for any state σ' that is not reachable (by performing some action) from σ.

Part (1a) states the actual changes of the relation R_O. For all pairs of states and traces $(\sigma', \gamma \circ S)$ that can be reached from (σ,γ) (by performing any possible step S) it holds that for all their ideal alternatives the last step of their trace is a step from α (i.e. in σ' it is obligated that $PREV(\alpha)$) and also their ideal alternatives were already ideal according to R_O before taking this step. Together this states so much as that the action denoted by α has become obligated in σ and all previous obligations are still in effect. If $COMMIT(\alpha)$ would bring us into a structure in which some state has no R'_O-successor, then $COMMIT(\alpha)$ is equivalent to **fail**.

The semantics of $COMMIT$ is now given as follows:

Definition 17

$$((\mathcal{A},\Sigma,\pi,R_{\mathcal{A}},R_O,I),(\sigma,\gamma)) \models [COMMIT(\alpha)]\phi \wedge COMMIT(\alpha) \neq_{\mathcal{D}} \textbf{fail} \Longleftrightarrow$$
$$((\mathcal{A},\Sigma,\pi,R_{\mathcal{A}},com(\alpha,\sigma,R_O),I)),(\sigma,\gamma)) \models \phi$$

Note that we cannot use the semantics of the formulas of the form $[\alpha]\phi$ with $\alpha \in Act$ to denote the semantics of $[\beta]\phi$ with $\beta \in MAct$ because when α is in Act the formula $[\alpha]\phi$ and the formula ϕ are evaluated in the same model, while if α is in $MAct$ the formulas $[\alpha]\phi$ and ϕ are evaluated in different models!

The following validities follow directly from the above definitions:

Proposition 3

1. $\models [DECIDE(\alpha)]INT(\alpha)$
2. $\models [DECIDE(\alpha)]\neg INT(\beta)$ *if* $\alpha \& \beta =_{\mathcal{D}}$ **fail**
3. $\models INT(\alpha) \longrightarrow [COMMIT(\alpha)]O(\alpha)$
4. $\models [DECIDE(\alpha)][COMMIT(\alpha)]O(\alpha)$
5. $\models (\neg O(\alpha) \wedge \neg INT(\alpha)) \longrightarrow [COMMIT(\alpha)]\neg O(\alpha)$
6. $\models O(\beta) \longrightarrow [COMMIT(\alpha)]O(\beta)$ *if* $\alpha \& \beta =_{\mathcal{D}}$ **fail**
7. $\models \neg O(\beta) \longrightarrow [COMMIT(\alpha)]\neg O(\beta)$ *if* $\alpha \& \beta =_{\mathcal{D}}$ **fail**
8. $\models [\beta]INT(\alpha_2) \longrightarrow [DECIDE(\alpha_1)][\beta]INT(\alpha_2)$
9. $\models [\beta]\neg INT(\alpha_2) \longrightarrow [DECIDE(\alpha_1)][\beta]\neg INT(\alpha_2)$
10. $\models [\beta]O(\alpha_2) \longrightarrow [COMMIT(\alpha_1)][\beta]O(\alpha_2)$
11. $\models [\beta]\neg O(\alpha_2) \longrightarrow [COMMIT(\alpha_1)][\beta]\neg O(\alpha_2)$

(1) states that after deciding to do the action denoted by α I also intend to do α. (2) states the complement namely, that if I decide to perform the action denoted by α I do not intend to perform an action that is incompatible with α. Actually (3) states the property for which the definitions are devised. If I intend to perform the action denoted by α then I am obligated to perform that action after I have committed myself to it. (4) states basically the same in a different format. (5),(6) and (7) state that a commitment has no influence on actions that are not intended and on actions that have nothing to do with the action committed to. The last four items state that decisions and commitments only influence the state in which they are made.

Having introduced all the ingredients in our logic, in the next section we will show how they can be used to model the examples given in the introduction.

4 Examples modelled

The first example models the decisions taken in a court case. We have simplified the formulas a little bit to make it easier to follow the example. First we assume that if a crime has been committed in the past and it is not yet punished then it should be our intention to punish the crime:

$$PAST(crime) \wedge \neg PAST(punish) \longrightarrow O(INT(punish))$$
$$\neg PAST(crime) \longrightarrow O(INT(dismiss))$$

I.e. we have a moral obligation to punish crimes and to dismiss the defendant if no crime has been committed. However, we still might decide to dismiss the defendant (because of mitigating circumstances or not enough evidence to convince us of the fact that the crime actually was committed by the defendant). To model these possibilities we use the following formulas:

$$INT(punish) \longrightarrow [DECIDE(dismiss)]\neg INT(punish)$$
$$INT(dismiss) \longrightarrow [DECIDE(punish)]\neg INT(dismiss)$$
$$PAST(crime) \longrightarrow [DECIDE(punish)]INT(punish) \wedge O(INT(punish))$$
$$PAST(crime) \longrightarrow [DECIDE(dismiss)]\neg INT(punish) \wedge O(INT(punish))$$
$$\neg PAST(crime) \longrightarrow [DECIDE(punish)]\neg INT(dismiss) \wedge O(INT(dismiss))$$
$$\neg PAST(crime) \longrightarrow [DECIDE(dismiss)]INT(dismiss) \wedge O(INT(dismiss))$$

The above states that it is morally wrong to punish someone for a crime he did not commit or to dismiss someone while he committed a crime. The above formulas do not state yet what the course of actions should be. This depends on the commitment to the decision that is reached:

$$INT(punish) \longrightarrow [COMMIT(punish)]O(punish)$$
$$INT(dismiss) \longrightarrow [COMMIT(dismiss)]O(dismiss)$$

The above formulas can model a conflict between a moral obligation and a legal obligation. If the crime has been committed by the defendant there is a (moral) obligation to punish the defendant. However, if we decide to dismiss him and commit ourselves to that decision there also exists a (legal) obligation to set the defendant free. I.e. we have $O(INT(punish)) \wedge O(dismiss)$.

The second example does not make explicit use of the decision operator. We assume that in the case that some money was subtracted from the account of a client of the bank by mistake and this mistake was rectified by a compensating money transfer later on then we want to inform the client and

have an obligation to restitute any interest that was payable because of a negative balance due to the mistaken money transfer. This is modelled by the following formulas:

$$PAST(mistake) \wedge PAST(compensate) \longrightarrow INT(inform\_client) \wedge O(rest\_interest)$$

$$PAST(mistake) \wedge PAST(compensate) \longrightarrow [COMMIT(inform\_client)]O(inform\_client)$$

The obligation to restitute the interest arises directly from the mistake made in the past, while the obligation to inform the client arises from a (voluntary) commitment. We can contrast the above formulation with the situation in which the mistake was recovered by executing a roll-back operation. In this case the client does not notice the mistake at all and therefore does not have to be informed of it. The client also does not miss any interest. Therefore we get the following formula:

$$PAST(mistake) \wedge PAST(rollback) \longrightarrow \neg INT(inform\_client) \wedge \neg O(rest\_interest)$$

The above simple examples show the power of the logic that is proposed in this paper. We simplified the examples to be able to concentrate on the central issues that we wanted to bring up. Therefore we did not model the fact that the compensating action should be executed after the mistake. We also assumed that only one mistake was made in the past. To model these points we would need to introduce sequences of actions and operators on them. This is left out because of the space limitations.

5 Discussion and Conclusions

In this paper we have given a modal approach to deontic logic that integrates static (Ought_to_be) and dynamic (Ought_to_do) deontic operators. Through the inclusion of a few (simple) temporal operators it is possible to express the dependence of obligations, prohibitions and permissions on a past situation.

The introduction of the two (meta-)actions $COMMIT$ and $DECIDE$ makes it possible to model (in some way desired) future behaviour. Because, within the logic we cannot simulate the actual occurrence of actions, the effects of this (desired) future behaviour are expressed in obligations to perform certain actions.

In section 4 we have shown that with this integrated deontic logic it is possible to model the practical and frequently occurring examples as given in the introduction in a simple and elegant way. Due to the limited space many aspects of the logic have not yet been discussed. For instance, we have not given an axiom and inference schema for this logic. We leave this for the full paper.

Another point that we have not discussed yet is the nature of the "intention" relation I. We have taken the semantic relation I as given. However, it is also possible to give this relation a deontic connotation by relating it to a deontic ordering of the states. Let $\leq_D$ denote a deontic ordering on the states such that $\sigma \leq_D \sigma'$ if σ is deontically worse than σ'. Then we can give the following restriction on the relation I:

Definition 18

$$I(\sigma, \sigma') \Longrightarrow R(\sigma, \sigma') \wedge \forall \sigma'' \in \Sigma (R(\sigma, \sigma'') \Longrightarrow \sigma'' \leq_D \sigma')$$

In order for this restriction to be sensible $<_D$ has to be reflexive and transitive. It does not have to be total over Σ, but it is plausible to require that it is total on every set $W \subseteq \Sigma$ such that $\exists \sigma \in \Sigma : \forall \sigma' \in W : R(\sigma, \sigma')$. I.e. the ordering is total on every set of states that is reachable from one state by performing some action. Some ideas about how to define this ordering can been found in [DMW94]. The above restriction states that only deontically preferable action can be the actions that are intended to be done next. I.e. the restriction ensures us that the model will behave in a moral way. Of course, other types of restrictions can be given. E.g. one might take the more realistic option of demanding the opposite of the above restriction. I.e. intended actions are never deontically preferred. Here we do not want to make a choice in this respect, but just point out the possibilities of the approach introduced in this paper.

References

[Aqv84] L. Åqvist. Deontic logic. In D.M. Gabbay and F. Guenthner, editors, *Handbook of Philosophical Logic II*, pages 605–714. Reidel, 1984.

[BW90] J.C.M. Baeten and W.P. Weijland. Process Algebra. Cambridge University Press, 1990.

[Bro86] M. Broy. A theory for nondeterminism, parallelism, communication and concurrency. *Theoretical Computer Science*, vol.45, pages 1–62, 1986.

[dBKM+86] J.W. de Bakker, J.N Kok, J.-J.Ch. Meyer, E.-R. Olderog, and J.I. Zucker. Contrasting themes in the semantics of imperative concurrency. In J.W. de Bakker, W.P. de Roever, and G. Rozenberg, editors, *Current Trends in Concurrency: Overviews and Tutorials*, pages 51–121. LCNS 224 Springer, Berlin, 1986.

[Date95] C.J. Date. An introduction to database systems Addison-Wesley, Amsterdam, 1995.

[DM90] F. Dignum and J.-J.Ch. Meyer. Negations of transactions and their use in the specification of dynamic and deontic integrity constraints. In M. Kwiatkowska, M.W. Shields, and R.M. Thomas, editors, *Semantics for Concurrency, Leicester 1990*, pages 61–80, Berlin, 1990. Springer.

[Dig92] F. Dignum. Using transactions in integrity constraints. *Workshop on Applied Logic*, Amsterdam, 1992.

[DMW94] F. Dignum, J.-J.Ch. Meyer, and R. Wieringa. A dynamic logic for reasoning about sub-ideal states. In J. Breuker, editor, *ECAI workshop on Artificial Normative Reasoning*, pages 79–92, Amsterdam, 1994.

[Eme89] E.A. Emerson. Temporal and Modal Logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science*, pages 995-1072, North-Holland, Amsterdam.

[Har79] D. Harel. *First Order Dynamic Logic*. Springer, 1979. Lecture Notes in Computer Science 68.

[LHM95] B. van Linder, W. van der Hoek and J.-J.Ch. Meyer How to motivate your agents. On making promises that you can keep. to appear as technical report of the RUU, Utrecht, The Netherlands.

[Mey87] J.-J.Ch. Meyer. A simple solution to the 'deepest' paradox in deontic logic. *Logique et Analyse, Nouvelle Série*, vol.30, pages 81–90, 1987.

[Mey88] J.-J.Ch. Meyer. A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, vol.29, pages 109–136, 1988.

[SV85] J.R. Searle and D. Vanderveken, *Foundations of illocutionary logic* Cambridge University Press. 1985.

[Sea69] J.R. Searle, *Speech Acts* Cambridge University Press. 1969.

[WWMD91] R. Wieringa, H. Weigand, J.-J.Ch. Meyer, and F. Dignum. The inheritance of dynamic and deontic integrity constraints. In *Annals of Mathematics and Artificial Intelligence 3*, pages 393–428. Baltzer A.G., 1991.

[Wri51] G.H. von Wright. Deontic logic. *Mind*, vol.60, pages 1–15, 1951.