A Dynamic Deontic Logic for Complex Contracts[☆]

Cristian Prisacariu[*,a], Gerardo Schneider[b,a]

[a]Department of Informatics – University of Oslo, Norway
[b]Department of Computer Science and Engineering – Chalmers | University of Gothenburg, Sweden

Abstract

We present a dynamic deontic logic for specifying and reasoning about complex contracts. The concepts that our contract logic $\mathcal{CL}$ captures are drawn from legal contracts, as we consider that these are more general and expressive than what is usually found in computer science (like in software contracts, web services specifications, or communication protocols). $\mathcal{CL}$ is intended to be used in specifying complex contracts found in computer science. This influences many of the design decisions behind $\mathcal{CL}$. We adopt an *ought-to-do* approach to deontic logic and apply the deontic modalities exclusively over complex actions. On top, we add the modalities of dynamic logic so to be able to reason about what happens after an action is performed. $\mathcal{CL}$ can reason about regular synchronous actions capturing the notion of actions *done at the same time*. $\mathcal{CL}$ incorporates the notions of contrary-to-duty and contrary-to-prohibition by attaching to the deontic modalities explicitly a reparation which is to be enforced in case of violations. Results of decidability and tree model property are given as well as specific properties for the modalities.

Key words: e-contracts, legal contracts, deontic logic, PDL, synchrony, semantics, normative structure, decidability, synchronous actions, action negation

Contents

[☆]Partially supported by the Nordunet3 project "COSoDIS – Contract-Oriented Software Development for Internet Services" (`http://www.ifi.uio.no/cosodis/`).

[*]Corresponding author's address: P.O.Box 1080 Blindern, N-0316 Oslo, Norway.

Email addresses: `cristi@ifi.uio.no` (Cristian Prisacariu), `gersch@chalmers.se` (Gerardo Schneider)

1. Introduction

The present paper reports on the state-of-the-art of the logic for contracts $\mathcal{CL}$. The goal of $\mathcal{CL}$ is to describe and prescribe, at an abstract level, behaviors of complex systems, as for instance concurrent programs, communicating intelligent agents, web services, or normative systems. From this point of view, $\mathcal{CL}$ needs to be expressive enough to capture behaviors of such systems. The purpose of this logic is not only to formalize such behaviors, but also to reason about them. Therefore, we aim at a decidable logic so to have hopes for automatic verification using formal tools like model-checking and run-time monitoring.

More precisely, $\mathcal{CL}$ has been designed to represent and reason about contracts (being that software contracts, web services, interfaces, communication protocols, etc), and it combines deontic logic (i.e., the logic of obligations, permissions, and prohibitions) [1] with propositional dynamic logic (PDL, the logic of actions) [2]. The deontic part of $\mathcal{CL}$ can express obligations, permissions and prohibitions over structured actions, as well as what happens when obligations or prohibitions are not respected. The dynamic part of $\mathcal{CL}$ expresses what happens after some action (possibly with complex structure) is performed.

A first version of the language $\mathcal{CL}$ has been presented in [3], where explicit temporal operators (always, eventually, and until) were part of the syntax. An encoding into a version of the modal μ-calculus with concurrent actions was used to give semantics. The $\mathcal{CL}$ language presented in this paper is more expressive and has a cleaner syntax (with no syntactic restrictions); it was first introduced in [4]. A variant of $\mathcal{CL}$ (without the propositional constants) was used in [5] for doing run-time monitoring of electronic contracts using a restricted semantics based on traces of actions. This semantics was specially designed for monitoring the actions of the contracting parties at run-time with the purpose of detecting when the contract is violated. The presentation of [5] did not give much explanation nor examples about the intuitions behind the choices in the design of $\mathcal{CL}$. We do this in the present paper and discuss the full semantics of $\mathcal{CL}$ based on *normative structures*. We focus on the intended properties of the language. This paper is a revised and extended version of [4], where besides presenting full proofs we have the following added contributions: (i) a complete new section on results concerning the underlying action algebra of $\mathcal{CL}$, (ii) new results concerning the decidability of the logic, and (iii) further results on its semantics.

In the rest of this introductory section we focus on the informal explanation of the language and in particular on its design decisions. These design decisions are biased by the fact that $\mathcal{CL}$ is intended to faithfully capture concepts and natural properties from electronic contracts and to avoid the main deontic paradoxes.

1.1. Motivation and design decisions

In this section we motivate the particular choices we made in the design of the $\mathcal{CL}$ contract specification language. We compare to related works and give informal intuitions and examples.

The design of $\mathcal{CL}$ and specially its semantics is driven by the need to express and reason about contracts. More particularly, $\mathcal{CL}$ needs to talk about obligations, permissions and prohibitions; but more specific, about actions and the fact that such actions are obligatory, permitted or forbidden. Actions abound in contracts, (particularly in legal contracts) as for example: “supply false information”, “notify”, or “pay later”. $\mathcal{CL}$ is working at a rather high level of abstraction and many of the details of the actions are abstracted away. In fact, much of the open problems and possible continuations for the $\mathcal{CL}$ logic are gathered around making such details of actions explicit and accounted for in a richer logic.[1]

Because $\mathcal{CL}$ talks about obligations applied to actions it can be included in the class of logics that take the so called ought-to-do approach to deontic logic. But more than this, $\mathcal{CL}$ needs to talk about *what happens after some action*, like in the example “after notification he must...”. Such notions are captured with the PDL part of $\mathcal{CL}$.

Ultimately $\mathcal{CL}$ is a logic which talks about *complex actions* because actions can have some more structure than just simple names like “pay”, “delay”, or “drive”. The minimal structure that appears in most contracts is: *sequencing* of actions, where one must do an action and then do another action, like “lower the Internet traffic and then pay”; and another kind of structuring is *choices*, like “one has the choice to pay with either euro or dollars”. The structure that is particular to $\mathcal{CL}$ is that of *actions done at the same time*, which are typically found in everyday contracts, as for example “drink and drive” or paraphrasing from our example above “delay and notify”. Our understanding of such concurrent actions and the way to abstract them in a logic is through the concurrency notion or synchrony. The integration of all these features is not trivial and was done in an algebraic formalism in [7]. We summarize the main results needed for the present paper in Section 2. In what follows we describe some related work.

We adopt a deontic logic over actions, as started by G.H. von Wright [8], and integrates these deontic operators with the PDL modality. In this sense $\mathcal{CL}$ might be thought of entering the line of dynamic deontic logics as started by J.-J. Ch. Meyer [9], but this is not the case as in $\mathcal{CL}$ the deontic modalities are not expressed in terms of the PDL modality (or in terms of the modal μ-calculus [10]). But still $\mathcal{CL}$ has both dynamic and deontic flavor.

$\mathcal{CL}$ integrates a notion of concurrent actions (i.e., the synchrony) and in this respect it intersects with both works from PDL with concurrent actions and with works in the dynamic deontic logic community.

Another important notion in legal contracts is that of a reparation sentence which is somehow directly related to the obligation or prohibition that it repairs. This is somehow related to the well known problem of contrary-to-duty, but in $\mathcal{CL}$ we are in the setting of actions and not in that of standard deontic logic. In $\mathcal{CL}$ the standard notion of contrary-to-duty takes a flavor of reparations enforced after the violating actions.

[1]For more open problems and research directions see [6].

The purpose of $\mathcal{CL}$ is to specify and reason about contracts, therefore it integrates the normative notions of *obligation*, *permission*, and *prohibition*. These have been extensively investigated in the deontic logic community since its introduction by von Wright in [1]. The deontic notions that we introduce in $\mathcal{CL}$ are different than the ones in standard deontic logic (SDL) in several respects, as discussed in the rest of this section.

(1) The deontic modalities are applied only over actions instead of over propositions (or state of affairs). This is known as the ought-to-do approach to deontic logic as opposed to the more classic ought-to-be approach of SDL. The ought-to-do approach has been advocated by G.H. von Wright [8] which argued that deontic logic would benefit from a "foundation of actions", since many of the philosophical paradoxes of SDL would then not occur; a point which is made clear and precise in [11]. Important contributions to this approach were done by K. Segerberg for introducing actions inside the deontic modalities [12, 13] and by the seminal work of J.-J.Ch. Meyer on dynamic deontic logic (DDL) [9] (see also [10, 14]).

Compared to [9, 10, 15, 16], which also consider deontic modalities (i.e., O, P, and F) applied over actions, the investigation presented in this paper at the level of the actions is different in several ways as we elaborate below. The formalization of the actions is summarized in Section 2 and thoroughly investigated in [7] where standard models are defined for actions and completeness results are established. The semantics of the $\mathcal{CL}$ language is based on this interpretation of the actions as special trees.

(2) The action combinators are the standard $+$ and $\cdot$ (for *choice* and *sequence*) but exclude the Kleene star *. This exclusion is only for the actions appearing inside the deontic modalities (i.e., the deontic actions); for otherwise we allow the Kleene * inside the dynamic box modality as is standard (as see in Section 4). None of the few papers that consider repetition (e.g. using *) as an action combinator under deontic modalities [15, 10] give a convincing motivation for having such recurring actions inside obligations, permissions, or prohibitions. In fact its use inside the deontic modalities seems counter-intuitive: take the expression $O(\alpha^*)$ - which, using for now our intuition for the Kleene * and the obligation modality O, is read "One is obliged to not pay, or pay once, or pay twice in a row, or..." – which puts no actual obligations; or take $P(\alpha^*)$ – "One has the right to do any sequence of action α" – which is a very shallow permission and is captured by the widespread *Closure Principle* in jurisprudence where *what is not forbidden is permitted* [12]. Moreover, as pointed out in [10], expressions like $F(\alpha^*)$ and $P(\alpha^*)$ can be simulated with the propositional dynamic logic (PDL) modalities along with deontic modalities over actions without the Kleene star *; anticipating the syntax and semantics that we present later in this paper, consider e.g. $F(\alpha^*) \stackrel{\Delta}{=} [\alpha^*]F(\alpha)$, where all the discussion above holds for α being the abstraction of any complex action.

The theory that we develop for $\mathcal{CL}$, i.e., the semantics and various proofs, is already quite involved without using the * inside the deontic modalities. If we were to add the Kleene * to capture some esoteric examples that one might find appealing the complexity that this would trigger in terms of theory and proofs does not justify its effort.

(3) $\mathcal{CL}$ defines an *action complement* operation which encodes the violation of an

obligation. Obligations (and prohibitions) can be violated by *not* doing the obligatory action (respectively *doing* the forbidden action). The action complement that we have is different from the various notions of action negation found in the literature on PDL or DDL-like logics [9, 17, 18, 19]. In [9], as in [17], action negation is with respect to the universal relation which for PDL gives undecidability. Decidability of PDL with negation of only atomic actions has been achieved in [18]. A so called "relativized action complement" is defined in [19] which is the complement of an action (not w.r.t. the universal relation but) w.r.t. a set of atomic actions closed under the application of some action operators. This kind of negation still gives undecidability when several action operators are involved.

In $\mathcal{CL}$ the *action complement* is a derived operator defined as a function which takes a compound action and returns another compound action, i.e., it is not a principal combinator like $+$, $\cdot$, or $\times$. Intuitively the complement comprises of *all* the immediate actions that *take us outside* the tree of the complemented action [10].

(4) One difference from the standard PDL is that we consider *deterministic* actions. This is natural and desired in legal contracts as opposed to the programming languages community where nondeterminism is an important notion. In programming languages a nondeterministic action can be "send message through the network" which may have two outcomes: wither the message is received, or the message is not received, as lost by the network. In contrast, a deterministic action (as in eg. deterministic automata) has a single outcome. In legal contracts the outcome of an action like "deposit 100$ in the bank account" is uniquely determined. In $\mathcal{CL}$ we take inspiration from the deterministic PDL which has been investigated in [20]. Deterministic PDL is undecidable if action negation (or intersection of actions) is added [17].

(5) We add a concurrency operator $\times$ to model that two actions are *done at the same time*. The model of concurrency that we adopt is the synchrony model of R. Milner's SCCS [21]. Synchrony is a natural choice when reasoning about the notion "at the same time" for human-like actions as we have in legal contracts (opposed to the instructions in a programming language). Moreover, from an algebraic point of view, synchrony is easy to integrate with the other regular operations on actions (the choice and the sequence).

The notion of synchrony has different meanings in different areas of computer science. Here we take the distinction between *synchrony* and *asynchrony* as presented in the SCCS calculus and later implemented in, e.g., the Esterel synchronous programming language [22]. We understand *asynchrony* as when two concurrent systems proceed at indeterminate relative speeds (i.e., their actions may have different non-correlated durations); whereas in the *synchrony* model each of the two concurrent systems instantaneously perform a single action at each time instant. This is an abstract view of the actions found in contracts which is good for reasoning about quite a big range of properties for contracts, like properties that do not take into consideration the structure or types of the actions. Such properties would look only at the interplay of actions, temporal ordering, choice, or existence of actions. If one needs actions which have durations (e.g., "work 3 hours") or which are parameterized by amounts (e.g., "deposit 100$") then $\mathcal{CL}$ has to be extended accordingly.

The *synchrony model* of concurrency takes the assumption that time is discrete and that basic actions are instantaneous and represent the time step. Moreover, at each time step all possible actions are performed, i.e., the system is considered *eager* and *active*. For this reason, if at a time point there is enabled an obligation to do an action, then this action must be immediately executed so that the obligation is not violated. Synchrony assumes a global clock which provides the time for all the actors (participants, parallel components) in the system. Note that for practical implementation purposes this is a rather strong assumption which offends the popular view from process algebras [23, 24]. On the other hand the mathematical framework of the synchrony model is much cleaner and more general than the asynchronous interleaving model (SCCS has the (asynchronous) CCS as a subcalculus [21]). The synchronous composition operator $\times$ is different from the classical $\|$ of CCS.

The synchrony model is better suited for *reasoning* about concurrent actions than for implementing concurrency as is the more low-level asynchrony model. Because of the assumption of an eager behavior for the actions the scope of the obligations (and of the other deontic modalities too) is immediate, making them transient obligations which are enforced only in the current point in time. One can get persistent obligations by using temporal operators, like the *always* operator. The eagerness assumption facilitates both reasoning about existence of the deontic modalities and about violations of the obligations or prohibitions.

Regarding the dynamic logic part, $\mathcal{CL}$ introduces the synchrony operation $\times$ on the actions inside the dynamic modality. Therefore, $\mathcal{CL}$ can use dynamic logic reasoning about synchronous actions and, from this point of view, it is included in the class of extensions of PDL that can reason about concurrent actions: PDL$^{\cap}$ with intersection of actions [25] which is undecidable for deterministic structures or concurrent PDL [26, 27] which adopts ideas from alternating automata [28]. Contrasting with the discouraging undecidability results from above, $\mathcal{CL}$ (with action complement and synchronous composition over deterministic actions inside the dynamic modality) is decidable. This makes $\mathcal{CL}$ more attractive for automation of reasoning about contracts.

(6) $\mathcal{CL}$ defines a *conflict relation* $\#_{\mathcal{C}}$ over actions which represents the fact that two actions cannot be done at the same time. This is necessary for detecting (and for ruling out) a first kind of *conflicts* in contracts: “Obligatory to go west and obligatory to go east” should result in a conflict because the actions “go west” and “go east” cannot be done at the same time (i.e., are conflicting). The second kind of conflicts that $\mathcal{CL}$ rules out are: “Obligatory to go west and forbidden to go west” which is a standard requirement on a deontic logic.

(7) In $\mathcal{CL}$ *conditional obligations* (or prohibitions) can be of two kinds.

a. The first kind is given with the propositional implication: $\mathcal{C}_1 \rightarrow O_{\mathcal{C}}(\alpha)$ which is read as “if $\mathcal{C}_1$ is the case then action α is obligatory” (e.g., “If Internet traffic is high then the Client is obliged to pay”).

b. The second kind is given with the dynamic box modality: $[\beta]O_{\mathcal{C}}(\alpha)$ which is read as “if action β was performed then action α becomes obligatory” (e.g., “After receiving necessary data, the Provider is obliged to offer password”).

(8) Regarding the deontic modalities, $\mathcal{CL}$ includes directly in the definition of the

obligation and prohibition the *reparations* in case of violations. The deontic modalities are $O_{\mathcal{C}}$ and $F_{\mathcal{C}}$ where $\mathcal{C}$ is a contract clause representing the reparation. This models the notions of contrary-to-duty obligations (CTDs) and contrary-to-prohibitions (CTPs) as found in deontic logic applied over actions like DDL [9, 14]. These notions are in contrast with the classical notion of CTD as found in the SDL literature [29, 30]. In SDL, what we call reparations are secondary obligations which hold in the same world as the primary obligation. In our setting, where the action changes the context (the world), one can see a violation of an obligation (or prohibition) only after the action is performed and thus the reparations are enforced in the next world (in the changed context).

The approach of $\mathcal{CL}$ to contrary-to-duty rules out many of the problems faced by SDL (like the gentle murderer paradox). On the other hand it does not capture the wording of the SDL examples. In the end of Section 4.1 we present the stand of $\mathcal{CL}$ w.r.t. some of the most important paradoxes of SDL.

(9) Standard deontic logic SDL, and other variants of it, consider one of the three deontic modalities as primitive (usually O or P) and the other two modalities are defined in terms of this primitive one using the propositional operators. To the contrary, the deontic modalities are *not interdefinable* in $\mathcal{CL}$. Only some of the implications that SDL makes hold in $\mathcal{CL}$, and we discuss these in Section 3.2.

(10) The *semantics* of $\mathcal{CL}$ is given in terms of *normative structures* and it is specially defined to capture several natural properties which are found in legal contracts. These are motivated (with examples) in Section 4.

A work, close in many respects with our work here, was recently presented in DEON [31] and is, like us, essentially inspired by [12] and [16]. Their work is particularly appealing because of the neat algebraic presentation. Moreover, [31] carefully investigates the different axiomatizations of permissions and prohibitions, where small differences in the intuitive understanding of their relations are being explicitly formalized by the set of characterizing axioms. Nevertheless, they base their nice presentation on a simpler set of actions, which are characterized by the nicely behaved Boolean algebra. Because of this they do not investigate sequences of actions, and also do not have an easy transition to the dynamic actions (i.e., the regular synchronous actions including the Kleene *, as we do with the synchronous Kleene algebra formalization). Another difference with $\mathcal{CL}$ is that [31] defines obligation in terms of permission, which is not the case in $\mathcal{CL}$. At the semantic level, the same notion of markers as we have in normative structures are used in [31] only that they have a more algebraic view of the sets of markers related to the interpretation of the actions in the spirit of Boolean algebra, as opposed to our interpretation in the spirit of Kleene algebra.

2. Synchronous Actions

In this section we present the formalism of the synchronous actions that are the basis of the $\mathcal{CL}$ logic. We provide here important results about synchronous actions needed in later sections when giving the semantics of $\mathcal{CL}$. We introduce actions gradually, first defining *deontic actions* which will be the actions used inside the deontic

(1) $\alpha + (\beta{+}\gamma) = (\alpha{+}\beta){+}\gamma$	(10) $\alpha\times(\beta\times\gamma) = (\alpha\times\beta)\times\gamma$
(2) $\alpha + \beta = \beta + \alpha$	(11) $\alpha\times\beta = \beta\times\alpha$
(3) $\alpha + \mathbf{0} = \mathbf{0} + \alpha = \alpha$	(12) $\alpha\times\mathbf{1} = \mathbf{1}\times\alpha = \alpha$
(4) $\alpha + \alpha = \alpha$	(13) $\alpha\times\mathbf{0} = \mathbf{0}\times\alpha = \mathbf{0}$
(5) $\alpha\cdot(\beta\cdot\gamma) = (\alpha\cdot\beta)\cdot\gamma$	(14) $a\times a = a \quad \forall a \in \mathcal{A}_B$
(6) $\alpha\cdot\mathbf{1} = \mathbf{1}\cdot\alpha = \alpha$	(15) $\alpha\times(\beta+\gamma) = \alpha\times\beta + \alpha\times\gamma$
(7) $\alpha\cdot\mathbf{0} = \mathbf{0}\cdot\alpha = \mathbf{0}$	(16) $(\alpha+\beta)\times\gamma = \alpha\times\gamma + \beta\times\gamma$
(8) $\alpha\cdot(\beta{+}\gamma) = \alpha\cdot\beta + \alpha\cdot\gamma$	(17) $(\alpha_\times\cdot\alpha)\times(\beta_\times\cdot\beta) = (\alpha_\times\times\beta_\times)\cdot(\alpha\times\beta), \forall\alpha_\times,\beta_\times\in\mathcal{A}_B^\times$
(9) $(\alpha{+}\beta)\cdot\gamma = \alpha\cdot\gamma + \beta\cdot\gamma$	

Table 1: Axioms of action equality.

modalities. These actions are then enriched with *tests* and the Kleene * operator, becoming, what we call, *dynamic actions* because these will be used inside the dynamic box modality of $\mathcal{CL}$. All these actions are called *synchronous* because all include the synchrony operator that we mentioned in the introduction.

Definition 2.1 (deontic actions). *Consider a finite set of* basic *(or atomic) actions $\mathcal{A}_B$ (denoted by $a, b, c, \ldots$). The special actions $\mathbf{0}, \mathbf{1} \notin \mathcal{A}_B$ are called respectively the* violating *and the* skip *actions. The action combinators are: "$+$" for* choice *of two actions, "$\cdot$" for* sequence *of two actions (or concatenation), and "$\times$" for* concurrent composition *(synchronously) of two actions. We generally call* compound actions *(or just actions) terms of $\mathcal{A}^{\mathcal{D}}$ (denoted $\alpha, \beta, \gamma, \ldots$, possibly primed, double-primed, etc.) obtained from basic actions, $\mathbf{0}$, and $\mathbf{1}$ using the action combinators. We call* $\times$-actions, *denoted $\alpha_\times$, $\beta_\times$, $\gamma_\times$ (possibly primed, etc.) the subset of actions $\mathcal{A}_B^\times \subset \mathcal{A}^{\mathcal{D}}$ generated from $\mathcal{A}_B$ using only the $\times$ constructor. The actions defined here will be referred to as* deontic actions *and can be seen as generated by the grammar below:*

$$\alpha \quad \triangleq \quad a \mid \mathbf{0} \mid \mathbf{1} \mid \alpha\times\alpha \mid \alpha\cdot\alpha \mid \alpha+\alpha \qquad \textit{(deontic actions)}$$

To avoid unnecessary parentheses we use the following precedence over the combinators: $+ < \cdot < \times$. Table 1 axiomatizes the equality between actions.

Actions as presented here are related to the more general algebraic structure called *synchronous Kleene algebra* in [7]. Note that $\mathbf{0}, \mathbf{1} \notin \mathcal{A}_B^\times$ and the inclusion of sorts $\mathcal{A}_B \subseteq \mathcal{A}_B^\times \subset \mathcal{A}^{\mathcal{D}}$. Also note that $\mathcal{A}_B^\times$ is finite up to the application of the axioms, because $\mathcal{A}_B$ is finite and $\times$ is idempotent over basic actions; see axiom (14). Because of the idempotence we take the liberty of using $\subseteq$ to compare elements of $\mathcal{A}_B^\times$, as $\alpha_\times$ can be seen as the set of basic actions that compose it (e.g. $a\times b \subseteq a\times b\times c$ or $a \not\subseteq b\times c$). We will say that β is *bigger* than α whenever $\alpha \subseteq \beta$. So, in the example above $a\times b\times c$ is bigger than $a\times b$.

In the rest of the paper we consider the set $\mathcal{A}_B^\times$ up to the application of the axioms of Table 1. Particularly, only axioms (10), (11), and (14) are applicable to $\times$-actions (i.e., when we talk about $\times$-actions we talk about the representative of an equivalence class of $\times$-actions w.r.t. axioms (10), (11), and (14)). This representative is the minimal one, like $a\times b$ instead of $a\times b\times b$ or $a\times b\times b\times b$. This is also why the $\mathcal{A}_B^\times$ is finite.

With $\mathcal{CL}$ we are reasoning about the structure of the complex actions, where the particular atomic actions of $\mathcal{A}_B$ are abstracted away to being just some symbols; these can be made concrete by the user, as for example in programming languages where each assignment that the program makes is one atomic action. In legal contracts an atomic action can be, e.g., “Client pays 100$”.

Intuitively, we consider that the actions are performed by somebody (being that a person, a program, or an e-agent). We talk about “performing actions” and one should not think of *processes “executing” actions* and operational semantics like in SCCS; we do not discuss operational semantics nor bisimulation equivalences in this paper.

Definition 2.2 (conflict relation). *Consider a symmetric and irreflexive relation over basic actions $\mathcal{A}_B$ called* conflict relation *and denoted by* $\#_\mathcal{C} \subseteq \mathcal{A}_B \times \mathcal{A}_B$.

The *conflict relation* is a notion often found in legal contracts and is given a priori. The intuition is that if two actions are in conflict then the actions cannot be done at the same time. This intuition explains the need for the following equational implication:

$$(18) \quad a \,\#_\mathcal{C}\, b \rightarrow a \times b = \mathbf{0} \qquad \forall a, b \in \mathcal{A}_B.$$

There is *no transitivity* of $\#_\mathcal{C}$ which is natural as also shown by the following example: action “drive” may be in conflict with both “drink” and “talk at the phone” but still one can, legally, “drink and talk at the phone” at the same time, though not possible physically.

From an algebraic point of view, the purpose of the $\#_\mathcal{C}$ relation is to add more structure on the algebra that was not there before (only from the properties of the operators). This extra structure comes from outside, from some properties given by an oracle (by the user) on the generators of the algebra (i.e., on the basic actions). By more structure is meant that new equalities hold depending on the information given through $\#_\mathcal{C}$. The purpose of $\#_\mathcal{C}$ is different than what is sometimes done in algebra: eg., on an idempotent semiring one defines a relation $\alpha \leq \beta$ iff $\alpha + \beta = \beta$ in terms of some special structure on the algebra, and studies this structure by studying this relation instead (eg., $\leq$ is reflexive, which means that the properties of the $+$ operator are such that $\alpha + \alpha = \alpha$, i.e., idempotence). In the case of $\#_\mathcal{C}$ we do not define it in terms of some existing structure (and properties) in the algebra (i.e., in terms of some equalities like $a \times b = \mathbf{0}$), but we use $\#_\mathcal{C}$ to impose some more structure on the algebra. Therefore we added the equational implication (18).

Definition 2.3 (canonical form). *We say that an action α is in* canonical form, *denoted by $\underline{\alpha}$, iff it is $\mathbf{0}$, $\mathbf{1}$, or has the following form:*

$$\underline{\alpha} \stackrel{\Delta}{=} +_{i \in I} \, (\alpha^i_\times \cdot \underline{\alpha^i})$$

where

a. $\alpha^i_\times \neq \alpha^j_\times$*, for all* $i, j \in I$*;*

b. for all $i \in I$*, either*

  (a) $\alpha^i_\times \in \mathcal{A}^\times_B$ *and* $\underline{\alpha^i} \in \mathcal{A}^\mathcal{D} \setminus \{\mathbf{0}, \mathbf{1}\}$ *is an action in canonical form, or*

  (b) $\alpha^i_\times \in \mathcal{A}^\times_B \cup \{\mathbf{1}\}$ *and* $\underline{\alpha^i}$ *is absent (i.e., when no more $\cdot$ applications exist).*

Note that $\alpha_\times^i$ does not contain twice the same action a, for any $a \in \mathcal{A}_B$ and $i \in I$, as it is part of $\mathcal{A}_B^\times$, thus, $a\times a$ is not in canonical form, but a and $a\times b$ are. The indexing set I is finite as $\mathcal{A}_B^\times$ is finite and $\alpha_\times^i$ are different. The purpose of the constraints in Definition 2.3.b is to not allow $\mathbf{1}$ to appear in the canonical forms, except for some very special cases: either the whole action $\underline{\alpha} = \mathbf{1}$; or $\mathbf{1}$ appears in a summation with other actions as summands. All other possible appearances of $\mathbf{1}$ are disallowed by this constraint 2.3.b. Particularly, 2.3.ba does not allow an $\underline{\alpha^i}$ alone to be $\mathbf{1}$ and 2.3.bb says that a $\alpha_\times^i \in \mathcal{A}_B^\times$ is allowed to take the value $\mathbf{1}$ (which was not the case before, as being defined from $\mathcal{A}_B^\times$) only in the case when it is not followed by any other action.

Example 2.1 Consider the complex action $\alpha = (a + b) \cdot (c + d)$ which is not in canonical form but is equivalent to the canonical form $\underline{\alpha} = a \cdot (c + d) + b \cdot (c + d)$ (obtained by applying axiom (9)). On the other hand α is equivalent also to $a\cdot c+a\cdot d+b\cdot c+b\cdot d$ (obtained by applying axioms (9) then (8)) which is not in canonical form because the constraint *a* in Definition 2.3 is not met as a appears twice as first element in the summation.

Related to the constraints in Definition 2.3.b the action $\mathbf{1} \cdot a$ is not in canonical form because of 2.3.bb (and neither is $b+\mathbf{1}\cdot a$), and action $a \cdot \mathbf{1}$ is not in canonical form because of 2.3.ba (and neither is $b + a \cdot \mathbf{1}$), but action $a + \mathbf{1}$ is in canonical form because there is nothing following the branch represented by $\mathbf{1}$ and therefore the constraint of 2.3.bb allows this branch to be $\mathbf{1}$. From the same reasons the action $a \cdot (\mathbf{1} + b)$ is also in canonical form. □

Theorem 2.4 ([7, Th.2.8]). *For any α there exists $\underline{\alpha}$ in canonical form s.t. $\alpha = \underline{\alpha}$.*

The proof in [7, Th.2.8] actually shows how to construct the canonical form of a given action in an algorithmic fashion.

In rewriting theory, to *apply an axiom* means to apply the rule obtained from directing the axiom, in our case we direct the axioms from left to right; see [32] for details on rewriting theory.

Theorem 2.5. *For a canonical form $\underline{\alpha} = +_{i\in I}\ \alpha_\times^i \cdot \underline{\alpha^i}$ only axiom (8) can be applied (and none other of the axioms of Table 1), modulo associativity and commutativity.*

Proof: Note first that we work modulo associativity and commutativity of $+$ and $\times$, and modulo associativity for $\cdot$ (thus we do not consider axioms (1), (2), (5), (10), (11)). The remaining axioms of Table 1 are considered directed from left to right.

In the rest of the proof we argue only for the first level of the canonical form because the definition is recursive. The same arguments, applied in an inductive manner, hold for the smaller subactions $\underline{\alpha^i}$ in canonical form.

Axiom (3) is not applicable because $\alpha_\times^i$ cannot be $\mathbf{0}$ and neither can $\underline{\alpha^i}$ because of Definition 2.3-b.(a). Axiom (4) is dealt with by the condition in Definition 2.3-a. The left part of axiom (6) cannot be applied because of the constraint in Definition 2.3-b.(a) which makes $\underline{\alpha^i} \neq \mathbf{1}$. The right part of (6) is not applicable because $\alpha_\times^i \neq \mathbf{1}$ when $\underline{\alpha^i}$ exists. Similar arguments using the c.(a) constraint again show that axiom (7) is not applicable. Clearly, axiom (9) is not applicable to a canonical form. Axioms (12) and (13) are dealt with by the fact that $\alpha_\times^i$ are from $\mathcal{A}_B^\times$ (contain only basic actions). Axiom (14) is taken care of by the same argument. The main purpose of the canonical form is to make sure that the axioms for $\times$ (like (15), (16), (17)) are applied exhaustively to the original action, and cannot be applied any more to the canonical form. In other words the axioms (15), (16), (17) push the $\times$ inside the action until it reaches the basic actions. $\Box$

The following corollary is immediate from the proof of Theorem 2.5 (applying axiom (8) to a canonical form breaks the canonicity).

Corollary 2.6. *The canonical form of an action α is* unique *(modulo associativity and commutativity).*

For the deontic modalities, one important notion is *action complement*. In our view action complement encodes the violation of an obligation (as we see later in Section 3). Intuitively, we say that the complement $\overline{\alpha}$ of action α is the action given by all the immediate actions which *take us outside* the tree of α. This view was aired in [10] but no formal definition was given. In our case we need to deal with synchronous actions too. The notion of action complement that we give shares ideas with [9] but it is not restricted to respect all the axioms of [9]; we want action complement to capture naturally what it means to violate an obligation in an eager system (where no idling is possible). With $\underline{\alpha}$ it is easy to formally define $\overline{\alpha}$.

Definition 2.7 (action complement). *The* action complement *is denoted by $\overline{\alpha}$ and is defined as a function $\overline{} : \mathcal{A}^{\mathcal{D}} \to \mathcal{A}^{\mathcal{D}}$ (i.e., action complement is not a principal combinator for the actions) and works on the equivalent canonical form $\underline{\alpha}$ as:*

$$\overline{\alpha} = \overline{\mathop{+}_{i\in I} \alpha^i_\times \cdot \underline{\alpha^i}} \triangleq \mathop{+}_{\beta_\times\in\overline{R}} \beta_\times + \mathop{+}_{j\in J} \left(\gamma^j_\times \cdot \overline{\mathop{+}_{i\in I'_j} \underline{\alpha^i}}\right)$$

Consider $R \triangleq \{\alpha^i_\times \mid i \in I\}$. The set $\overline{R}$ contains all the $\times$-actions $\beta_\times$ with the property that none of the actions $\alpha^i_\times$ are included in $\beta_\times$:

$$\overline{R} \triangleq \{\beta_\times \mid \beta_\times \in \mathcal{A}^\times_B \textit{ and } \forall i \in I, \alpha^i_\times \not\subseteq \beta_\times\};$$

and $\gamma^j_\times \in \mathcal{A}^\times_B$ and $\exists \alpha^i_\times \in R$ s.t. $\alpha^i_\times \subseteq \gamma^j_\times$. The indexing set $I'_j \subseteq I$ is defined for each $j \in J$ as:

$$I'_j \triangleq \{i \in I \mid \alpha^i_\times \subseteq \gamma^j_\times\}.$$

Complement of $\mathbf{1}$ is $\mathbf{0} = \overline{\mathbf{1}}$ and complement of $\mathbf{0}$ is $\mathbf{1} = \overline{\mathbf{0}}$.

The complement operation formalizes the fact that an action is not performed. In an eager system not performing an action means that some other action is performed (because the system is not allowed to idle). For a complex action this boils down to either not performing any of its immediate actions $\alpha^i_\times$, or by performing one of the immediate actions and then not performing the remaining action. Note that to perform an action $\alpha^i_\times$ means to perform any action that includes $\alpha^i_\times$. Therefore in the complement we may have actions $\gamma^j_\times$ which include more immediate actions.

Example 2.2 Consider the complex action $\alpha = a \cdot b + c \cdot d$, with $\mathcal{A}_B = \{a, b, c, d\}$, and assume to perform $\gamma^j_\times = a \times c$. At this point we need to look at both actions b and d in order to derive the complement, e.g. performing now d means that α was done, whereas performing c means that α was not done (and $a \times c \cdot c$ must be part of complement). $\square$

Our intention is to define the *minimal* action that describes the complement of some α; with the idea that the complement is an action that *immediately* takes us outside the tree of α. By minimal we refer to the number of single-step actions (or, as we define later, the *length* of the complement should be minimal; i.e., the length of its sequences). As soon as an action steps outside the tree of α there is no point in describing the rest of the actions that may follow, because no matter which these are the α still remains not done. In this way, as shown in Proposition 2.8, the action complement does not have infinite sequences of actions, and at the same time, this complement is enough to describe what it means to violate an obligation and, hence, enables us to determine where the reparations should be placed (see related details later in the semantics of the deontic modalities).

Example 2.3 We give some simple and illustrative examples for action complement. Consider $\mathcal{A}_B = \{a, b\}$, then: $\overline{a} = b$; $\overline{a \cdot b} = b + a \cdot a$; $\overline{b + a \cdot b} = a \cdot a$; $\overline{\mathbf{1} + a} = \mathbf{0}$. $\square$

The following result states that our notion of action complement always produces an action in canonical form.

Proposition 2.8. *The complement operation returns a (finitely described) deontic action which is in canonical form.*

Proof: For the first part of the proposition we prove that $\overline{\alpha}$ has no infinite application of the $+$ constructor (we say "no infinite branching") and also no infinite application of the $\cdot$ constructor (we say "no infinite depth"). In both cases we use induction on the structure of the action complement. The basis of the induction is clearly satisfied as $\mathbf{0}$, $\mathbf{1}$, and all $a \in \mathcal{A}_B$ have both finite branching and finite depth.

$\overline{R}$ is finite because $\overline{R} \subseteq \mathcal{A}_B^{\times}$, where $\mathcal{A}_B^{\times}$ is finite, and thus $+_{\beta_\times \in \overline{R}}\, \beta_\times$ is finitely branching. The indexing set J is finite (having maximum size $|\mathcal{A}_B^{\times}|$) thus $+_{j \in J}\ \gamma_\times^j$ is finitely branching. Lastly, the indexing sets I'_j are finite subsets of the I, hence $+_{i \in I'_j}\, \underline{\alpha^i}$ is a subaction of α. Thus we apply the induction hypothesis to it and deduce that its complement $\overline{+_{i \in I'_j}\, \underline{\alpha^i}}$ is finitely branching, for any $I'_j \subseteq I$. We conclude that $\overline{\alpha} = +_{\beta_\times \in \overline{R}}\, \beta_\times \ + \ +_{j \in J}\ \gamma_\times^j \cdot \overline{+_{i \in I'_j}\, \underline{\alpha^i}}$ is finitely branching.

It remains to prove that $\overline{\alpha}$ has no infinite depth. The first part of the action complement (i.e., $+_{\beta_\times \in \overline{R}}\, \beta_\times$) introduces only branches of finite depth 1. For the second part (i.e., $+_{j \in J}\ \gamma_\times^j \cdot \overline{+_{i \in I'_j}\, \underline{\alpha^i}}$) we can apply the induction hypothesis to $+_{i \in I'_j}\, \underline{\alpha^i}$ because, as we discussed before, this is a subaction of α. Thus, we have that each $\overline{+_{i \in I'_j}\, \underline{\alpha^i}}$ has finite depth. These are concatenated to the $\gamma_\times^j$ actions which have depth 1, thus, making all the second choice of finite depth, and hence the $\overline{\alpha}$ has finite depth.

For the second part of the proposition it is easy to see that the action complement respects the canonical form. Action complement is a choice of sequences, each sequence being either a single $\times$-action (i.e., from $+_{\beta_\times \in \overline{R}}\, \beta_\times$) or an $\times$-action $\gamma_\times^j$ followed by another action $\overline{+_{i \in I'_j}\, \underline{\alpha^i}}$ which we know by induction that is in canonical form. $\square$

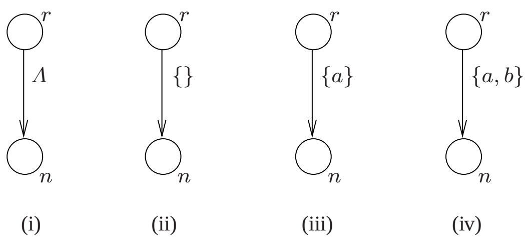

Figure 1: Trees corresponding to $\mathbf{0}$, $\mathbf{1}$, $a \in \mathcal{A}_B$, and $a \times b \in \mathcal{A}_B^{\times}$.

An important result from [7] that we need later is the interpretation of the deontic actions as rooted trees.

Definition 2.9 (rooted tree). *A* rooted tree with labeled edges *is an acyclic connected graph* $(\mathcal{N}, \mathcal{E}, \mathcal{A}_B)$ *with a designated node* r *called* root *node.* $\mathcal{N}$ *is the set of* nodes *and* $\mathcal{E}$ *is the set of* labeled edges *between nodes (in graphical notation* $n \xrightarrow{\alpha} m$*). The labels* $\alpha \in 2^{\mathcal{A}_B}$ *are* sets of basic labels*. Labels are compared for set equality (or set inclusion). Note the special* empty set *label. We consider a special label* Λ *to stand for an impossible label. We restrict our presentation to* finite *rooted trees (i.e., there is no infinite path in the graph starting from the root node). The set of all such defined trees is denoted* $\mathcal{T}$*.*

Notation: When the label of an edge is not important (i.e. it can be any label) we may use the notation $n \longrightarrow m$ instead of $n \xrightarrow{\alpha} m \,\forall \alpha \in 2^{\mathcal{A}_B}$. Each node in $\{m \,|\, n \longrightarrow m\}$ is called a *child* node of n. We denote by $|n|$ the *depth* of the node n in the tree; which is the number of edges needed to reach n from the root. A path of a tree is denoted $\sigma \in T$. A path which cannot be extended with a new edge is called *final*. The final nodes on each final path are called *leaf nodes*; denoted by $\mathit{leafs}(T) = \{n \mid n \text{ is a leaf node}\}$. The *height* of a tree, denoted $h(T)$, is the maximum of $|n|$ for all the leaf nodes n. We write $T_1 \doteq T_2$ when two trees are equal modulo renaming of the nodes (i.e. isomorphic).

Theorem 2.10 (interpretation of deontic actions [7]). *For any action* α *there exists a tree representation corresponding to the canonical form* $\underline{\alpha}$*.*

Proof: The *representation* is an interpretation function $T : \mathcal{A}^{\mathcal{D}} \to \mathcal{T}$ which interprets all actions as trees. More precisely, given an arbitrary action of $\mathcal{A}^{\mathcal{D}}$, the canonical form is computed first and then T generates the tree representation of the canonical form. Because the canonical form of an action is unique, cf. Corollary 2.6, the tree representation is indeed a function. We do not give an algorithm for computing the canonical form as one may simply apply exhaustively all the axioms excluding (8).

The function T is defined inductively, on canonical forms only. The basis of the induction is to interpret $\mathbf{0}$, $\mathbf{1}$, and each $\times$-action of $\mathcal{A}_B^{\times}$ as a tree with edges labeled from $2^{\mathcal{A}_B}$, as pictured in Fig. 1. Recall that actions of $\mathcal{A}_B^{\times} \cup \{\mathbf{0}, \mathbf{1}\}$ are in canonical form. For a general action in canonical form $\underline{\alpha} = +_{i \in I}\ \alpha_{\times}^{i} \cdot \underline{\alpha^{i}}$ the tree is generated by adding one

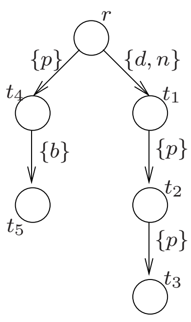

Figure 2: Tree interpretation for $p \cdot b + (d \times n) \cdot p \cdot p$.

branch to the root node for each element $\alpha^i_\times$ of the top summation operation. The label of the branch is the set $\{\alpha^i_\times\}$ corresponding to the $\times$-action. The construction continues inductively by attaching at the end of each newly added branch the tree interpretation of the smaller action $\underline{\alpha^i}$. Intuitively, $+$ provides the branching in the tree, and $\cdot$ provides the parent-child relation on each branch. □

Example 2.4 Consider the interpretation of the action $p \cdot b + (d \times n) \cdot p \cdot p$ as the tree pictured in Fig. 2. □

We now extend our grammar of actions with new constructs, namely the Kleene star and tests.

Definition 2.11 (dynamic synchronous actions). *The dynamic actions add to the deontic actions the Kleene star * operator and a set of* Boolean tests *denoted $\mathcal{A}^?$. The elements of the set $\mathcal{A}^?$ are called* tests *(or* guards*) and are included in the set of actions (i.e., tests are special actions). Tests are generated from a finite set $\mathcal{A}^?_B$ of* basic tests*. We denote tests by φ (possibly indexed) and basic tests by ϕ. The dynamic actions are constructed with the grammar below:*

$$\delta \triangleq a \mid \varphi? \mid \delta + \delta \mid \delta \cdot \delta \mid \delta \times \delta \mid \delta^* \qquad \textit{(dynamic actions)}$$
$$\varphi \triangleq \phi \mid \mathbf{0} \mid \mathbf{1} \mid \varphi + \varphi \mid \varphi \cdot \varphi \mid \varphi \times \varphi \mid \neg\varphi \qquad \textit{(tests)}$$

Note the overloading of the operators $+, \cdot, \times$, and constants $\mathbf{0}, \mathbf{1}$: over arbitrary actions they have the meaning as before, whereas, over tests they take the meaning of the well known disjunction (for $+$), conjunction (for $\cdot$ and $\times$), falsity and truth (for $\mathbf{0}$ and $\mathbf{1}$). The behavior over tests is given by the axioms that are required, which are those of Boolean algebra. In particular, $\cdot$ and $\times$ respect different axioms when applied to normal actions, though in the case of tests they turn out to have the same behavior, acting both as a Boolean conjunction. This is clear for $\times$, and probably less intuitive for $\cdot$. The reason for the latter behaving as a conjunction is that checking two simple tests in sequence is independent on which one is tested first, so the result is the same as checking their conjunction (see details in [7]).

In [7] it was shown how it can be associated to any dynamic synchronous action an *automaton on guarded synchronous strings*; this is similar to the interpretation as trees that we gave for deontic actions. The completeness result of [7] ensures that working with the dynamic synchronous actions or with the automata on guarded synchronous strings is the same (they are interchangeable). This result can be instantiated to deontic actions to ensure that working with the deontic actions or with their tree representations is the same.

Definition 2.12 (guarded synchronous strings). *Over the set of basic tests $\mathcal{A}_B^?$ we define* atoms *as functions $\nu : \mathcal{A}_B^? \to \{0,1\}$ assigning a Boolean value to each basic test. Consider the finite alphabet $\Sigma = 2^{\mathcal{A}_B} \setminus \{\emptyset\}$ of all nonempty subsets of basic actions (denoted x, y). A* guarded synchronous string *(denoted by u, v, w) is a sequence*

$$w = \nu_0 x_1 \nu_1 \dots x_n \nu_n, \quad n \geq 0,$$

where ν_i are atoms. We define $\mathit{first}(w) = \nu_0$ and $\mathit{last}(w) = \nu_n$. Denote by $Atoms = \{0,1\}^{\mathcal{A}_B^?}$ the set of all atoms ν. We call a synchronous string *a guarded synchronous string stripped of all the atoms ν_i (i.e., the synchronous string associated with the w above is just $x_1 \dots x_n$).*

Proposition 2.13 (automata for guards [7]). *An* automaton $A = (S, \Sigma, S_0, \rho, F)$ *consists of a finite set of states S together with a transition relation ρ between these states which is labeled by letters from the alphabet Σ. An automaton accepts sets of strings (also called the language of words accepted by the automaton) where each accepting string is the sequence of labels coming from a sequence of transitions where the first transition starts in an initial state from S_0 and the last one ends in a final state from F.*

For the set of $Atoms$ there exists a class of finite state automata which accept all and only the subsets of atoms. We denote the set of all such automata by $\mathcal{M}$ and one automaton by $M \in \mathcal{M}$.

We can now give the representation of the dynamic synchronous actions as the two-level hierarchical automata defined below.

Definition 2.14 (automata on guarded synchronous strings). *Consider a two level finite automaton $A^{\mathcal{G}} = (S, \Sigma, S_0, \rho, F, \lceil\cdot\rceil)$. It consists at the first level of a finite automaton on synchronous strings $(S, \Sigma, S_0, \rho, F)$, together with a map $\lceil\cdot\rceil : S \to \mathcal{M}$. An automaton on synchronous strings consists of a finite set of* states *S, the finite* alphabet *$\Sigma = 2^{\mathcal{A}_B} \setminus \{\emptyset\}$, a set of* initial *designated states $S_0 \subseteq S$, a* transition relation *$\rho : \Sigma \to S \times S$, and a set of* final *states F. The mapping $\lceil\cdot\rceil$ associates with each state on the first level an automaton $M \in \mathcal{M}$ as defined in Proposition 2.13 which accepts atoms. The automata in the states make the second (lower) level. Denote the language of atoms accepted by $\lceil s \rceil$ with $\mathcal{L}(\lceil s \rceil)$.*

Theorem 2.15 (interpretation of dynamic actions [7]). *For any dynamic action δ there is a corresponding automaton $A^{\mathcal{G}}(\delta)$ which accepts the same set of guarded synchronous strings that δ describes.*

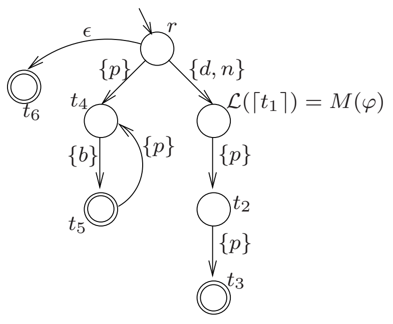

Figure 3: Automaton on guarded synchronous strings for $(p \cdot b)^* + (d \times n) \cdot \varphi? \cdot p \cdot p$. The picture omits $\mathcal{L}(\lceil t_i \rceil) = M(Atoms)$ when $i \in \{2,3,4,5,6\}$.

Example 2.5 Let us consider the automaton $A^{\mathcal{G}}(\delta)$ for the dynamic synchronous action $\delta = (p \cdot b)^* + (d \times n) \cdot \varphi? \cdot p \cdot p$, depicted in Fig. 3. The $A^{\mathcal{G}}(\delta)$ automaton has in state t_1 (on the second level) an automaton $M(\varphi)$ corresponding to the test φ which accepts all and only the atoms that constitute a satisfying interpretation for φ. In all the other states of $A^{\mathcal{G}}(\delta)$ we have the trivial automaton for $\top$ that accepts any atom. □

3. Deontic Modalities over Synchronous Actions

In this section we introduce the deontic part of the $\mathcal{CL}$ logic and we work only with deontic modalities over synchronous actions (and the underlying propositional language).

Definition 3.1. *The deontic expressions of the $\mathcal{CL}$ logic are constructed by the grammar below:*

$$\begin{array}{lcll} \mathcal{C} & := & \phi \mid O_{\mathcal{C}}(\alpha) \mid P(\alpha) \mid F_{\mathcal{C}}(\alpha) \mid \mathcal{C} \to \mathcal{C} \mid \bot & \textit{(deontic formulas)} \\ \alpha & := & a \mid \mathbf{0} \mid \mathbf{1} \mid \alpha \times \alpha \mid \alpha \cdot \alpha \mid \alpha + \alpha & \textit{(deontic actions)} \end{array}$$

We call an expression $\mathcal{C}$ a (general) *contract clause*. A contract clause is built using the classical propositional implication operator $\to$, where the other operators $\wedge, \vee, \neg, \leftrightarrow, \top, \oplus$ (exclusive or) are expressed in terms of $\to$ and $\bot$ as in propositional logic. The building blocks of a contract clause are the propositional constants ϕ drawn from a finite set Φ_B and the deontic modalities $O_{\mathcal{C}}(\alpha)$, $P(\alpha)$, and $F_{\mathcal{C}}(\alpha)$.[2] These represent respectively the obligation, permission, and prohibition of performing a given *action* α. Intuitively $O_{\mathcal{C}}(\alpha)$ states the obligation to perform α, and the *reparation* $\mathcal{C}$ in case the obligation is *violated*, i.e., whenever α is not performed. The reparation may

[2]For this article, where the examples of $\mathcal{CL}$ formulas are not big, the notation $O_{\mathcal{C}}$ and $F_{\mathcal{C}}$ is pleasing and intuitive; but in the bigger examples these may be cluttering and undesired. In such a case a user may choose a different notation, like $O(\alpha)\lceil\mathcal{C}$, and an example like $O_{F_{\mathcal{C}_1}(\beta)\wedge\mathcal{C}_2}(\alpha)$ would look nicer as $O(\alpha)\lceil((F(\beta)\lceil\mathcal{C}_1) \wedge \mathcal{C}_2)$.

be any contract clause. The modality $O_{\mathcal{C}}(\alpha)$ (resp. $F_{\mathcal{C}}(\alpha)$) represents what is called CTD (resp. CTP) in dynamic deontic logic. Obligations without reparations are written as $O_{\perp}(\alpha)$ where $\perp$ (and conversely $\top$) is the Boolean $false$ (respectively $true$). We usually write $O(\alpha)$ instead of $O_{\perp}(\alpha)$. Obligations with no reparation are sometimes in the literature called *categorical* because they must not be violated (i.e., there is no reparation for their violation, thus a violation would give violation of the whole contract). The prohibition modality $F_{\mathcal{C}}(\alpha)$ states that the action α is forbidden and in case the prohibition is violated the reparation $\mathcal{C}$ is enforced. Note that it is possible to express nested CTDs and CTPs. Permissions have no reparations associated because they cannot be violated; permissions can only be exercised. The logical expressions of $\mathcal{CL}$ are interpreted over Kripke-like structures which we call *normative structures*.

Definition 3.2 (normative structure). *A* normative structure *is* $K^{\mathcal{N}}=(\mathcal{W}, R_{2^{\mathcal{A}_B}}, \mathcal{V}, \varrho)$ *where:*

- $\mathcal{W}$ *is a set of* worlds *(also called states);*
- $2^{\mathcal{A}_B}$ *contains the* labels *of the structure as sets of basic actions from the finite set* $\mathcal{A}_B$*.* $R_{2^{\mathcal{A}_B}} : 2^{\mathcal{A}_B} \to 2^{\mathcal{W}\times\mathcal{W}}$ *returns for each label a* partial function *on the set of worlds (written as a relation);*
- $\mathcal{V} : \Phi_B \to 2^{\mathcal{W}}$ *is a* valuation function *of the propositional constants returning a set of worlds where the constant holds;*
- $\varrho : \mathcal{W} \to 2^{\Psi}$ *is a marking function which marks each world with markers from* $\Psi = \{\circ_a, \bullet_a \mid a \in \mathcal{A}_B\}$*. The marking function respects the restriction that no world can be marked by both* $\circ_a$ *and* $\bullet_a$*, for any* $a \in \mathcal{A}_B$*.*

A pointed normative structure *is a normative structure with a designated world* i *(denoted by* $\langle K^{\mathcal{N}}, i\rangle$*).*

We denote by an indexed t a node of a tree (or by r the root) and by an indexed s (or i for initial) a state of a normative structure. We use the graphical notation $s \xrightarrow{\alpha} s'$ for the transitions of the normative structures too. We sometimes abuse the notation by writing $s \in K^{\mathcal{N}}$ for $s \in \mathcal{W}$ of $K^{\mathcal{N}}$, and $s \xrightarrow{\alpha} s' \in K^{\mathcal{N}}$ for $s \xrightarrow{\alpha} s' \in R_{2^{\mathcal{A}_B}}$ of $K^{\mathcal{N}}$. Note that we consider both the tree from before and the normative structures to have the same set of basic labels $\mathcal{A}_B$.

$K^{\mathcal{N}}$ is *deterministic* as for each label from one world there is at most one successor world; i.e., the partial function requirement. We use deterministic structures because in the deontic realm, as in legal contracts, each action (like "deposit 100$ in bank account") must have a well determined behavior (i.e., the actions do not have a nondeterministic outcome). The deterministic restriction of Kripke structures was investigated in [20]. The marking function and the markers are needed to identify obligatory (i.e., $\circ$) and prohibited (i.e., $\bullet$) actions. Markers with different purposes were used in [9] to identify violations of obligations, in [15] to mark permitted transitions, and in [16] to identify permitted events.

As an example let us consider the normative structure in Fig. 4, which has five states and two constant propositions $\Phi_B = \{\phi, \phi'\}$. The valuation function assigns to each proposition a set of states, e.g., $\mathcal{V}(\phi) = \{s_1, s_3, s_5\}$.

A first difference between normative structures and the standard Kripke structures is that the labels in normative structures can be sets; in Fig. 4 there is one transition labeled by $\{d, n\}$ (i.e., $R_{2^{\mathcal{A}_B}}(\{d, n\}) = \{(s_1, s_2)\}$) and two transitions labeled $\{p\}$ (i.e.,

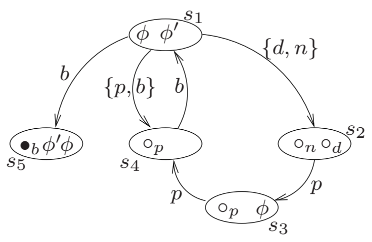

Figure 4: Example of a normative structure.

$R_{2^{\mathcal{A}_B}}(\{p\}) = \{(s_2, s_3), (s_3, s_4)\}$). The second difference is that normative structures have a marking function, and in our example it marks s_2 by $\{\circ_n, \circ_d\}$, s_5 by $\{\bullet_b\}$, and s_3, s_4 by $\{\circ_p\}$ each.

A normative structure has essentially two components: the labeled transitions (related to the actions that can be performed from each state) and the marking function (related to obligations and prohibitions). Without the marking function we are in the known setting of Kripke structures where an action logic like the propositional dynamic logic PDL is what we need to talk about the labeled transitions, i.e., about the actions and their regular structure. This is enough to talk about programs and their execution, nondeterminism, or loops.

In order to talk about which actions are obligatory and which are forbidden we introduce the marking function. This decorates the states of the Kripke structure with markers, giving rise to what we called normative structure.[3] If one needs to talk only about executions of actions, then the markers are superfluous. But if one needs to talk about which executions are respecting and which are violating some obligation of a complex action, then the markers come into play. In fact, the purpose of normative structures is to give a representation to a contract (i.e., to obligations and prohibitions), and this is what we investigate in this paper. Talking about which traces of actions respect or violate a contract was done in [5] and that work should have formal connections with the present, but these are still work in progress. (See [6, Sec.4.4.1] for preliminary results in this direction.)

Remarks: the markers of Ψ can be seen as special propositional constants, i.e., $\Psi \subset \Phi_B$, and the marking function as part of the valuation function. Our choice is to separate these, as the markers have different purpose from the propositional constants and the valuation. The purpose of the valuation is to represent the outcome of the actions, whereas the purpose of the markers is to represent the deontic content of the actions (i.e., which actions are obligatory, permitted, or forbidden). Moreover, the presentation of the semantics of $\mathcal{CL}$ is more clean.

In order to relate the semantic domain of our language with the underlying algebra of actions on which the deontic modalities operate on, we will give a formal relation-

[3]To be precise, a normative structure is also deterministic, whereas a general Kripke structure may not.

ship between normative structures and trees through a notion of simulation.

Definition 3.3 (simulation). *For a tree $T = (\mathcal{N}, \mathcal{E}, \mathcal{A}_B)$ and a normative structure $K^{\mathcal{N}} = (\mathcal{W}, R_{2^{\mathcal{A}_B}}, \mathcal{V}, \varrho)$ we define a relation $\mathcal{S} \subseteq \mathcal{N} \times \mathcal{W}$ which we call the* simulation *of the tree node by the state of the structure:*

$$t\,\mathcal{S}\,s \text{ iff } \forall t \xrightarrow{\gamma} t' \in T, \exists s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}} \text{ s.t. } \gamma \subseteq \gamma' \wedge t'\mathcal{S}s', \text{ and}$$
$$\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}} \text{ with } \gamma \subseteq \gamma' \text{ implies } t'\,\mathcal{S}\,s'.$$

We say that a tree T, with root r is simulated by a normative structure $K^{\mathcal{N}}$ w.r.t. a state s, denoted $T\ \mathcal{S}_s\ K^{\mathcal{N}}$, if and only if $r\,\mathcal{S}\,s$.

Note two differences with the classical definition of simulation: first, the labels of the normative structure may be bigger (a superset) than the labels in the tree. This can be intuitively motivated by the idea that respecting an obligatory action means executing an action which includes it (is bigger). We can drop this condition and consider only $\gamma = \gamma'$, in which case we call the relation *strong simulation* and denote by $\mathcal{S}^s$. The second difference is that any transition in the normative structure that can simulate an edge in the tree must enter under the simulation relation. This is because from the state s' onwards we need to be able to continue to look in the structure for the remaining tree (to see that it is simulated). Trivially, any strong simulation relation is also a simulation relation. We can weaken the definition of simulation by combining the two conditions into: $\forall t \xrightarrow{\gamma} t' \in T, \forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ with $\gamma \subseteq \gamma'$ then $t'\,\tilde{\mathcal{S}}\,s'$. We call the resulting relation *partial simulation* and denote it by $\tilde{\mathcal{S}}$.

Example 3.1 Consider the tree from Fig. 2 and denote it by $T(\alpha)$ where $\alpha = p \cdot b + (d \times n) \cdot p \cdot p$. This tree is simulated by the normative structure $K^{\mathcal{N}}$ of Fig. 4 w.r.t. the state s_1. It is easy to check that $r\,\mathcal{S}\,s_1$: for the edge $r \xrightarrow{\{p\}} t_4$ we find in $K^{\mathcal{N}}$ the transition $s_1 \xrightarrow{\{p,b\}} s_4$ that respects the inclusion of labels. We also have that $t_4\,\mathcal{S}\,s_4$ since for the only edge $t_4 \xrightarrow{\{b\}} t_5$ there is the transition $s_4 \xrightarrow{\{b\}} s_1$ in $K^{\mathcal{N}}$ simulating it (the second constraint from the definition of simulation is satisfied trivially for $t_4 \xrightarrow{\{b\}} t_5$ because there is no other transition from s_4 in $K^{\mathcal{N}}$). Moreover, for the edge $r \xrightarrow{\{p\}} t_4$ the second condition for simulation is satisfied because there is no other transition from s_1 with a label that includes $\{p\}$. For the second edge $r \xrightarrow{\{d,n\}} t_1$ we find the transition $s_1 \xrightarrow{\{d,n\}} s_2$ in $K^{\mathcal{N}}$ that respects all the simulation conditions (checking these is done as before). If we were to change the label of the transition (s_1, s_4) to $\{p\}$ then the tree would still be simulated but it would also be *strongly* simulated.

Consider a simpler example of a tree $T(b)$ interpreting the basic action b. This too is strongly simulated by the structure $K^{\mathcal{N}}$ w.r.t. s_1 because, for the only edge of the tree labeled with $\{b\}$ we find the transition $s_1 \xrightarrow{\{b\}} s_5$ which *has exactly the same label*, and the leaf node of the tree is trivially simulated by s_5 because there are no edges out of it (note that this holds for any leaf node of a tree). On top, the second simulation condition also holds because the only other transition that has a label that includes b is $s_1 \xrightarrow{\{p,b\}} s_4$ and for this transition s_4 trivially simulates the leaf node. $\square$

Given a tree that is simulated by a normative structure, we can isolate a maximal substructure of the latter that simulates the given tree. As we will see later, it is also useful to identify what parts of the normative structure do not simulate the tree, giving place to what we call the non-simulating remainder of the structure. These notions are formally defined in Definition 3.4 and 3.5.

Definition 3.4 (maximal simulating structure). *For a tree T that is simulated by a normative structure $K^{\mathcal{N}}$ w.r.t. the state i (in notation $T\ \mathcal{S}_i\ K^{\mathcal{N}}$), we denote by $K^{T,i}_{max} = (\mathcal{W}', R'_{2^{\mathcal{A}_B}}, \mathcal{V}', \varrho')$ and call* the maximal simulating structure w.r.t. T and i of $K^{\mathcal{N}}$ *the maximal sub-structure of $K^{\mathcal{N}}$ respecting the following, where $K^{\mathcal{N}} = (\mathcal{W}, R_{2^{\mathcal{A}_B}}, \mathcal{V}, \varrho)$:*

a. $i \in \mathcal{W}'$, $\mathcal{V}' = \mathcal{V}|_{\mathcal{W}'}$ *and* $\varrho' = \varrho|_{\mathcal{W}'}$

b. $\forall t \xrightarrow{\gamma} t' \in T$ *then* $\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ *s.t.* $t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma' \wedge t'\,\mathcal{S}\,s'$

$$s' \in \mathcal{W}' \text{ and } s \xrightarrow{\gamma'} s' \in R'_{2^{\mathcal{A}_B}}.^{4}$$

Definition 3.5 (non-simulating remainder). *We call the* non-simulating remainder of $K^{\mathcal{N}}$ w.r.t. T and i *the sub-structure $K^{T,i}_{rem} = (\mathcal{W}'', R''_{2^{\mathcal{A}_B}}, \mathcal{V}'', \varrho'')$ of $K^{\mathcal{N}}$ that is maximal and respects the following:*

a. $s \xrightarrow{\gamma} s'' \in R''_{2^{\mathcal{A}_B}}$ *iff* $s \xrightarrow{\gamma} s'' \notin K^{T,i}_{max} \wedge s \in K^{T,i}_{max} \wedge \exists s \xrightarrow{\gamma'} s' \in K^{T,i}_{max}$,

b. $s \in \mathcal{W}''$ *iff s is part of a transition in* $R''_{2^{\mathcal{A}_B}}$,

c. $\mathcal{V}'' = \mathcal{V}|_{\mathcal{W}''}$, *and* $\varrho'' = \varrho|_{\mathcal{W}''}$.

For the two formal definitions above consider the following example.

Example 3.2 Take the simple action b with the trivial tree $T(b)$ which has only one edge $r \xrightarrow{\{b\}} t_1$. We have discussed above that this tree is strongly simulated by the structure w.r.t. node s_1 of the normative structure in Fig. 4. The maximal simulating structure $K^{T(b),s_1}_{max}$ is the substructure obtained from $K^{\mathcal{N}}$ by deleting the states s_2 and s_3 (and the associated transitions too) as well as the transition $s_4 \xrightarrow{b} s_1$. The non-simulating remainder structure is obtained from $K^{\mathcal{N}}$ by deleting the worlds s_3, s_4, and s_5. For the more complex action from Fig. 2 the non-simulating remainder is the substructure that has only the worlds s_1 and s_5 and the transition between them. □

Intuitively, the maximal simulating structure captures all and only those transitions from the initial normative structure that enter in the simulation relation w.r.t. a tree. The non-simulating remainder structure captures all those transitions that do not enter the simulation relation but that somehow relate to (or simulate) the complement of the tree (or only that part of the complement tree that is in the normative structure that we talk about). Anticipating the semantics, the maximal simulating structure is used for marking states with $\circ$ marks, whereas the non-simulating remainder is used to determine which state should necessarily not be marked. In few words, the maximal simulating structure has all and only those transitions from the normative structure that

[4]Note that $t'\,\mathcal{S}\,s'$ actually follows from the first two requirements ($t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma'$). We mention it in the definition for the convenience of the reader, because it is important to keep in mind that the states that the definition recursively visits are all simulating some node in the tree.

$$\begin{array}{lll}
K^{\mathcal{N}}, i \models \varphi & \text{iff} & i \in \mathcal{V}(\varphi). \\
K^{\mathcal{N}}, i \not\models \bot & & \\
K^{\mathcal{N}}, i \models \mathcal{C}_1 \to \mathcal{C}_2 & \text{iff} & \text{whenever } K^{\mathcal{N}}, i \models \mathcal{C}_1 \text{ then } K^{\mathcal{N}}, i \models \mathcal{C}_2. \\
K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) & \text{iff} & T(\alpha)\ \mathcal{S}_i\ K^{\mathcal{N}}, \quad \text{and} \\
 & & \quad \forall t \xrightarrow{\gamma} t' \in T(\alpha), \forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}} \text{ s.t. } t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma', \\
 & & \qquad \forall a \in \mathcal{A}_B \text{ if } a \in \gamma \text{ then } \circ_a \in \varrho(s'), \quad \text{and} \\
 & & \quad \forall s \xrightarrow{\gamma'} s' \in K^{T(\alpha),i}_{rem}, \\
 & & \qquad \forall a \in \mathcal{A}_B \text{ if } a \in \gamma' \text{ then } \circ_a \notin \varrho(s'), \quad \text{and} \\
 & & \quad K^{\mathcal{N}}, s \models \mathcal{C} \quad \forall s \in K^{\mathcal{N}} \text{ with } t\,\mathcal{S}^s\,s \wedge t \in \mathit{leafs}(T(\overline{\alpha})). \\
K^{\mathcal{N}}, i \models F_{\mathcal{C}}(\alpha) & \text{iff} & T(\alpha)\ \tilde{\mathcal{S}}_i\ K^{\mathcal{N}} \text{ then} \\
 & & \quad \forall \sigma \in T(\alpha) \text{ a final path s.t. } \sigma\ \mathcal{S}_i\ K^{\mathcal{N}}, \\
 & & \quad \forall t \xrightarrow{\gamma} t' \in \sigma, \forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}} \text{ with } t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma', \\
 & & \qquad \forall a \in \mathcal{A}_B \text{ if } a \in \gamma' \text{ then } \bullet_a \in \varrho(s') \quad \text{and} \\
 & & \quad K^{\mathcal{N}}, s \models \mathcal{C} \quad \forall s \in K^{\mathcal{N}} \text{ with } t\,\mathcal{S}\,s \wedge t \in \mathit{leafs}(\sigma). \\
K^{\mathcal{N}}, i \models P(\alpha) & \text{iff} & T(\alpha)\ \mathcal{S}_i\ K^{\mathcal{N}}, \text{ and} \\
 & & \quad \forall t \xrightarrow{\gamma} t' \in T(\alpha), \forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}} \text{ s.t. } t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma', \\
 & & \qquad \forall a \in \mathcal{A}_B \text{ if } a \in \gamma \text{ then } \bullet_a \notin \varrho(s').
\end{array}$$

Table 2: Semantics for the deontic modalities over synchronous actions.

enter the simulation relation. The non-simulating remainder has those transitions that did not enter in the simulation relation but which are directly connected to the maximal simulating structure.

We have now all the necessary definitions to introduce the semantics of our deontic modalities over synchronous actions.

Definition 3.6 (semantics). *We give in Table 2 a recursive definition of the* satisfaction relation $\models$ *of a formula $\mathcal{C}$ w.r.t. a pointed normative structure $\langle K^{\mathcal{N}}, i\rangle$; it is written $K^{\mathcal{N}}, i \models \mathcal{C}$ and is read as "$\mathcal{C}$ is* satisfied *in the normative structure $K^{\mathcal{N}}$ at state i". We write $K^{\mathcal{N}}, i \not\models \mathcal{C}$ whenever $K^{\mathcal{N}}, i \models \mathcal{C}$ is not the case. We say that "$\mathcal{C}$ is* globally satisfied *in $K^{\mathcal{N}}$", and write $K^{\mathcal{N}} \models \mathcal{C}$ iff $\forall s \in K^{\mathcal{N}}$, $K^{\mathcal{N}}, s \models \mathcal{C}$. A formula is* satisfiable *iff $\exists K^{\mathcal{N}}, \exists s \in K^{\mathcal{N}}$ s.t. $K^{\mathcal{N}}, s \models \mathcal{C}$. A formula is* valid *(denoted $\models \mathcal{C}$) iff $\forall K^{\mathcal{N}}, K^{\mathcal{N}} \models \mathcal{C}$.*

The propositional connectives have the classical semantics. More interesting and particular to our logic is the interpretation of the deontic modalities.

For $O_{\mathcal{C}}$ the semantics has basically two parts:

- The first part of the semantics is the interpretation of the obligation.
  - The first line says how to walk on the structure depending on the tree of the action α. The test for simulation must succeed, which means that all the tree of the action is found in the structure also. The simulation relation is used because in the structure there may be transitions labeled with actions that are greater than the actions in α, which intuitively, if we do these

actions then the obligation of α is still respected. The simulation relation also takes care that all the choices of an action appear as transitions in the structure. We cannot specify an obligation of an action that does not appear as the label of some transition in the normative structure; this would mean that the action is forbidden (as we see further).

The simulation relation makes sure that in the normative structure however one would choose some transition that has label respecting the action, from the state reached the simulation of the rest of the action can continue (this is from the second condition in the Definition 3.3 of simulation). We do not require only that it exists some such choice of transition (i.e., only the first condition of Definition 3.3), but that with any such good choice one can continue to simulate the rest of the action.

– The second and third lines mark all the transitions (their ending states) of the structure which simulate edges from the tree with markers $\circ_a$ corresponding to the labels of the simulated edge. This is needed both for the proof of the synchrony property $O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta) \rightarrow O_{\mathcal{C}}(\alpha \times \beta)$ and also in proving $O_{\mathcal{C}}(\alpha) \rightarrow \neg F_{\mathcal{C}}(\alpha)$ which relates obligations and prohibitions.

  In this part of the semantics we work in the maximal simulating structure. This part is where is checked that the transitions are actually obligatory (or precisely, part of their labels, because we are interested only in the labels of the tree, which may be smaller than those in the maximal simulating structure). Intuitively, we require that all simulating transitions be labeled by $\circ$ markers because otherwise it would mean that there are ways to do the next step of the action which the normative structure sees as not being obligatory, but they could even be forbidden.

– Lines four and five ensure that no other reachable relevant transitions of the structure (i.e., from the non-simulating remainder structure) are marked with obligation markers $\circ$. This is essential in the proof of the key Lemma A.1 (see appendix)[5] of the synchrony result given in Theorem 3.21.

  Because of this last requirement we can have several unwanted implications (as are called later in Proposition 3.25) an example being that the obligation of a choice of actions does not imply that one of the choices is obligatory alone (here this part of the semantics plays a crucial role).

  Intuitively we have only one single obligation of a big complex action and all the transitions in the normative structure that simulate the tree of this big action are marked according to the lines two and three from before. Lines four and five say that anything else outside this big action should not be marked with $\circ$ markers. From the unwanted implications of Proposition 3.25 one can see that there is little compositionality when it comes to obligations. A compositional result for $\mathcal{CL}$ is that of Theorem 3.21 which puts together two contract clauses, each specifying the obligation of some

[5]All lemmas and corollaries starting with 'A' are auxiliary results presented (together with their proofs) in the appendix.

action, into one single obligation of a bigger action that comes from the synchronous composition of the two smaller actions (i.e., the two actions must be done at the same time).

- The second part is just the last line and states that if the obligation is violated (i.e., the complement $\overline{\alpha}$ of the action is performed) then the reparation $\mathcal{C}$ should hold at every possible violating state. This part of the definition is similar to the definition of the box modality of PDL only that here it is applied to the complement of the action. We look at the leafs of the tree $T(\overline{\alpha})$ and because the $\overline{\alpha}$ contains all possible violating actions it is enough to use strong simulation and thus look in the normative structure for exactly the same labels as the ones in the tree of the complement action).

For the $F_{\mathcal{C}}$ modality:

- We use partial simulation $\tilde{\mathcal{S}}_i$ in order to have our intuition that if an action is not present as a label of a transition of the model then the action is *by default* considered forbidden.

- In the second line we consider *all* final paths in order to respect the intuition that prohibition of a choice must prohibit all, i.e., $F(a + b) = F(a) \wedge F(b)$. Note that we are interested only in *final* paths simulated by the structure because for the other paths some of the transitions are missing in the structure and thus there is some action on the sequence which is forbidden.

- In the third line we consider all the edges on each final path in order to respect the intuition that forbidding a sequence means forbidding all the actions on that sequence. For a chosen edge we look for *all* the transitions of the normative structure from the chosen node which have a label *greater* than the label of the edge; this is in order to respect the intuition that forbidding an action implies forbidding any action that is greater, i.e., $F(a) \rightarrow F(a \times b)$. For the same reason we need to mark with $\bullet$ markers corresponding to the label of the transition and not to the label of the edge of the tree, as we do for obligations.

- The last line states that if the prohibition is violated then the reparation $\mathcal{C}$ must hold in all the states where the violation is observed. A violation of a prohibition is observed at the states simulating the leaf nodes of the final paths that we consider. This is because in these states it means that we executed the whole complex action α that was prohibited (precisely, one of its branches).

The semantics of P specifies that $\bullet$ markers should not be present in order to capture the principle that *what is not forbidden is permitted*. The semantics of O, P, or F hint at the trace-based semantics of Process Logic [33] and to some extent to the modalities of [15].

Note that we need two kinds of markers: $\circ$ and $\bullet$. The $\bullet$ marker is for determining which actions are prohibited. The missing of the $\bullet$ marker is for permissions. Whereas the $\circ$ marker (which requires that the forbid marker is not present) is for obligations.

Example 3.3 Consider the normative structure from Fig. 4. In state s_3 the obligation $O(p)$[6] holds. In fact we could have any reparation because there is no other transition represented in this normative structure, and therefore, the last condition in the semantics of O is trivially satisfied (i.e., $O_\top(p)$ and $O_\mathcal{C}(p)$ also hold for any contract clause $\mathcal{C}$). But the fact that even $O_\bot(p)$ holds tells us that in this state there is a categorical obligation. This showed that a particular state is characterized by all the formulas that hold at that state.

In the same structure at state s_1, $O_\top(p)$ does not hold (i.e., $s_1 \not\models O_\top(p)$) because there are other outgoing transitions from s_1 that are marked with $\circ$. For the same reason $s_1 \not\models O_\top(d\times n)$. But also $s_1 \not\models O_\top(p\times b)$ because the $\circ_b$ is not marking state s_4. In state s_1 we actually have an obligation of a choice: $s_1 \models O_\top(p + d\times n)$. There are other more informative formulas that hold in s_1, formulas that talk about the possible reparations in case of violating this obligation: $s_1 \models O_{(\phi\wedge\phi')}(p + d\times n)$.

In the same state s_1 we have also a prohibition: $F_\top(b)$ because state s_5 is marked by $\bullet_b$. Actually the same reparation as before can be attached to this prohibition also: $s_1 \models F_{(\phi\wedge\phi')}(b)$.

3.1. Decidability for deontic modalities

We prove that the deontic modalities over synchronous actions have the *tree model property*. It was argued [34, 35] that the tree model property is an more basic property of many modal logics than decidability is. Decidability proofs for modal logics are often based on a tree model property; and this is the method that we use in this section. There are several ways of proving decidability starting from a tree model property. In this section we use the *selection* method. Moreover, there are modal logics that have the tree model property but are not decidable. In Section 4 we show that the general $\mathcal{CL}$ logic has the tree model property, but we do not manage to show full decidability.

The road-map for this section is to show first that the deontic modalities as defined above have the tree model property (in Corollary 3.10). Then we use the method of *selection* to show that this tree can be pruned in such a way that we are left with a finite tree (cf. Lemma 3.15) that is related to the initial structure and formula in the way that if the formula holds in the initial structure at some node then it will also hold in the root of this finite tree (i.e., Theorem 3.9). From this we get the result of decidability of satisfiability for the deontic modalities of $\mathcal{CL}$; i.e., we can check if a formula is satisfiable, and this amounts to checking all such finite trees related to the formula in question.

Intuitively, the tree model property says that instead of working with arbitrary Kripke-like models, which have a graph structure, it is enough to work with models that have a tree-like structure. Trees are much simpler and well behaved structures than graphs are. Moreover, there are well established techniques for modal logics and trees.

Definition 3.7. *A* pointed tree structure $\langle TK^\mathcal{N}, \varepsilon\rangle = (\mathcal{W}^T, R^T_{2^{\mathcal{A}_B}}, \mathcal{V}^T, \varrho^T)$*, is a pointed normative structure* $\langle K^\mathcal{N}, s\rangle$ *satisfying the restrictions of Definition 2.9 and:*

a) the nodes are characterized by strings over natural numbers $\mathcal{W}^T \subset \mathbb{N}^*$*, with* $s = \varepsilon$*;*

[6] Recall that we omit the reparation when this one is just $\bot$.

b) for each label $\alpha \in 2^{\mathcal{A}_B}$ the partial function $R^T_{2^{\mathcal{A}_B}}(\alpha) : \mathcal{W}^T \to \mathcal{W}^T$ respects the restriction: $R^T_{2^{\mathcal{A}_B}}(\alpha)(x) = xn$ where $x, xn \in \mathcal{W}^T$ and $n \in \mathbb{N}$;

c) for any $\alpha \neq \beta \in 2^{\mathcal{A}_B}$ then $R^T_{2^{\mathcal{A}_B}}(\alpha)(x) \neq R^T_{2^{\mathcal{A}_B}}(\beta)(x)$ for any $x \in \mathcal{W}^T$.

Condition a) above labels each state with a (unique) string over natural numbers. Condition b) guarantees that the structure contains no cycles, and c) that two transitions starting in the same state and with different labels (actions) go to different states. The definition is for standard tree structures (with labels on their edges) but the notation used is adapted to the proofs below (the notation is taken from similar proofs of decidability, e.g. in [17, 36]).

The following lemma shows that it is always possible to obtain a pointed tree structure from a given pointed normative structure (it is an adaptation of the standard unfolding construction).

Lemma 3.8 (tree model). *Given a pointed normative structure $\langle K^{\mathcal{N}}, i\rangle$ we can construct an associated tree structure $\langle TK^{\mathcal{N}}, \varepsilon\rangle$.*

Proof: The technique that we use is known in modal logics as the tree unfolding of a Kripke structure [37, 17]. For a pointed normative structure $\langle K^{\mathcal{N}}, i\rangle = (\mathcal{W}, R_{2^{\mathcal{A}_B}}, \mathcal{V}, \varrho)$ we can view the set of worlds $\mathcal{W} = \{0, 1, 2, \ldots\}$ to be the natural numbers $\mathbb{N}$; and we define the set $\mathcal{W}^T[i] \subset \mathbb{N}^*$ to be the set of finite paths starting from i. Moreover, we enrich the paths to contain also the labels by which the path was formed. For this we interplace between the nodes labels from $2^{\mathcal{A}_B}$. More precisely, i is considered the empty string ε, the paths of depth one are $\varepsilon\alpha s$ such that $s \in \mathcal{W}$ and $i \xrightarrow{\alpha} s$ is a transition in $K^{\mathcal{N}}$. We define a function $\rho : \mathcal{W}^T[i] \to \mathcal{W}$ which assigns to each path the state in which the path ends; e.g. $\rho(\varepsilon\alpha s'\beta s'') = s''$. Note that two paths $x\alpha s$ and $x\beta s$ are regarded as different. Consider the set $\mathcal{W}[i] = \{\rho(x) \mid x \in \mathcal{W}^T[i]\}$ of states reachable (by any path) from the node i. The function $\rho : \mathcal{W}^T[i] \to \mathcal{W}[i]$ is a surjection therefore it exists the corresponding function ρ^{-1} which returns sets of traces from $\mathcal{W}^T[i]$.

For the pointed normative structure $\langle K^{\mathcal{N}}, i\rangle$ we construct the pointed tree structure $\langle TK^{\mathcal{N}}, \varepsilon\rangle = (\mathcal{W}^T[i], R^T_{2^{\mathcal{A}_B}}, \mathcal{V}^T, \varrho^T)$. The function $R^T_{2^{\mathcal{A}_B}}$ assigns a partial function $R^T_{2^{\mathcal{A}_B}}(\alpha) : \mathcal{W}^T[i] \to \mathcal{W}^T[i]$ to each α (we write the partial functions as sets of pairs of argument/value) which is defined as:

$$R^T_{2^{\mathcal{A}_B}}(\alpha) = \{(x, x\alpha s) \mid (\rho(x), s) \in R_{2^{\mathcal{A}_B}}(\alpha)\}.$$

The valuation function $\mathcal{V}^T$ is defined in terms of $\mathcal{V}$:

$$\mathcal{V}^T(\phi) = \rho^{-1}(\mathcal{V}(\phi)).$$

Above, we used the standard pointwise extension of the function ρ^{-1} over a set of elements as argument. The marking function ϱ^T is defined in terms of ϱ:

$$\varrho^T(x) = \varrho(\rho(x)).$$

It is easy to see that $\langle TK^{\mathcal{N}}, \varepsilon\rangle$ is a tree structure with root node ε. We can check that $\langle TK^{\mathcal{N}}, \varepsilon\rangle$ is a normative structure. The restrictions imposed by Definition 3.7 on the function $R^T_{2^{\mathcal{A}_B}}$ are met. Precisely, for any of the partial functions $R^T_{2^{\mathcal{A}_B}}(\alpha)$ it cannot be the case that $R^T_{2^{\mathcal{A}_B}}(\alpha)(x) = y\alpha s$ where $x \neq y$ (i.e., the first restriction is met). Take now two different actions $\alpha \neq \beta$ then it cannot be the case that $R^T_{2^{\mathcal{A}_B}}(\alpha)(x) = R^T_{2^{\mathcal{A}_B}}(\beta)(x)$ because $R^T_{2^{\mathcal{A}_B}}(\alpha)(x) = x\alpha s \neq x\beta s' = R^T_{2^{\mathcal{A}_B}}(\beta)(x)$ even if $s = s'$. $\Box$

The following result relates satisfiability of a contract in trees and normative structures. It is the essence of the tree model property. Intuitively, if a formula holds at a state in a normative structure then the formula holds at the root of the tree obtained from the unfolding of the normative structure starting with that particular state.

Theorem 3.9. *For a pointed structure $\langle K^{\mathcal{N}}, i\rangle$ we have:*

$$TK^{\mathcal{N}}, x \models \mathcal{C} \quad \textit{iff} \quad K^{\mathcal{N}}, \rho(x) \models \mathcal{C} \tag{1}$$

$$TK^{\mathcal{N}}, \varepsilon \models \mathcal{C} \quad \textit{iff} \quad K^{\mathcal{N}}, i \models \mathcal{C} \tag{2}$$

Proof: The proof of (2) follows from (1) by replacing x with ε (and thus $\rho(\varepsilon) = i$). The proof of (1) is done by induction on the structure of the formula $\mathcal{C}$. It has lengthy but easy cases as it needs to prove each of the conditions in the definitions of the semantics of the deontic modalities.

Basis: The case for when $\mathcal{C} = \bot$ is trivial. The second base case is when $\mathcal{C} = \phi$. Then $TK^{\mathcal{N}}, x \models \phi$ iff $x \in \mathcal{V}^T(\phi)$ which means that $\rho(x) \in \rho(\mathcal{V}^T(\phi))$. By the definition of $\mathcal{V}^T$ from Lemma 3.8 it means that $\rho(x) \in \rho(\rho^{-1}(\mathcal{V}(\phi)))$ which is $\rho(x) \in \mathcal{V}(\phi)$. This is equivalent to $K^{\mathcal{N}}, \rho(x) \models \phi$ and the proof is finished.

Case for $\mathcal{C} = P(\alpha)$. We prove that $TK^{\mathcal{N}}, x \models P(\alpha)$ iff $K^{\mathcal{N}}, \rho(x) \models P(\alpha)$.

First we prove that $T(\alpha)\ \mathcal{S}_x\, TK^{\mathcal{N}}$ iff $T(\alpha)\ \mathcal{S}_{\rho(x)}\, K^{\mathcal{N}}$, which is equivalent to proving $r\,\mathcal{S}\,x$ iff $r\,\mathcal{S}\,\rho(x)$, where r is the root of $T(\alpha)$. Because we use this result in several places we refer to it as the *simulation result*. The Definition 3.3 says that from $r\,\mathcal{S}\,x$ we have that $\forall r \xrightarrow{\alpha_\times} t \in T(\alpha)$ then $\exists x \xrightarrow{\gamma} x\gamma s \in TK^{\mathcal{N}}$ s.t. $\alpha_\times \subseteq \gamma$ and $t\,\mathcal{S}\,x\gamma s$ (where, throughout this proof, we consider $s \in \mathbb{N}$). More precisely, $x \xrightarrow{\gamma} x\gamma s \in TK^{\mathcal{N}}$ means that $(x, x\gamma s) \in R^T_{2^{\mathcal{A}_B}}(\gamma)$, which, by the definition of $R^T_{2^{\mathcal{A}_B}}$ from Lemma 3.8, implies that $(\rho(x), s) \in R_{2^{\mathcal{A}_B}}(\gamma)$. Thus we have that $\forall r \xrightarrow{\alpha_\times} t \in T(\alpha)$ then $\exists \rho(x) \xrightarrow{\gamma} s \in K^{\mathcal{N}}$ s.t. $\alpha_\times \subseteq \gamma$. By applying a recursive reasoning with $t\,\mathcal{S}\,x\gamma s$ we also get that $t\,\mathcal{S}\,s$ in $K^{\mathcal{N}}$. The recursive reasoning is possible because the trees associated to deontic actions are finite, i.e., have finite height. This means that eventually we reach a leaf node in the tree (say t); and any leaf node is trivially simulated by any state (as from a leaf there is no edge to look at). To finish the simulation result we prove the second condition from the definition of the simulation relation. We use *reductio ad absurdum* and assume $\exists \rho(x) \xrightarrow{\gamma} s \in K^{\mathcal{N}}$ with $\alpha_\times \subseteq \gamma$ for which $t\,\mathcal{S}\,s$ is not the case. From Lemma 3.8 we know that $\exists x \xrightarrow{\gamma} x\gamma s \in TK^{\mathcal{N}}$, and from $T(\alpha)\ \mathcal{S}_x\, TK^{\mathcal{N}}$ we also know that $\forall x \xrightarrow{\gamma} x\gamma s \in TK^{\mathcal{N}}$ with $\alpha_\times \subseteq \gamma$ we have $t\,\mathcal{S}\,x\gamma s$ which, by a similar recursive argument, means that $t\,\mathcal{S}\,s$, hence the contradiction. The proof for the right to left direction is analogues, using Lemma 3.8.

We continue to prove the second condition from the definition of the semantics of P; i.e.

$\forall r \xrightarrow{\gamma} t \in T(\alpha), \forall x \xrightarrow{\gamma'} x\gamma' s \in TK^{\mathcal{N}}$ s.t. $r\,\mathcal{S}\,x \wedge \gamma \subseteq \gamma'$

then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\bullet_a \notin \varrho^T(x\gamma' s)$
$\Leftrightarrow$
$\forall r \xrightarrow{\gamma} t \in T(\alpha), \forall \rho(x) \xrightarrow{\gamma'} s \in K^{\mathcal{N}}$ s.t. $r\,\mathcal{S}\,\rho(x) \wedge \gamma \subseteq \gamma'$
then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\bullet_a \notin \varrho(s)$.

Consider the implication "$\Rightarrow$". For an arbitrary $\rho(x) \xrightarrow{\gamma'} s \in K^{\mathcal{N}}$ we know from Lemma 3.8 that $\exists x \xrightarrow{\gamma'} x\gamma' s \in TK^{\mathcal{N}}$ and from the hypothesis we know that $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\bullet_a \notin \varrho^T(x\gamma' s)$. By the definition of ϱ^T, from Lemma 3.8, we know that $\varrho^T(x\gamma' s) = \varrho(\rho(x\gamma' s)) = \varrho(s)$, and thus, we have our conclusion $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\bullet_a \notin \varrho(s)$.

Consider now "$\Leftarrow$". For an arbitrary $x \xrightarrow{\gamma'} x\gamma' s \in TK^{\mathcal{N}}$ we know that we have $\exists \rho(x) \xrightarrow{\gamma'} s \in K^{\mathcal{N}}$ with $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\bullet_a \notin \varrho(s)$. Consider now, by *reductio ad absurdum*, that $\exists a \in \mathcal{A}_B$ with $a \in \gamma$ and $\bullet_a \in \varrho^T(x\gamma' s)$. By the definition of ϱ^T we have that $\bullet_a \in \varrho(s)$ which is a contradiction with the hypothesis. Thus the case is finished.

Inductive step:

Case for $\mathcal{C} = O_{\mathcal{C}}(\alpha)$.[7] We prove that $TK^{\mathcal{N}}, x \models O_{\mathcal{C}}(\alpha)$ iff $K^{\mathcal{N}}, \rho(x) \models O_{\mathcal{C}}(\alpha)$ under the inductive hypothesis $\forall x \in TK^{\mathcal{N}}$ then $TK^{\mathcal{N}}, x \models \mathcal{C} \Leftrightarrow K^{\mathcal{N}}, \rho(x) \models \mathcal{C}$.

We have proven that $T(\alpha)\ \mathcal{S}_x\, TK^{\mathcal{N}}$ iff $T(\alpha)\ \mathcal{S}_{\rho(x)}\, K^{\mathcal{N}}$ in the *simulation result*. We now prove the second requirement from the definition of the semantics of obligations, i.e., we prove the double implication:

$\forall r \xrightarrow{\gamma} t \in T(\alpha), \forall x \xrightarrow{\gamma'} x\gamma' s \in TK^{\mathcal{N}}$ s.t. $r\,\mathcal{S}\,x \wedge \gamma \subseteq \gamma'$
then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \in \varrho^T(x\gamma' s)$
$\Leftrightarrow$
$\forall r \xrightarrow{\gamma} t \in T(\alpha), \forall \rho(x) \xrightarrow{\gamma'} s \in K^{\mathcal{N}}$ s.t. $r\,\mathcal{S}\,\rho(x) \wedge \gamma \subseteq \gamma'$
then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \in \varrho(s)$.

The proof is similar to what we did for permissions. We consider only the "$\Rightarrow$" implication. For an arbitrary $\rho(x) \xrightarrow{\gamma'} s \in K^{\mathcal{N}}$ we know that we have $x \xrightarrow{\gamma'} x\gamma' s \in TK^{\mathcal{N}}$ and from the hypothesis we know that $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \in \varrho^T(x\gamma' s)$. By the definition of ϱ^T we have that $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \in \varrho(s)$.

To prove the third condition from the definition of the semantics of $O_{\mathcal{C}}(\alpha)$ we prove the following double implication:

$\forall x \xrightarrow{\gamma} x\gamma s \in TK_{rem}^{T(\alpha),x}$ then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \notin \varrho^T(x\gamma s)$
$\Leftrightarrow$
$\forall \rho(x) \xrightarrow{\gamma} s \in K_{rem}^{T(\alpha),\rho(x)}$ then $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \notin \varrho(s)$.

We do the proof of "$\Rightarrow$" using the *reductio ad absurdum* principle (the proof of "$\Leftarrow$" is analogous). Suppose that $\exists \rho(x) \xrightarrow{\gamma} s \in K_{rem}^{T(\alpha),\rho(x)}$ and $\exists a \in \mathcal{A}_B$ with $a \in \gamma$

[7]For notation simplicity we use the same symbol $\mathcal{C}$, but the $\mathcal{C}$ in the subscript of O, the reparation, is not the same as the one on the left of the equal sign, which comes from the initial enunciation of the theorem.

s.t. $\circ_a \in \varrho(s)$. This implies that there also exists the transition $x \xrightarrow{\gamma} x\gamma s \in TK^{\mathcal{N}}$ for which, from the definition of ϱ^T from Lemma 3.8, it also holds that $\circ_a \in \varrho^T(x\gamma s) = \varrho(s)$. This requires two subcases:

a. $x \xrightarrow{\gamma} x\gamma s \in TK_{rem}^{T(\alpha),x}$ which, from the precondition of the implication, means that $\circ_a \notin \varrho^T(x\gamma s)$, resulting in a contradiction;

b. $x \xrightarrow{\gamma} x\gamma s \notin TK_{rem}^{T(\alpha),x}$. From the assumption $\exists \rho(x) \xrightarrow{\gamma} s \in K_{rem}^{T(\alpha),\rho(x)}$, by way of Definition 3.5, it means that $\rho(x) \in K_{max}^{T(\alpha),\rho(x)}$ and $\exists \rho(x) \xrightarrow{\alpha_\times} s' \in K_{max}^{T(\alpha),\rho(x)}$. This implies that $x \in TK_{max}^{T(\alpha),x}$ and $\exists x \xrightarrow{\alpha_\times} x\alpha_\times s' \in TK_{max}^{T(\alpha),x}$, which, together with the assumption of this case and by way of Definition 3.5, means that $x \xrightarrow{\gamma} x\gamma s \in TK_{max}^{T(\alpha),x}$. By the Definition 3.4 of the maximal simulating structure it implies that $\exists r \xrightarrow{\alpha'_\times} t \in T(\alpha)$ s.t. $r\,\mathcal{S}\,x \wedge (\alpha'_\times \subseteq \gamma) \wedge t\,\mathcal{S}\,x\gamma s$. By the reasoning we did at the beginning for the $T(\alpha)\ \mathcal{S}_x\, TK^{\mathcal{N}}$ we conclude that $r\,\mathcal{S}\,\rho(x) \wedge (\alpha_\times \subseteq \gamma) \wedge t\,\mathcal{S}\,s$ which means that $\rho(x) \xrightarrow{\gamma} s \in TK_{max}^{T(\alpha),\rho(x)}$. Thus we have a contradiction.

We need to prove the last condition from the definition of the semantics of $O_{\mathcal{C}}(\alpha)$; i.e., we prove the double implication:

$$TK^{\mathcal{N}}, x \models \mathcal{C} \quad \forall x \in TK^{\mathcal{N}} \text{ with } t\,\mathcal{S}^s\,x \wedge t \in \mathit{leafs}(T(\overline{\alpha}))$$
$$\Leftrightarrow$$
$$K^{\mathcal{N}}, s \models \mathcal{C} \quad \forall s \in K^{\mathcal{N}} \text{ with } t\,\mathcal{S}^s\,s \wedge t \in \mathit{leafs}(T(\overline{\alpha})).$$

Here we use the induction hypothesis. We prove only the forward implication by *reductio ad absurdum* and assume that $\exists s \in K^{\mathcal{N}}$ s.t. s is reached by following *exactly* (because of the strong simulation condition $\mathcal{S}^s$) one final path in the tree of the complemented action $T(\overline{\alpha})$. For this state we assume $K^{\mathcal{N}}, s \not\models \mathcal{C}$. We have, thus, the sequence of transitions in $K^{\mathcal{N}}$: $r \xrightarrow{\alpha_\times^1} 1, 1 \xrightarrow{\alpha_\times^2} 2, \ldots, n-1 \xrightarrow{\alpha_\times^n} n$ (recall that we consider the states of $K^{\mathcal{N}}$ to be labeled with natural numbers) where $n = s$. For each of these transitions there is a transition in $TK^{\mathcal{N}}$: $\exists \varepsilon \xrightarrow{\alpha_\times^1} \varepsilon\alpha_\times^1 1$, $\exists \varepsilon\alpha_\times^1 1 \xrightarrow{\alpha_\times^2} \varepsilon\alpha_\times^1 1\alpha_\times^2 2$, $\ldots$, $\exists \varepsilon\alpha_\times^1 \ldots n-1 \xrightarrow{\alpha_\times^n} \varepsilon\alpha_\times^1 \ldots \alpha_\times^n n$. From this and the left part of the implication we have that $TK^{\mathcal{N}}, \varepsilon\alpha_\times^1 \ldots \alpha_\times^n n \models \mathcal{C}$. By the inductive hypothesis it means that $K^{\mathcal{N}}, \rho(\varepsilon\alpha_\times^1 \ldots \alpha_\times^n n) \models \mathcal{C}$ which is $K^{\mathcal{N}}, n \models \mathcal{C}$ (or $K^{\mathcal{N}}, s \models \mathcal{C}$). Hence, the contradiction and the end of the proof.

For the natural obligations from Definition 3.19 in the next section we need to treat the naturalness condition too; this means proving the following double implication:

$$\exists \gamma \text{ s.t. } T(\alpha \times \gamma) \doteq TK_{max}^{T(\alpha),x}$$
$$\Leftrightarrow$$
$$\exists \gamma' \text{ s.t. } T(\alpha \times \gamma') \doteq TK_{max}^{T(\alpha),\rho(x)}.$$

We actually prove that $TK_{max}^{T(\alpha),x} \doteq TK_{max}^{T(\alpha),\rho(x)}$ which implies that $\gamma = \gamma'$ solves the double implication. Note first that $TK_{max}^{T(\alpha),\rho(x)}$ is the tree unfolding of the $K_{max}^{T(\alpha),\rho(x)}$ maximal simulating structure of $K^{\mathcal{N}}$ w.r.t. the state $\rho(x)$, whereas,

$TK_{max}^{T(\alpha),x}$ is the maximal simulating structure coming from the tree unfolding of $K^{\mathcal{N}}$ w.r.t. the state $\rho(x)$. We use a recursive reasoning working on levels of the two trees, beginning at the first level of edges, those starting in the roots of the two trees.

Pick some arbitrary edge $\varepsilon \xrightarrow{\gamma} \varepsilon\gamma s \in TK_{max}^{T(\alpha),\rho(x)}$ for which we want to find a corresponding edge in $TK_{max}^{T(\alpha),x}$. This means that it exists $\rho(x) \xrightarrow{\gamma} s$ a transition in $K_{max}^{T(\alpha),\rho(x)}$.[8] Because x is the root of $TK_{max}^{T(\alpha),x}$ we can find the edge $x \xrightarrow{\gamma} x\gamma s \in TK_{max}^{T(\alpha),x}$, which is the edge we were looking for.

For the forward direction pick some arbitrary edge $x \xrightarrow{\gamma} x\gamma s \in TK_{max}^{T(\alpha),x}$. This means that we have an edge $\rho(x) \xrightarrow{\gamma} s \in K_{max}^{T(\alpha),\rho(x)}$ which means that we have the desired edge $\varepsilon \xrightarrow{\gamma} \varepsilon\gamma s \in TK_{max}^{T(\alpha),\rho(x)}$, as $\rho(x) = \rho'(\varepsilon)$.

The case for $\mathcal{C} = F_{\mathcal{C}}(\alpha)$ follows similar reasoning as for $O_{\mathcal{C}}$ only that care must be taken when dealing with the partial simulation relation $\tilde{\mathcal{S}}$.

The case for the propositional implication uses simple structural induction. $\square$

Note that we have proven Theorem 3.9 both for general $\mathcal{CL}$ (i.e., with general obligations) and for $\mathcal{CL}$ restricted to natural obligations, as in the next section, i.e., where we can reason only about natural obligations as the semantics constrains us to.

Corollary 3.10 (tree model property).
If $\mathcal{C}$ has a model $K^{\mathcal{N}}$ then it has a tree model $TK^{\mathcal{N}}$.

Proof: This follows immediately from equation (2) of the Theorem 3.9 which says that if a formula $\mathcal{C}$ is true in a state i of a model $K^{\mathcal{N}}$ then there exists a tree model $TK^{\mathcal{N}}$, as in Lemma 3.8, in which the formula is true at state ε. $\square$

Next we prove that the deontic modalities alone have the *finite model property*. There are several techniques for proving decidability of modal logics by establishing a finite model property; where a known one is called *filtration*. Filtration is especially used for dynamic logics, like the PDL, from which we borrow the dynamic box modality in $\mathcal{CL}$ in the next section. However, it is rather hard to use the filtration technique in our case. In PDL the clever Fischer-Ladner closure was needed so to determine the subformulas of a dynamic modality with a complex action inside (e.g. $[a \cdot (b + c)]\varphi$). In our case we do not know what are subformulas of an obligation of a complex action like $O_{\mathcal{C}}(a \cdot (b + c))$. We use, instead, the *selection* technique for proving the finite model property [36, sec.2.3]. Selection is known especially for modal logics where a tree model property has been established, like is our case. The basic idea is that given a possibly infinite model, the selection technique selects and removes (possibly infinite) parts of this model eventually ending up with a finite model. This selection and removing is done carefully so that the satisfiability of the formula of interest is not broken.

[8] Note that there should be two ρ functions, one coming from the unfolding of $K^{\mathcal{N}}$ (which is the one in the double implication) and another ρ' function (which is not visible) coming from the unfolding of the $K_{max}^{T(\alpha),\rho(x)}$. Actually here $\rho'(\varepsilon) = \rho(x)$.

Before proving the finite model property (Theorem 3.16) we give some necessary definitions and prove auxiliary results.

Definition 3.11 (action length). *The* length of an action α *is defined (inductively) as a function* $l : \mathcal{A}^{\mathcal{D}} \to \mathbb{N}$ *from deontic actions to natural numbers.*

- $l(\mathbf{1}) = l(\mathbf{0}) = 0,$
- $l(a) = 1,$ *for any basic action* a *of* $\mathcal{A}_B$,
- $l(\alpha \times \beta) = l(\alpha + \beta) = max(l(\alpha), l(\beta)),$
- $l(\alpha \cdot \beta) = l(\alpha) + l(\beta).$

The length function counts the number of actions in a sequence of actions given by the $\cdot$ constructor. It returns the maximum from all the choices in an action, i.e., the length of the maximal sequence.

The following proposition states that complementing a (complex) action does not increase its length.

Proposition 3.12. *For any action* α *we have* $l(\overline{\alpha}) \leq l(\alpha)$.

Proof : A careful inspection of the Definition 2.7 of action complement easily shows that $\overline{\alpha}$ does not add $\cdot$ combinators at bigger length than those in the canonical form of α. The complement operation is applied recursively and at each recursive step it generates paths of length 1 for paths of length 1 or greater in the original $\underline{\alpha}$; or it generates paths of length $1+$ length generated in the next recursive step. This happens for each $\cdot$ found in $\underline{\alpha}$. Therefore, $\overline{\alpha}$ cannot have paths of greater length than the paths in $\underline{\alpha}$.

It remains to show that the length of an action is greater than the length of its canonical form. Because to obtain the canonical form of an action it is enough to apply the axioms of Table 1 except (8), cf. Theorem 2.5, we check that for each axiom the left action has length greater or equal to the right action (as the axioms are directed from left to right). This check is easy and we skip details. For axioms (1), (2), (10), (11) the left hand side (lhs) has the same length as the right hand side (rhs) because of the associativity and commutativity properties of the max operation on natural numbers. Also equal lengths of the actions on both sides of the axioms (4) and (14) comes from idempotence of max. For (5) use associativity of $+$ over natural numbers. Also equality for axioms (9) and (15), (16) comes from the distributivity of $+$ over max and of max over max, respectively. For (3), (12), and (6) use the facts that the number 0 is the neutral element for respectively max and $+$. For axiom (17) we observe that the length of the $\alpha_\times$ and $\beta_\times$ is 1 and also of $\alpha_\times \times \beta_\times$; from these, $max(1+l(\alpha), 1+l(\beta)) = 1+max(l(\alpha), l(\beta))$. The only two axioms for which the length of the lhs is strictly greater that the length of the rhs are (7) and (13), as it is clear that the right hand sides have length 0 which is less than whatever length the actions on the left hand side have. $\square$

We now relate the length of an action with the height of its tree.

Corollary 3.13. *For any action* α *we have* $h(T(\overline{\alpha})) \leq h(T(\alpha)) = l(\alpha)$.

Proof: This is a corollary of both Theorem 2.10 and Proposition 3.12. □

Definition 3.14 (depth of formula). *We define the* depth *of a formula inductively as a function d from formulas to natural numbers:*

- $d(\phi) = d(\perp) = 0$*;*
- $d(\mathcal{C}_1 \to \mathcal{C}_2) = max(d(\mathcal{C}_1), d(\mathcal{C}_2))$*;*
- $d(P(\alpha)) = l(\alpha)$*;*
- $d(O_\mathcal{C}(\alpha)) = d(F_\mathcal{C}(\alpha)) = l(\alpha) + d(\mathcal{C})$.

Example 3.4 Consider the action $\delta = p \cdot b + (d \times n) \cdot p \cdot p$ with its corresponding tree pictured in Fig. 2. The length of δ is 3, equal to the length of its longer right branch. The tree in Fig. 2 has clearly hight also 3. For a formula like $O_{F_\perp(d)}(\delta)$ the depth is $3 + 1$, being equal to the length of δ plus the depth of $F_\perp(d)$ which is $l(d) = 1$ plus the depth of $\perp$ which is 0. □

Lemma 3.15. *Take a formula $\mathcal{C}$ with depth k. If $TK^\mathcal{N}, \varepsilon \models \mathcal{C}$ then $\mathcal{C}$ holds in the root of the tree structure $\langle TK^\mathcal{N}, \varepsilon\rangle$ restricted to paths of maximum depth k (i.e., where all nodes of depth greater than are removed).*

Proof: We use induction on the structure of the formula $\mathcal{C}$.

Base case: The proof for formulas $\perp$ and ϕ which have depth 0 is simple as we need to inspect only the root node ε therefore we need only nodes of depth 0 in the tree structure.

For the formula $P(\alpha)$ which has depth $l(\alpha)$ we need to inspect only those nodes of $\langle TK^\mathcal{N}, \varepsilon\rangle$ that respect the simulation relation. Therefore, the maximum depth of a node is the maximum length in the final paths of $T(\alpha)$, and thus the maximum depth of the nodes in $\langle TK^\mathcal{N}, \varepsilon\rangle$ is $h(T(\alpha))$ which, by Corollary 3.13, is $l(\alpha)$.

Inductive step: When $\mathcal{C}$ is of the form $\mathcal{C}_1 \to \mathcal{C}_2$ the depth of the formula is the maximum of the depths of the two subformulas. The semantics says that we need to check first if $\mathcal{C}_1$ holds, which by the inductive hypothesis it means that we need a subtree of depth at most $d(\mathcal{C}_1)$. If $\mathcal{C}_1$ holds we need to check also $\mathcal{C}_2$ which, by the inductive hypothesis, requires also a subtree of depth at most $d(\mathcal{C}_2)$. Overall, we need to check a subtree of $\langle TK^\mathcal{N}, \varepsilon\rangle$ with depth at most $max(d(\mathcal{C}_1), d(\mathcal{C}_2))$.

The proof for the formulas $O_\mathcal{C}(\alpha)$ and $F_\mathcal{C}(\alpha)$ is similar and we treat here only the proof for obligations. The semantics of $O_\mathcal{C}(\alpha)$ says that we need to check first the obligation alone which requires nodes of depth at most $h(T(\alpha)) = l(\alpha)$ because of the simulation relation. Secondly, we need to check that the reparation $\mathcal{C}$ holds at the states corresponding to the leaf nodes of the complement $\overline{\alpha}$. By Corollary 3.13 we know that these states are at depth at most $l(\alpha)$. The induction hypothesis says that to check $\mathcal{C}$ it is required a tree of height at most the depth $d(\mathcal{C})$. Therefore, to check $O_\mathcal{C}(\alpha)$ it requires to check a subtree of $\langle TK^\mathcal{N}, \varepsilon\rangle$ where the nodes have a depth at most $l(\alpha) + d(\mathcal{C})$. □

Based on the above auxiliary results we can prove now the finite model property.

Theorem 3.16 (finite model property). *If a formula has a model then it has a finite model.*

Proof : We can work equivalently in pointed structures and then we need to prove that if a formula is satisfied in a pointed structure then it is satisfied in a finite pointed structure. Take a formula $\mathcal{C}$ of depth k which is satisfiable in the pointed structure $K^{\mathcal{N}}, i$. By Corollary 3.10 we know that $\mathcal{C}$ is satisfied in the tree-like pointed structure $TK^{\mathcal{N}}, \varepsilon$. Note that the tree might have both infinite depth and infinite branching.

By Lemma 3.15 we put a bound on the depth of the tree which is related to the formula. Because we work with deterministic structures and because the set of labels $2^{\mathcal{A}_B}$ is finite we have a guaranteed finite branching. Therefore the model is finite. □

As a corollary of the above theorem we have a (relative) decidability result.

Corollary 3.17 (decidability).

a. The logic with general obligations as in the semantic Definition 3.6 is decidable.

b. The logic with natural obligations is decidable iff the naturalness constraint is decidable.

As a side remark, we have proven the tree model property in Theorem 3.9 for general obligations as well as for natural obligations. Therefore, in both cases it is enough to check finite trees for satisfiability, but the difference is that when checking for natural obligations we need to test that the naturalness constraint is satisfied. The decidability of whether an action γ satisfies the naturalness constraint (cf. Propositional 3.20) has been an open problem for a while. Recent results in [38] (see details in the technical report [39]) show an algorithm to find a suitable γ if one exists (i.e., a decision procedure is given).

3.2. Properties of the deontic modalities

The semantics of the deontic modalities is rather involved; it is based on an algebraic formalism for the actions which are interpreted as rooted trees. The information in the trees (compared to sets of traces [33]) is used by the particular notion of simulation relation to know how to walk on the normative structure in the search of the markings to determine the truth value of the deontic modality. The rest of the complications in the semantics are necessary for capturing several intuitive properties of the deontic modalities which we discuss in this section.

The following validities are the counterparts of the ones found in SDL only that here they are in an ought-to-do setting where the deontic modalities are applied over actions. The following examples give intuition for the logical validities of Proposition 3.18 (taken from [40]):

- Obligation of an action implies that the action is permitted:
$$\models O_{\mathcal{C}}(\alpha) \to P(\alpha)$$
E.g.: “Client is obliged to pay” then “Client has the right to pay”.

- Permission of performing an action implies that the action is not forbidden:
$$\models P(\alpha) \rightarrow \neg F_{\mathcal{C}}(\alpha)$$
E.g.: "Provider has the right to alter personal data" then "Provider is not forbidden to alter personal data".

- If two action expressions represent the same action then the obligation of one action should imply the obligation of the other action:
$$\text{if } \alpha = \beta \text{ then } \models O_{\mathcal{C}}(\alpha) \leftrightarrow O_{\mathcal{C}}(\beta)$$
E.g.: "pay or delay" is the same as "delay or pay" then "Client is obliged to pay or delay" should be the same as "Client is obliged to delay or pay".

- The obligation of the violating action cannot appear in a contract ($\models \neg O_{\mathcal{C}}(\mathbf{0})$); obligation of terminating the contract can be modeled. The obligation to do nothing can be trivially inserted in any contract ($\models O_{\mathcal{C}}(\mathbf{1})$).

Proposition 3.18 (validities). *The following statements hold:*

$$\begin{array}{rlr@{\qquad}rlr}
 & \models \neg O_{\mathcal{C}}(\mathbf{0}) & (3) & & \models O_{\mathcal{C}}(\alpha) \rightarrow P(\alpha) & (6) \\
serial & \models O_{\mathcal{C}}(\mathbf{1}) & (4) & \textit{if } \alpha = \beta \textit{ then} & \models O_{\mathcal{C}}(\alpha) \leftrightarrow O_{\mathcal{C}}(\beta) & (7) \\
 & \models P(\alpha) \rightarrow \neg F_{\mathcal{C}}(\alpha) & (5) & & \models O_{\mathcal{C}}(\alpha) \rightarrow \neg F_{\mathcal{C}}(\alpha) & (8)
\end{array}$$

Proof: For the proof of (3), i.e., $\models \neg O_{\mathcal{C}}(\mathbf{0})$, we need to show that there is no model which makes $O_{\mathcal{C}}(\mathbf{0})$ true. This is because of the definition of $\neg O_{\mathcal{C}}(\mathbf{0})$ as $O_{\mathcal{C}}(\mathbf{0}) \rightarrow \perp$ which is true only if $O_{\mathcal{C}}(\mathbf{0})$ is false. By *reductio ad absurdum* suppose that it exists a model which makes $O_{\mathcal{C}}(\mathbf{0})$ true. This means (by the definition of the semantics of O) that the tree interpreting **0** must be simulated by the model. But this is not possible because of the special label Λ appearing in the tree of **0** which does not appear in the labels of the normative structures.

To prove (4), i.e., that $O_{\mathcal{C}}(\mathbf{1})$ is valid in the *serial* normative structures, take any normative structure that respects the seriality condition from standard modal logics; i.e., that from each state there exists an outgoing transition. The tree interpreting $\mathbf{1}$ is trivially simulated by any normative structure because the only edge of the tree is labeled with the empty set and thus any transition of the structure simulates the edge. The second condition in the semantics of O is satisfied as there is no basic label a in the label of the edge. It is clear that any edge on the first level of the structure enters into the maximal simulating structure and therefore the non-simulating remainder $K_{rem}^{I(\mathbf{1}),i}$ is empty, and the third condition is trivially satisfied. Because $\overline{\mathbf{1}} = \mathbf{0}$ then there is no state s to satisfy the requirements of the last condition and thus it is trivially satisfied too.

Note that $O_{\mathcal{C}}(\mathbf{1})$ is valid only in the serial normative structures, i.e., having from each state at least one outgoing transition. We consider that seriality is natural to have because in order to be able to say that something is obligatory that something must exists in the structure (i.e., the action must label some transition, even in the case of the empty action $\mathbf{1}$ and the empty label $\{\,\}$ that models it).

For the proof of (6), i.e., $\models O_{\mathcal{C}}(\alpha) \rightarrow P(\alpha)$, take an arbitrary pointed normative structure $K^{\mathcal{N}}, i$ which makes $O_{\mathcal{C}}(\alpha)$ true. This means that $T(\alpha)\ \mathcal{S}_i\ K^{\mathcal{N}}$. This is the first part from the semantics of $P(\alpha)$. Moreover, from the semantics of $O_{\mathcal{C}}(\alpha)$ we have that $\forall t \xrightarrow{\gamma} t' \in T(\alpha)$, $\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ s.t. $t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma'$ then $\forall a \in \gamma$ we have $\circ_a \in \varrho(s')$. Because of the restriction on the marking function we get $\forall a \in \gamma$ we have $\bullet_a \notin \varrho(s')$. This makes the second requirement in the semantics of $P(\alpha)$.

For the proof of (5), i.e., $\models P(\alpha) \rightarrow \neg F_{\mathcal{C}}(\alpha)$, we use *reductio ad absurdum* and assume that it exists a pointed structure $K^{\mathcal{N}}, i$ which satisfies $P(\alpha)$ and also $F_{\mathcal{C}}(\alpha)$. From the semantics of P, knowing that $\mathcal{S} \subseteq \tilde{\mathcal{S}}$, we conclude that $T(\alpha)\ \tilde{\mathcal{S}}_i\, K^{\mathcal{N}}$. Now take any final path σ in the tree of α; from the semantics of P it holds that $\sigma\ \mathcal{S}_i\, K^{\mathcal{N}}$. Moreover, for any edge $t \xrightarrow{\gamma} t' \in \sigma$ it holds, by the semantics of P, that $\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ with $t\,\mathcal{S}\,s$ then $\forall a \in \gamma$, $\bullet_a \notin \varrho(s')$. But the semantics of $F_{\mathcal{C}}(\alpha)$ requires that $\forall a \in \gamma'$ then $\bullet_a \in \varrho(s')$ which is not possible as $\gamma \subseteq \gamma'$ (i.e., it exists at least one $\bullet_a \notin \varrho(s')$ with $a \in \gamma'$).

To prove (7) notice that the semantics of O is based on the interpretation of the actions as trees. Therefore, because the actions are equal, the tree interpretations denote the same tree (up to isomorphism). Thus, the semantics for $O_{\mathcal{C}}(\alpha)$ is the same as that for $O_{\mathcal{C}}(\beta)$ because they are working with the same tree $T(\alpha) \doteq T(\beta)$.

The proof of (8) can be obtained from (6) and (5). $\square$

In our ought-to-do setting, the formula $O_{\mathcal{C}}(\mathbf{1})$ is the counterpart of the $O(\top)$ from standard deontic logic. As the proof above shows, this formula is not valid in any normative structure, but only in the serial ones. This formula essentially says that in any contract any agent is obliged to do action "skip". This fact is harmless because if there is any other "real" obligation (in the same world) then the property from Theorem 3.21 below would combine the two obligations and by virtue of the fact that $\mathbf{1}$ is identity element for $\times$, the $O_{\mathcal{C}}(\mathbf{1})$ will be essentially swallowed by the real obligation. In consequence, $O_{\mathcal{C}}(\mathbf{1})$ is not visible when other obligations are present. Alone, $O_{\mathcal{C}}(\mathbf{1})$ requires that any normative structure is serial, i.e., that from any world there is an outgoing transition. Otherwise, obligation to "skip" does not impose anything; does not impose any markings in the structure and does not impose any particular labeling of any transition. It has an analogous behavior with $O(\top)$ of SDL.

In Theorem 3.21 (see below) we give a property for obligations over synchronous actions. The theorem states that if "there exists an obligation to do action α and there is also an obligation to do action β" (in the same current world) then we should be able to infer that "there is an obligation to do both actions α and β at the same time". This property does not hold for general obligations (with the definition that we gave before), but only for some restricted obligations, which we call *natural obligations*.

The purpose of natural obligations is not necessarily a technical one but also a practical one. The naturalness constraint refers mainly to choices of actions; when deciding which of the actions to choose the model should not influence the decision.

Definition 3.19 (natural obligations). *An obligation $O_{\mathcal{C}}(\alpha)$ is called* natural *iff in addition to the semantics of Definition 3.6 the following* naturalness constraint *is respected:*

$$\exists \gamma \text{ s.t. } \quad T(\alpha \times \gamma) \doteq TK_{max}^{T(\alpha),\varepsilon} \tag{9}$$

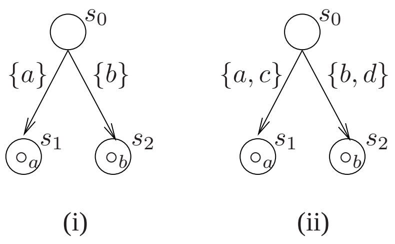

Figure 5: Examples for natural obligations.

The following result is intended to be the algebraic equivalent of naturalness, and hence, algebraic methods can be employed to solve it. The proposition provides a practical way to check that the naturalness constraint is satisfied.

Proposition 3.20. *The naturalness constraint reduces to showing that there exists a deontic action γ s.t. $\alpha \times \gamma = \alpha^T$, where α^T is the action in canonical form corresponding to the tree $TK_{max}^{T(\alpha),\varepsilon}$.*

Proof: This is a consequence of the definition above and of the completeness result of [7] which says that for any tree as in Theorem 2.10 there is a corresponding action. □

Example 3.5 Let us consider the model of Fig. 5-(i) in which $O(a + b)$ holds at state s_0. Change this model by adding a c to the left label and a d to the right label, as in Fig. 5-(ii). $O(a + b)$ still holds at state s_0, but is not a natural obligation; intuitively, when deciding which of a or b to choose one needs to take into account the two distinct actions c and d. If we were to add the same label c to both branches then the naturalness constraint is satisfied, as one does not care about the extra action c when choosing. □

Theorem 3.21 (synchrony property). *For natural obligations we have:*

$$\models O_{\mathcal{C}}(\alpha) \land O_{\mathcal{C}}(\beta) \rightarrow O_{\mathcal{C}}(\alpha \times \beta) \tag{10}$$

Proof: We need to prove that $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha \times \beta)$ under the assumption $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \land O_{\mathcal{C}}(\beta)$. Using Lemma A.4 we have that $T(\alpha \times \beta)\ \mathcal{S}_i\ K^{\mathcal{N}}$ which is the first requirement in the semantics of $O_{\mathcal{C}}$. For the proof of Lemma A.4 the naturalness constraint is essential. The proofs of Lemma A.7 and Lemma A.8 are also based on the naturalness constraint. These two lemmas give the second and the third requirement in the semantics of $O_{\mathcal{C}}(\alpha \times \beta)$. The last requirement is proven as Lemma A.9. □

We now show how the above result can be generalized to the conjunction of obligations containing different reparations.

Proposition 3.22. *For natural obligations we have:*

$$\models O_{\mathcal{C}_1}(\alpha) \land O_{\mathcal{C}_2}(\beta) \rightarrow O_{\mathcal{C}_1 \lor \mathcal{C}_2}(\alpha \times \beta) \tag{11}$$

Proof : The proof of this result is the same as that for Theorem 3.21 until we need to use Lemma A.9. The statement of this last lemma needs to be changed accordingly and the new proof needs only little change as we discuss here. What we need to prove now is that if $K^{\mathcal{N}}, \underline{i} \models O_{\mathcal{C}_1}(\alpha) \wedge O_{\mathcal{C}_2}(\beta)$ then $K^{\mathcal{N}}, s \models \mathcal{C}_1 \vee \mathcal{C}_2 \quad \forall s \in N$ with $t\, \mathcal{S}^s\, s \wedge t \in \mathit{leafs}(T(\overline{\alpha \times \beta}))$. The proof of this result is essentially the same as the proof of Lemma A.9 with the following observations. In the forth paragraph we proved that from the condition $\forall \alpha^i_\times \times \beta^j_\times : \alpha^i_\times \times \beta^j_\times \not\subseteq \gamma$ we can conclude that either $\forall \alpha^i_\times : \alpha^i_\times \not\subseteq \gamma$ or $\forall \beta^j_\times : \beta^j_\times \not\subseteq \gamma$. This is used in the last paragraph by taking one of the hypothesis for which this holds, because it does not make a difference which hypothesis we take as both have the same reparation $\mathcal{C}$. This is not the case for the present corollary. Here it is important which of $\forall \alpha^i_\times : \alpha^i_\times \not\subseteq \gamma$ or $\forall \beta^j_\times : \beta^j_\times \not\subseteq \gamma$ holds; but we do not know. Therefore we use the disjunction of the reparations $\mathcal{C}_1 \vee \mathcal{C}_2$, and regardless of which of the two holds we use the appropriate hypothesis to make the disjunction true. □

The following corollary points out *conflicts* that are avoided in the logic because of the semantics. These are usual requirements when reasoning about legal contracts. A contract with two clauses "Obliged to pay" and "Forbidden to pay" can never be respected. The same with a contract stating "Obliged to go west" and "Obliged to go east" (as "go west" and "go east" cannot be done at the same time, i.e., are conflicting).

Corollary 3.23 (conflicts). *The following statements hold:*

$$\models \neg(O_{\mathcal{C}}(\alpha) \wedge F_{\mathcal{C}}(\alpha)) \tag{12}$$

$$\models \neg(P(\alpha) \wedge F_{\mathcal{C}}(\alpha)) \tag{13}$$

$$\textit{if } \alpha \,\#_{\mathcal{C}}\, \beta \textit{ then } \models \neg(O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)) \tag{14}$$

Proof : The proof of (12) follows by propositional reasoning from (8) and the proof of (13) follows from (5). The proof of (14) follows from (3) and Theorem 3.21 as we show next. Because $\alpha \,\#_{\mathcal{C}}\, \beta$ then $\alpha \times \beta = \mathbf{0}$, by axiom (18), and therefore $O_{\mathcal{C}}(\alpha \times \beta)$ is $O_{\mathcal{C}}(\mathbf{0})$. From Theorem 3.21 we get that $\models \neg(O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)) \leftarrow \neg O_{\mathcal{C}}(\alpha \times \beta)$ and from the above we have that $\models \neg(O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)) \leftarrow \neg O_{\mathcal{C}}(\mathbf{0})$. By *modus ponens* using (3) we get $\models \neg(O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta))$. □

We give now some examples for the validities presented in Proposition 3.24 below.

- Prohibition of an action implies that any *bigger* action is prohibited:
  $$F_{\mathcal{C}}(\alpha) \rightarrow F_{\mathcal{C}}(\alpha \times \beta)$$
  E.g.: "Client is forbidden to supply false information" then we also know that "Client is forbidden to supply false information and at the same time supply correct information".

  We comment more on this property. One may think of an example like "One is forbidden to smoke" but still "One is permitted to smoke and (at the same time) sit outside in the open air". This example seems to contradict the above property. The confusion comes from the wording of the above example. A more correct wording would be: "One is forbidden to smoke and (at the same time) sit in a public place" where this action is no longer smaller than the action "smoke and (at the same time) sit outside in the open air" and thus the prohibition and the permission go along together.

- The prohibition of a choice of two actions α or β is the same as having both prohibition of α and prohibition of β:
$$F_{\mathcal{C}}(\alpha+\beta) \leftrightarrow F_{\mathcal{C}}(\alpha) \wedge F_{\mathcal{C}}(\beta)$$
E.g.: "Client is forbidden to pay in dollars or to pay in euros" implies that "Client is forbidden to pay in dollars" and that "Client is forbidden to pay in euros".

- Permission of a choice of actions is the same as permission of all the actions in the choice; i.e., validity (17).

Proposition 3.24. *The following statements hold:*

$$\models F_{\mathcal{C}}(\alpha) \rightarrow F_{\mathcal{C}}(\alpha\times\beta) \tag{15}$$
$$\models F_{\mathcal{C}}(\alpha+\beta) \leftrightarrow F_{\mathcal{C}}(\alpha) \wedge F_{\mathcal{C}}(\beta) \tag{16}$$
$$\models P(\alpha+\beta) \leftrightarrow P(\alpha) \wedge P(\beta) \tag{17}$$

Proof: We give first quick proof arguments. The proof of the first validity is based on the fact that paths in $T(\alpha\times\beta)$ contain (i.e., have bigger labels than) paths of $T(\alpha)$. The proof of the second validity is based on the fact the the paths of $T(\alpha+\beta)$ which satisfy the condition in the semantics are the same as the paths of $T(\alpha)$ and $T(\beta)$ together.

For the proof of (15) consider an arbitrary pointed structure $\langle K^{\mathcal{N}}, i\rangle$ which satisfies $F_{\mathcal{C}}(\alpha)$. In order to show that $K^{\mathcal{N}}, i \models F_{\mathcal{C}}(\alpha\times\beta)$ we need to take an arbitrary final path $\sigma \in T(\alpha\times\beta)$ which satisfies $\sigma\ \mathcal{S}_i\ K^{\mathcal{N}}$ and show that for any edge $t \xrightarrow{\gamma} t'$ on this path we have $\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ with $t\,\mathcal{S}\,s \wedge \gamma \subseteq \gamma'$ then $\forall a \in \mathcal{A}_B$ if $a \in \gamma'$ then $\bullet_a \in \varrho(s')$. Note that if a path $\sigma \in T(\alpha\times\beta)$ exists then it exists also a path $\sigma' \in T(\alpha)$ which has all the labels on the edges smaller than the corresponding ones in σ. Therefore, together with the assumption $\sigma\ \mathcal{S}_i\ K^{\mathcal{N}}$ it means that σ' also satisfies $\sigma'\ \mathcal{S}_i\ K^{\mathcal{N}}$. Because of this, we can apply the semantics for the expression $F_{\mathcal{C}}(\alpha)$ to deduce that for all edges $t \xrightarrow{\gamma} t' \in \sigma'$ all transitions $s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ satisfying $\gamma \subseteq \gamma'$ also satisfy $\forall a \in \mathcal{A}_B$ if $a \in \gamma'$ then $\bullet_a \in \varrho(s')$. For these edges we can find corresponding edges in σ that have labels γ'' which includes γ. Because $\gamma \subseteq \gamma''$ it means that all the transitions $s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ that respect $\gamma'' \subseteq \gamma'$ are among (possibly fewer than) the transitions before, for σ'. But all these transitions we know that respect $\forall a \in \mathcal{A}_B$ if $a \in \gamma'$ then $\bullet_a \in \varrho(s')$. The proof is finished.

It should be simple to see that the opposite implication does not always hold; i.e., $\not\models F_{\mathcal{C}}(\alpha\times\beta) \rightarrow F_{\mathcal{C}}(\alpha)$. This is because we cannot guarantee that by taking all the paths $\sigma' \in T(\alpha\times\beta)$ which satisfy $\sigma'\ \mathcal{S}_i\ K^{\mathcal{N}}$ we will consider all the paths $\sigma \in T(\alpha)$, because there may be paths with labels smaller that those in $T(\alpha\times\beta)$ which are still good paths for $T(\alpha)$ (see Proposition 3.25 for a counterexample).

The proof of $\models F_{\mathcal{C}}(\alpha+\beta) \leftrightarrow F_{\mathcal{C}}(\alpha) \wedge F_{\mathcal{C}}(\beta)$ is simpler. It is easy to see that the tree $T(\alpha+\beta)$ contains all the final paths σ of the two trees $T(\alpha)$ and $T(\beta)$ which satisfy

$\sigma\ \mathcal{S}_i\ K^{\mathcal{N}}$. Therefore, the double implication is immediate: if we consider $F_{\mathcal{C}}(\alpha + \beta)$ true than the traces in $T(\alpha + \beta)$ respect all the conditions of the semantics and thus all the traces in $T(\alpha)$ respect the conditions in the semantics, making $F_{\mathcal{C}}(\alpha)$ true (and the same for $F_{\mathcal{C}}(\beta)$).

The proof of (17) is similar to the proof of (16). □

In the design decisions for $\mathcal{CL}$ we give special attention to what we call *unwanted implications*. This kind of "properties" are rather scarce and neglected in the literature. We give here unwanted implications that are related only to the deontic modalities (Proposition 3.25).

- The prohibition of doing two actions at the same time does not imply that any of the two actions is prohibited (i.e., the converse of (15) does not always hold).
  $$\not\models F_{\mathcal{C}}(\alpha\times\beta) \rightarrow F_{\mathcal{C}}(\alpha)$$
  E.g.: "One is forbidden to drink and drive at the same time" does not imply that "One is forbidden to drink" and neither that "One is forbidden to drive".

- Obligation of an action α does not imply obligation of any concurrent action that contains α. Similarly, obligation of a concurrent action does not imply obligation of any of its composing actions.
  $$\not\models O_{\mathcal{C}}(\alpha) \rightarrow O_{\mathcal{C}}(\alpha\times\beta);$$
  $$\not\models O_{\mathcal{C}}(\alpha\times\beta) \rightarrow O_{\mathcal{C}}(\alpha).$$
  E.g.: "Obligation to drive" should not imply "Obligation to drive and drink at the same time". For the second unwanted implication consider "Obligation to smoke and sit outside" which should not imply "Obligation to smoke".

- Similarly with permissions:
  $$\not\models P(\alpha) \rightarrow P(\alpha\times\beta)$$
  $$\not\models P(\alpha\times\beta) \rightarrow P(\alpha)$$
  E.g.: "Permitted to smoke and sit outside in open air" does not imply "Permitted to smoke" because if one sits inside a restaurant then one is forbidden to smoke.

  The first implication is related to the free choice permission paradox on page 45 in the setting of concurrent actions, where from permission to smoke one would imply the permission to smoke and kill at the same time.

- Obligation of a choice of actions constrains that only the actions in the choice can be done but the choice itself is left open, the one on which the obligation is enforced has the freedom of choosing. Therefore, none of the actions in the choice is obligatory by itself because the freedom of choosing would be lost.
  $$\not\models O_{\mathcal{C}}(\alpha+\beta) \rightarrow O_{\mathcal{C}}(\alpha).$$
  $$\not\models O_{\mathcal{C}}(\alpha) \rightarrow O_{\mathcal{C}}(\alpha+\beta);$$
  E.g.: "Client is obliged to pay or to delay payment" should not imply that "Client is obliged to delay payment". For the second unwanted implication "Obliged to mail the letter" should not imply "Obliged to mail the letter or burn the letter".

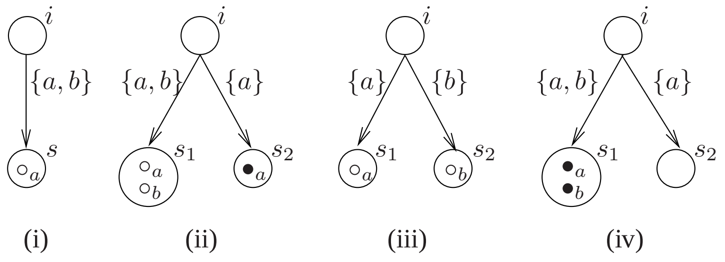

Figure 6: Counterexamples for Prop.3.25.

- As consequence of the above we have:

$$\not\models O_{\mathcal{C}}(\alpha+\beta) \rightarrow O_{\mathcal{C}}(\alpha\times\beta);$$
$$\not\models O_{\mathcal{C}}(\alpha\times\beta) \rightarrow O_{\mathcal{C}}(\alpha+\beta).$$

Proposition 3.25 (unwanted implications). *The following statements hold:*

$$\not\models O_{\mathcal{C}}(\alpha) \rightarrow O_{\mathcal{C}}(\alpha\times\beta) \quad (18)$$
$$\not\models O_{\mathcal{C}}(\alpha\times\beta) \rightarrow O_{\mathcal{C}}(\alpha) \quad (19)$$
$$\not\models O_{\mathcal{C}}(\alpha+\beta) \rightarrow O_{\mathcal{C}}(\alpha\times\beta) \quad (20)$$
$$\not\models O_{\mathcal{C}}(\alpha\times\beta) \rightarrow O_{\mathcal{C}}(\alpha+\beta) \quad (21)$$
$$\not\models O_{\mathcal{C}}(\alpha) \rightarrow O_{\mathcal{C}}(\alpha+\beta) \quad (22)$$
$$\not\models O_{\mathcal{C}}(\alpha+\beta) \rightarrow O_{\mathcal{C}}(\alpha) \quad (23)$$
$$\not\models F_{\mathcal{C}}(\alpha\times\beta) \rightarrow F_{\mathcal{C}}(\alpha) \quad (24)$$
$$\not\models P(\alpha\times\beta) \rightarrow P(\alpha) \quad (25)$$
$$\not\models P(\alpha) \rightarrow P(\alpha\times\beta) \quad (26)$$
$$\not\models P(\alpha) \rightarrow P(\alpha+\beta) \quad (27)$$

Proof : The proof is simple by giving for each not valid statement a counterexample, all of which are collected in Fig.6. The model of Fig. 6(i) makes $O_{\mathcal{C}}(a)$ true in state i but $O_{\mathcal{C}}(a\times b)$ does not hold (for (18)) and neither does $O_{\mathcal{C}}(a+b)$ (for (22)). In the model of Fig. 6(ii) $O_{\mathcal{C}}(a\times b)$ holds in state i but $O_{\mathcal{C}}(a)$ does not hold (for (19)) and also $O_{\mathcal{C}}(a+b)$ does not hold (for (21)). In the model of Fig. 6(iii) $O_{\mathcal{C}}(a+b)$ holds in state i but $O_{\mathcal{C}}(a\times b)$ does not hold (for (20)) and also $O_{\mathcal{C}}(a)$ does not hold (for (23)). The model of Fig. 6(iv) makes $F_{\mathcal{C}}(a\times b)$ true in state i but the same state does not make $F_{\mathcal{C}}(a)$ true (for (24)). For (25) take the structure in Fig. 6(ii) which is a model for $P(\alpha\times\beta)$ but not a model for $P(\alpha)$. For (26) Fig. 6(i) is an obvious example, and as well for (27) because the model satisfies $P(a)$ but not $P(a+b)$. $\square$

4. The Full Contract Logic

The contract logic $\mathcal{CL}$ adds to the deontic modalities from Section 3 the dynamic logic modality applied over synchronous actions.

Definition 4.1. *The syntax of $\mathcal{CL}$ is given in Table 3. The dynamic logic modality $[\cdot]\mathcal{C}$ is parameterized by the dynamic actions δ. The expression $[\delta]\mathcal{C}$ is read as: "after the*

$$\begin{array}{rcll}
\mathcal{C} & := & \phi \mid O_{\mathcal{C}}(\alpha) \mid P(\alpha) \mid F_{\mathcal{C}}(\alpha) \mid \mathcal{C} \to \mathcal{C} \mid [\delta]\mathcal{C} \mid \ \bot & (\mathcal{CL}\text{ expressions}) \\
\alpha & := & a \mid \mathbf{0} \mid \mathbf{1} \mid \alpha \times \alpha \mid \alpha \cdot \alpha \mid \alpha + \alpha & (\text{deontic actions}) \\
\delta & := & a \mid \mathbf{0} \mid \mathbf{1} \mid \delta \times \delta \mid \delta \cdot \delta \mid \delta + \delta \mid \delta^* \mid \varphi? & (\text{dynamic actions}) \\
\varphi & := & \phi \mid \mathbf{0} \mid \mathbf{1} \mid \varphi \vee \varphi \mid \varphi \wedge \varphi \mid \neg\varphi & (\text{tests})
\end{array}$$

Table 3: Syntax of the contract language $\mathcal{CL}$.

action δ is performed $\mathcal{C}$ must hold". The other propositional operators are, as before, defined in terms of two (functionally complete set) $\{\to, \bot\}$, whereas the dynamic existential modality $\langle\delta\rangle\mathcal{C}$ is defined as the dual of box: $\langle\delta\rangle\mathcal{C} \triangleq \neg[\delta]\neg\mathcal{C}$.

In $\mathcal{CL}$ we can write *conditional* obligations, permissions and prohibitions of two different kinds. As an example let us consider conditional obligations. The first kind is represented by $[\delta]O(\alpha)$, which may be read as "after performing δ, one is obliged to do α". The second kind is modeled using the implication operator: $\mathcal{C} \to O(\alpha)$, which is read as "If $\mathcal{C}$ holds then one is obliged to perform α".

Propositional dynamic logic (PDL) makes an interplay between the actions and the formulas; i.e., it has formulas as actions (tests $\varphi?$) and it has actions defining the formulas (the box modality $[\delta]$). The intuition of the *test action* is that $\varphi?$ can be performed only if the formula φ holds in the current world. A sequence action $\varphi? \cdot \alpha$ can be viewed as a *guarded* action because α can be performed only if the test $\varphi?$ succeeds. Note that we use, what is called, *poor tests* as we do not allow for a modal formula to be a test, but only Boolean tests; i.e., we cannot ask modal questions using the dynamic or the deontic modalities.

There are two differences between the actions δ which appear inside the PDL modality $[\cdot]$ and the actions α which are allowed inside the deontic modalities. We argued before against not having the Kleene * for the α actions. Regarding the tests, if we allow *deontic test actions*, like $F(\alpha)?$ inside the deontic modalities it would break the ought-to-do approach because they introduce the formulas inside the action formalism; i.e., we could write formulas like $O_{\mathcal{C}}(F(\alpha)?)$. This constitutes a combination of ought-to-do and ought-to-be (for this direction check [41]). Moreover, adding tests inside the deontic modalities does not integrate with our way of giving semantics; we do not know how to mark obligatory (or prohibited) tests, as we do with the actions. Moreover, in case of violation, the reparation $\mathcal{C}$ is enforced in the same world as the O and the F, i.e., there is no state change. Therefore, we could reason only with the propositional logic part of $\mathcal{CL}$. The example $O_{\mathcal{C}}(F(\alpha)?)$ is read as "It is obligatory (in the current world) that the test $F(\alpha)?$ holds (in the current world), otherwise (if the test does not hold) the reparation $\mathcal{C}$ should be enforced afterwards". Compared to deontic actions, tests do not change the world: if a tests succeeds then we remain in the same world and execute the next action, if the test fails then the whole action sequence fails. We can achieve the same by only using the $\mathcal{CL}$ language as it is. The example above is specified in $\mathcal{CL}$ as $F(\alpha) \vee (\neg F(\alpha) \wedge \mathcal{C})$ which is read as above (we reword it here to match the formula better): "(In the current world) it is forbidden to do α or it is not forbidden to do α and the formula $\mathcal{C}$ holds".

$K^{\mathcal{N}}, i \models [\delta]\mathcal{C}$ iff $\forall s \in K^{\mathcal{N}}$ with $(i,s) \in R_{2^{\mathcal{A}_B}}(\delta)$ then $K^{\mathcal{N}}, s \models \mathcal{C}$.

$$\begin{aligned} R_{2^{\mathcal{A}_B}}(\delta) \;=\; \{(s,s') \mid\ & \exists k, \exists \sigma = \sigma_0 \ldots \sigma_k \text{ a final path in } A^{\mathcal{G}}(\delta), \\ & \exists s_0 \ldots s_k \in K^{\mathcal{N}} \text{ with } s_0 = s \text{ and } s_k = s', \text{ and} \\ & \forall 0 \le i \le k,\ \mathcal{V}(s_i) \in \mathcal{L}(\lceil \sigma_i \rceil), \text{ and} \\ & \forall 0 \le i < k \text{ with } \sigma_i \xrightarrow{\alpha_\times^i} \sigma_{i+1} \in \sigma \text{ then } (s_i, s_{i+1}) \in R_{2^{\mathcal{A}_B}}(\alpha_\times^i)\} \end{aligned}$$

Table 4: Semantics for $\mathcal{CL}$.

Definition 4.2 (semantics). *The semantics for the dynamic modality of $\mathcal{CL}$ is given in Table 4. The rest of the syntactic constructs of $\mathcal{CL}$ (i.e., the deontic and propositional operators) have the semantics from Table 2.*

The expression $[\delta]\mathcal{C}$ is evaluated in a state i of the normative structure $K^{\mathcal{N}}$, depending on the automata representation of the dynamic action δ. Essentially, the semantics needs to evaluate the expression $\mathcal{C}$ to true in all states s reachable from the initial state i by following the automaton $A^{\mathcal{G}}(\delta)$ of the dynamic action δ. All the states s reached from i are given by the relation described by the dynamic action, i.e., $(i,s) \in R_{2^{\mathcal{A}_B}}(\delta)$. The relation $R_{2^{\mathcal{A}_B}}(\delta)$ is not as simple to describe as was the case with the deontic actions where we needed to look only at single steps. In the case of dynamic actions we need to look several steps in the structure. We take the approach introduced in [42] and use the automata $A^{\mathcal{G}}(\delta)$, as in Definition 2.14, interpreting the dynamic actions δ.

The relation $R_{2^{\mathcal{A}_B}}(\delta)$ is defined as the set of all pairs of states (s,s') having the property that there is an accepting path σ in $A^{\mathcal{G}}(\delta)$ that is matched by a sequence of states in $K^{\mathcal{N}}$. A sequence of states matches the path σ iff all the edges $\sigma_i \xrightarrow{\alpha_\times} \sigma_{i+1} \in \sigma$ are matched by the corresponding transitions $(s_i, s_{i+1}) \in R_{2^{\mathcal{A}_B}}(\alpha_\times)$ (i.e., the indexes have to match as well as the labels $\alpha_\times$) and the valuation for the states has to conform with the sets of atoms of the corresponding nodes on the path (i.e., $\mathcal{V}(s_i) \in \mathcal{L}(\lceil \sigma_i \rceil)$). From this matching sequence of states take the first and the final state as the pair we are looking for. The conformance test $\mathcal{V}(s_i) \in \mathcal{L}(\lceil \sigma_i \rceil)$ is required to ensure than any test action from δ is satisfied at the particular state; i.e., if the valuation of the state corresponds to one of the atoms (atoms are encodings of valuations) that are encoded by the automaton $\lceil \sigma_i \rceil$ of the node σ_i.

Example 4.1 Consider the normative structure of Fig. 4 changed s.t. the transition (s_1, s_4) is labeled only by $\{p\}$. Also consider the automaton $A^{\mathcal{G}}(\delta)$ from Fig. 3 corresponding to the dynamic action $\delta = (p \cdot b)^* + (d \times n) \cdot \varphi? \cdot p \cdot p$. This time we assume that the complex test $\varphi?$ is a conjunction that contains ϕ'; therefore this test fails in all states where the propositional constant ϕ' is not present. We want to check if the formula $[\delta]\phi$ holds in state s_1. The automaton $A^{\mathcal{G}}(\delta)$ has the following final paths: $\{(r, t_1, t_2, t_3), (r, t_6), (r, t_4, t_5), (r, t_4, t_5, (t_4, t_5)^*)\}$. We calculate $R_{2^{\mathcal{A}_B}}(\delta) = \{(s_1, s_1)\}$. At a closer look, the path (r, t_1, t_2, t_3) does not contribute to the $R_{2^{\mathcal{A}_B}}(\delta)$ because $\mathcal{V}(s_2) \notin \mathcal{L}(\lceil t_1 \rceil)$ (i.e., the automaton $\lceil t_1 \rceil$ accepts only atoms that make φ true, but $\mathcal{V}(s_2)$ makes ϕ' false, hence φ false). Also path (r, t_6) does not contribute, but the paths $(r, t_4, t_5, (t_4, t_5)^*)$ are matched by sequences of states s_1, s_4, s_1 and $s_1, s_4, s_1, (s_4, s_1)^*$. Therefore, $s_1 \models [\delta]\phi$ because $s_1 \models \phi$. Consider now a slight modification $\delta' = (p \cdot b)^* + (d \times n) \cdot p \cdot p$ (i.e., the test $\varphi?$ does not appear). In this case the language $\mathcal{L}(\lceil t_1 \rceil)$ is the universal language (i.e., all the atoms) and hence, the path (r, t_1, t_2, t_3) now contributes to $R_{2^{\mathcal{A}_B}}(\delta)$ by adding the pair (s_1, s_4). Because of this $s_1 \not\models [\delta']\phi$ since one of the pairs in $R_{2^{\mathcal{A}_B}}(\delta)$ does not respect the condition of the semantics; i.e., $s_4 \not\models \phi$. Consider a subsequent modification $\delta'' = (p \cdot b)^* + (d{\times}n) \cdot p$ (i.e., the last p action is removed). The automaton $A^{\mathcal{G}}(\delta)$ from Fig. 3 has the node t_3 removed, and thus the final path (r, t_1, t_2) contributed to $R_{2^{\mathcal{A}_B}}(\delta)$ with the pair (s_1, s_3). In this case $s_1 \models [\delta'']\phi$ because in both s_1 and s_3 the formula ϕ holds. Consider a last modification of δ'' to $\delta''' = (p \cdot \phi'? \cdot b)^* + (d{\times}n) \cdot p$. In this case all the paths $(r, t_4, t_5, (t_4, t_5)^*)$ cannot contribute to $R_{2^{\mathcal{A}_B}}(\delta)$ any more because the valuation $\mathcal{V}(s_4)$ does not make ϕ' true and hence is not part of $\mathcal{L}(\lceil t_4 \rceil)$ which contains only those atoms that make ϕ' true; nevertheless $s_1 \models [\delta''']\phi$ since in the only remaining state s_3 we have $s_3 \models \phi$. □

4.1. Properties of the $\mathcal{CL}$ logic

The validities and non-validities results for the deontic modalities of Section 3.2 hold for $\mathcal{CL}$ also. Adding the dynamic modality does not affect these. Besides, we have extra properties that deal with the combination of deontic and dynamic modalities.

Denote by $\mathbf{any} = +_{\alpha_\times \in \mathcal{A}_B^\times} \alpha_\times$ the choice between *all* the $\times$-actions. For all $\alpha_\times \in \mathcal{A}_B^\times$ denote by $\langle\langle \alpha_\times \rangle\rangle\mathcal{C}$ the formula $\langle \alpha_\times \times \mathbf{any} \rangle \mathcal{C}$ and by $[[\alpha_\times]]\mathcal{C}$ the formula $[\alpha_\times \times \mathbf{any}]\phi$. Note that $\langle\langle \cdot \rangle\rangle$ and $[[\cdot]]$ are duals in this definition. Extend $[[\cdot]]$ to all actions $\alpha \in \mathcal{A}$, as is done in the standard PDL (the definition for $\langle\langle \cdot \rangle\rangle$ is analogous):

$$
\begin{aligned}
[[\alpha + \beta]]\mathcal{C} &= [[\alpha]]\mathcal{C} \wedge [[\beta]]\mathcal{C} \\
[[\alpha \cdot \beta]]\mathcal{C} &= [[\alpha]][[\beta]]\mathcal{C} \\
[[\alpha^*]]\mathcal{C} &= \mathcal{C} \wedge [[\alpha]][[\alpha^*]]\mathcal{C}
\end{aligned}
$$

Remark that the syntactic construct $[[\cdot]]$ enhances the $[\cdot]$ only locally, i.e., only for one step moves. It enhances in the sense that it contains not only those single transitions labeled by $\{\alpha_\times\}$ but also those labeled by set labels that include $\{\alpha_\times\}$, i.e., are bigger, up to the biggest label $\mathcal{A}_B$.

An important requirement when modeling electronic contracts is that the obligation of a sequence of actions $O_{\mathcal{C}}(\alpha \cdot \alpha')$ must be equal to the obligation of the first action $O_{\mathcal{C}}(\alpha)$ and after the first obligation is respected the second obligation must hold $O_{\mathcal{C}}(\alpha')$. To respect the obligation $O_{\mathcal{C}}(\alpha)$ means to do any action bigger than α which is captured with the syntactic construction $[[\cdot]]$. Note that if $O_{\mathcal{C}}(\alpha)$ is violated then the reparation $\mathcal{C}$ must be enforced (must hold) and the second obligation is discarded, i.e., is not necessarily enforced.

Proposition 4.3. *The following statements hold:*

$$\models [[\alpha_\times]]\mathcal{C} \rightarrow [[\alpha_\times \times \beta_\times]]\mathcal{C} \tag{28}$$

$$\models O_{\mathcal{C}}(\alpha \cdot \beta) \leftrightarrow O_{\mathcal{C}}(\alpha) \wedge [[\alpha]]O_{\mathcal{C}}(\beta) \tag{29}$$

$$\models F_{\mathcal{C}}(\alpha \cdot \beta) \leftrightarrow F_{\top}(\alpha) \wedge [[\alpha]]F_{\mathcal{C}}(\beta) \tag{30}$$

where $\alpha_\times, \beta_\times \in \mathcal{A}_B^\times$.

Proof: The proof of (28) is easy as we are concerned only with $\times$-actions. Trivially, for an action $\alpha_\times$ the tree of the action $\alpha_\times \times \mathbf{any}$ has all edges with labels including $\alpha_\times$ (the tree has height 1). Similarly, all the edges of the tree of $\alpha_\times \times \beta_\times \times \mathbf{any}$ contain $\alpha_\times \times \beta_\times$ and, because $\alpha_\times \subseteq \alpha_\times \times \beta_\times$, they contain also $\alpha_\times$. Because of these, all the transitions in the structure that are relevant for evaluating $[[\alpha_\times \times \beta_\times]]\mathcal{C}$ are part of the transitions relevant for $[[\alpha_\times]]\mathcal{C}$ and, hence, $\mathcal{C}$ holds in all their ending states. Proof finished as whenever $[[\alpha_\times]]\mathcal{C}$ holds in a state, $[[\alpha_\times \times \beta_\times]]\mathcal{C}$ holds too.

To prove (29) we need a series of results which are easy to check but tedious; we just state these results. To prove the left to right implication it is easy to see that $O_\mathcal{C}(\alpha \cdot \beta) \to O_\mathcal{C}(\alpha)$ as we discuss further. Trivially, $T(\alpha) \subset T(\alpha \cdot \beta)$ from which it is easy to deduce that if $T(\alpha \cdot \beta)\ \mathcal{S}_i\ K^\mathcal{N}$ then $T(\alpha)\ \mathcal{S}_i\ K^\mathcal{N}$ (i.e., the first line in the semantics of $O_\mathcal{C}(\alpha)$). From the same results above it is clear that all the transitions $s \xrightarrow{\gamma'} s' \in K^\mathcal{N}$ that are relevant for the semantics of $O_\mathcal{C}(\alpha)$ are among the transitions that are relevant in the semantics of $O_\mathcal{C}(\alpha \cdot \beta)$ and hence they respect the second condition in the semantics of obligation. A second result easy to verify is that $K_{rem}^{T(\alpha),i} \subset K_{rem}^{T(\alpha \cdot \beta),i}$ which implies trivially the third condition in the semantics of $O_\mathcal{C}(\alpha)$. Related to this result is that $T(\overline{\alpha}) \subseteq T(\overline{\alpha \cdot \beta})$ which means that $\mathit{leafs}(T(\overline{\alpha})) \subseteq \mathit{leafs}(T(\overline{\alpha \cdot \beta}))$ which makes the last requirement in the semantics of $O_\mathcal{C}(\alpha)$ trivially true.

To finish the left to right implication we prove $O_\mathcal{C}(\alpha \cdot \beta) \to [[\alpha]]O_\mathcal{C}(\beta)$. Recall that the construction of $T(\alpha \cdot \beta)$ first constructs $T(\alpha)$ and $T(\beta)$ and then just attaches the whole $T(\beta)$ to all the leafs of $T(\alpha)$ (i.e., replaces each leaf with the root of $T(\beta)$). The action α is a deontic action and, hence, its interpretation is a tree and the semantics of $[[\cdot]]$ follows all the transitions in $K^\mathcal{N}$ that are bigger than the edges of this tree. Therefore, these are all the transitions that participate in the $T(\alpha \cdot \beta)\ \mathcal{S}_i\ K^\mathcal{N}$; actually in the second condition of the simulation relation. This simple observation gives all the rest of the proof; it implies that all the states where we have to evaluate $O_\mathcal{C}(\beta)$ are part of $T(\alpha \cdot \beta)\ \mathcal{S}_i\ K^\mathcal{N}$, actually $T(\beta)\ \mathcal{S}_s\ K^\mathcal{N}$ where s is related to the leafs of $T(\alpha)$. The second and third conditions in the semantics of $O_\mathcal{C}(\beta)$ in the states s follow similarly. The last condition holds because $T(\overline{\beta}) \subseteq T(\overline{\alpha \cdot \beta})$.

The proof of the right to left implication follows a similar tedious argument but the main intuition is as follows. To get the semantics of $O_\mathcal{C}(\alpha \cdot \beta)$ we need to achieve two main goals: (1) to walk on the $K^\mathcal{N}$ structure according to $T(\alpha \cdot \beta)$ and to find all the appropriate $\circ$ markers, (2) $\mathcal{C}$ must hold at the appropriate violating states. Walking on $K^\mathcal{N}$ goes well and finds all the necessary markers until reaching the leaf nodes of the first part of the tree, i.e., of $T(\alpha)$, because it comes from the semantics of $O_\mathcal{C}(\alpha)$. Nevertheless, we can continue because all these states are the same as the states reached through $[[\alpha]]$. Therefore, because of the semantics of $O_\mathcal{C}(\beta)$ from all these states we can continue until we reach the leaf nodes of the big tree $T(\alpha \cdot \beta)$. For the second part it is easy to see that all the states of $K^\mathcal{N}$ reached because of the tree $T(\overline{\alpha \cdot \beta})$ are the same as the states reached because of the tree $T(\overline{\alpha})$ together with those reach through making first α and then following $T(\overline{\beta})$.

The proof of (30) is similar only that it reasons about partial simulations. To remark is that the first prohibition has a trivial reparation. From a practical point of view the prohibition is irrelevant because it does not impose any restrictions. Technically, the reparation $\mathcal{C}$ holds (is enforced) only in the states corresponding to the leafs of the tree

$T(\alpha \cdot \beta)$, therefore, there is no information about what holds at the leafs of $T(\alpha)$. $\square$

The validity (30) hints at the fact that prohibitions as defined in $\mathcal{CL}$ could be modeled using the dynamic operator $[\cdot]$ (or maybe using the syntactic construct $[[\cdot]]$), but this is a simply a conjecture at the moment of writing this paper.

The PDL version with automata inside the dynamic modality from [42] is proven decidable with a method that builds finite models based on a variant of Fischer-Ladner closure and using Hintikka-like sets. This proof can be easily adapted to our automata over guarded synchronous strings and to our dynamic modality over synchronous actions. Therefore, if we consider only the propositional part and the dynamic modality of $\mathcal{CL}$ we have a decidable extension of PDL which can talk about synchronous actions. Unfortunately, we could not give a proof of decidability for the deontic modalities using a Fischer-Ladner closure method. Therefore, we cannot combine the proof based on finite tree models for the deontic modalities with a proof based on Fischer-Ladner closure for the dynamic modality to obtain the decidability of the full $\mathcal{CL}$.

On the other hand, the PDL logic does not have the finite tree model property because of the Kleene *. This applies to our dynamic modality over synchronous actions too. However, we show in Theorem 4.4 that the dynamic modality over synchronous actions has the tree model property. This together with Theorem 3.9 give the tree model property for full $\mathcal{CL}$. From the tree model one just needs to find the right selection method to obtain a bounded tree model property (i.e., bounded branching) as was done for the modal μ-calculus [43, 44]. Using this, one can prove decidability by a standard translation into the SnS logic which is decidable [45]. Although standard, these techniques are specific and quite involved. In consequence we left the work of investigating and adapting these techniques to the setting of $\mathcal{CL}$ as an open problem for future work.

Theorem 4.4 (tree model for $\mathcal{CL}$). *For a pointed structure $\langle K^{\mathcal{N}}, i\rangle$ we have:*

$$TK^{\mathcal{N}}, x \models [\delta]\mathcal{C} \quad \textit{iff} \quad K^{\mathcal{N}}, \rho(x) \models [\delta]\mathcal{C} \tag{31}$$

Proof: The proof follows the semantics of $[\cdot]$ and uses an argument similar to what we did in the proof of Theorem 3.9 for the last part of the case for obligations. This means we use structural induction and assume $TK^{\mathcal{N}}, x' \models \mathcal{C}$ *iff* $K^{\mathcal{N}}, \rho(x') \models \mathcal{C}$.

For the left to right implication we use *reductio ad absurdum* and assume that $\exists s \in K^{\mathcal{N}}$ s.t. $(\rho(x), s) \in R_{2^{\mathcal{A}_B}}(\delta)$ for which $K^{\mathcal{N}}, s \not\models \mathcal{C}$. Having $(\rho(x), s) \in R_{2^{\mathcal{A}_B}}(\delta)$ it means that $\exists \sigma = \sigma_0 \dots \sigma_k$ a final path in $A^{\mathcal{G}}(\delta)$ and $\exists s_0 \dots s_k \in K^{\mathcal{N}}$ s.t. $s_0 = \rho(x)$, $s_k = s$, $\forall 0 \leq i \leq k$, $\mathcal{V}(s_i) \in \mathcal{L}(\lceil \sigma_i \rceil)$, and for any $\sigma_i \xrightarrow{\alpha_\times^i} \sigma_{i+1} \in \sigma$ we have $(s_i, s_{i+1}) \in R_{2^{\mathcal{A}_B}}(\alpha_\times^i)$. By Lemma 3.8 it means that for $(\rho(x), s_1) \in R_{2^{\mathcal{A}_B}}(\alpha_\times^0)$ we find $(x, x\alpha_\times^0 s_1) \in R^T_{2^{\mathcal{A}_B}}(\alpha_\times^0)$; and the same for all i, ending with the transition $(x\alpha_\times^0 \dots s_{k-1}, x\alpha_\times^0 \dots \alpha_\times^{k-1} s_k) \in R^T_{2^{\mathcal{A}_B}}(\alpha_\times^{k-1})$. Because the valuation functions $\mathcal{V}$ and $\mathcal{V}^T$ agree on all propositional constants ϕ it means that $\mathcal{V}^T(x\alpha_\times^0 \dots s_i) \in \mathcal{L}(\lceil \sigma_i \rceil)$. In this way we have found for the final path σ the sequence of states $x, x\alpha_\times^0 s_1, \dots, x\alpha_\times^0 \dots \alpha_\times^{k-1} s_k$ in the tree model that satisfy the conditions for $(x, x\alpha_\times^0 \dots \alpha_\times^{k-1} s_k) \in$

$R^T_{2^{\mathcal{A}_B}}(\delta)$ and, by the left part of the implication, we have $TK^{\mathcal{N}}, x\alpha^0_\times \ldots \alpha^{k-1}_\times s_k \models \mathcal{C}$. We use the inductive hypothesis to obtain $K^{\mathcal{N}}, \rho(x\alpha^0_\times \ldots \alpha^{k-1}_\times s_k) \models \mathcal{C}$ which is the same as $K^{\mathcal{N}}, s_k \models \mathcal{C}$; but $s_k = s$ and hence we get the contradiction.

The right to left implication follows analogous arguments. □

In the following we argue that the most important paradoxes of deontic logic are avoided in $\mathcal{CL}$, either because they are not expressible in the language or because they are excluded by the semantics.

Ross's Paradox [46] in natural language it is expressed as: *a. "It is obligatory that one mails the letter"*; *b. "It is obligatory that one mails the letter or one burns the letter"*. In SDL these are expressed as: *a.* $O(p)$, *b.* $O(p \vee q)$. The problem is that in SDL one can make the inference $O(p) \rightarrow O(p \vee q)$.

Remark 4.5. *Ross's paradox does not hold in $\mathcal{CL}$.*

Argumentation : Basically, Ross's paradox says that it is counter intuitive to have $O(a) \rightarrow O(a + b)$ (e.g., *"Obligation to drink implies obligation to drink or to kill"*). In $\mathcal{CL}$ this inference is not possible as witnessed by Proposition 3.25(22).

The Good Samaritan Paradox [47] in natural language it is expressed as: *a. "It ought to be the case that Jones helps Smith who has been robbed"*; *b. "It ought to be the case that Smith has been robbed"*; and the natural inference *c. "Jones helps Smith who has been robbed if and only if Jones helps Smith and Smith has been robbed"*. In SDL the first two are expressed as: *a.* $O(p \wedge q)$, *b.* $O(q)$. The problem is that in SDL one can derive that $O(p \wedge q) \rightarrow O(q)$ which is counter intuitive in the natural language.

Remark 4.6. *The Good Samaritan paradox can not be expressed in $\mathcal{CL}$.*

Argumentation : The Good Samaritan paradox uses *ought-to-be* and is more delicate to transform it into our *ought-to-do* approach. The transformation looks like: $\varphi \rightarrow O(a)$ which means that *"If Smith has been robbed (i.e., φ) then John is obliged to help Smith (i.e., $O(\alpha)$)"*. We can not express in $\mathcal{CL}$ obligations over conjunction of two actions that are not performed concurrently as this paradox is expressed in SDL. Also, with our representation of the paradox we cannot deduce φ; i.e., that *Smith has been robbed*.

The Free Choice Permission Paradox [46] in natural language it is expressed as: *a. "You may either sleep on the sofa or sleep on the bed"*; *b. "You may sleep on the sofa and you may sleep on the bed"*. In SDL this is: *a.* $P(p \vee q)$, *b.* $P(p) \wedge P(q)$. The natural intuition tells that $P(p \vee q) \rightarrow P(p) \wedge P(q)$. In SDL this leads to $P(p) \rightarrow P(p \vee q)$ which is $P(p) \rightarrow P(p) \wedge P(q)$, so $P(p) \rightarrow P(q)$. As an example: *"If one is permitted something, then one is permitted anything"*.

Remark 4.7. *The Free Choice Permission paradox does not exist in $\mathcal{CL}$.*

Argumentation: The Free Choice Permission paradox basically says that from having one permission we may infer that we have any permission. That is: $P(a) \rightarrow P(a+b)$ or $P(a) \rightarrow P(a) \wedge P(b)$. Neither of the two implications hold in our approach. The second one is obvious. The first one is ruled out by Corollary 3.25-(27) which is a consequence of Proposition 3.24-(17).

Sartre's Dilemma [48] in natural language is expressed as: *a.* It is obligatory to meet Jones now (as promised to Jones); *b.* It is obligatory to not meet Jones now (as promised to Smith). In SDL this is: *a.* $O(p)$, *b.* $O(\neg p)$. The problem is that in the natural language the two obligations are intuitive and often happen, where the logical formulas are inconsistent when put together (in conjunction) in SDL.

Remark 4.8. *Sartre's Dilemma is not expressible in our approach.*

Argumentation: In $\mathcal{CL}$ we cannot write negation of actions, like *not meet Jones*, and thus cannot have obligations on top. Thus, syntactically is not possible to write this paradox in $\mathcal{CL}$. However, Sartre's dilemma can be reformulated in contracts terminology as: *Obliged to meet John and Forbidden to meet John*. This is written in $\mathcal{CL}$ as $O(a) \wedge F(a)$ which is a well formed formula. But this results in a contradiction because of Corollary 3.23-(12). In conclusion, neither this reformulation that can be represented in $\mathcal{CL}$ does not constitute a paradox because the formula does not hold.

Chisholm's Paradox [49] in natural language is expressed as: *a.* John ought to go to the party; *b.* If John goes to the party then he ought to tell them he is coming; *c.* If John does not go to the party then he ought not to tell them he is coming; *d.* John does not go to the party. In SDL these are expressed as: *a.* $O(p)$, *b.* $O(p \rightarrow q)$, *c.* $\neg p \rightarrow O(\neg q)$, *d.* $\neg p$. The problem is that in SDL one can infer $O(q) \wedge O(\neg q)$ which is due to statement *b*.

Remark 4.9. *The Chisholm's paradox is avoided in $\mathcal{CL}$.*

Argumentation: The propositions of the Chisholm's paradox are expressed in $\mathcal{CL}$ as: *a.* $O(a)$, *b.* $[a]O(b)$, *c.* $[\overline{a}]O(\overline{b})$. Note first that formulas *a.* and *c.* give the CTD formula $O_{\mathcal{C}}(a)$ of $\mathcal{CL}$ where $\mathcal{C} = O(\overline{b})$. The problem in SDL was that one may infer both $O(b)$ and $O(\overline{b})$ holding in the same world. This is not our case because $O(b)$ holds only after doing action a, where $O(\overline{b})$ holds only after doing the contradictory action $\overline{a}$. Therefore, we can not have in the same world both $O(b)$ and $O(\overline{b})$.

5. Conclusion

In this paper we have presented the action-based contract logic $\mathcal{CL}$. As our objective has been to present the theoretical background of $\mathcal{CL}$ including its Kripke-like semantics and further results concerning the decidability of (fragments of) the logic, we could not expand on applications. Case studies on how to use $\mathcal{CL}$ for writing specifications of contracts are presented elsewhere (e.g. see [40, 50]).

Concerning verification, an encoding of a variant of the logic presented here into NuSMV has been presented in [40], while the papers [50, 51] present the theory and a tool for conflict analysis. With the development of a Kripke semantics, we are now in conditions to develop a specific model checker for $\mathcal{CL}$.

Besides the development of a model checker, future work includes the development of a proof system, further investigation on full decidability of the logic, the study of the use of $\mathcal{CL}$ as a semantic framework for other languages for services lacking formal semantics, and an extension with real-time. The latter in particular is appealing as most real contracts contain timing constraints.

Acknowledgements: We thank Martin Steffen and Olaf Owe for useful comments on earlier versions of this work.

References

[1] G. H. V. Wright, Deontic logic, Mind 60 (1951) 1–15.

[2] M. J. Fischer, R. E. Ladner, Propositional modal logic of programs, in: 9th ACM Symposium on Theory of Computing (STOC'77), ACM, 1977, pp. 286–294.

[3] C. Prisacariu, G. Schneider, A formal language for electronic contracts, in: FMOODS'07, Vol. 4468 of LNCS, Springer, 2007, pp. 174–189.

[4] C. Prisacariu, G. Schneider, CL: An Action-based Logic for Reasoning about Contracts, in: Workshop on Logic, Language, Informations and Computation (WOLLIC'09), Vol. 5514 of LNCS, Springer, 2009, pp. 335–349.

[5] M. Kyas, C. Prisacariu, G. Schneider, Run-time Monitoring of Electronic Contracts, in: ATVA'08, Vol. 5311 of LNCS, Springer, 2008, pp. 397–407.

[6] C. Prisacariu, A Dynamic Deontic Logic over Synchronous Actions, Ph.D. thesis, Department of Informatics, University of Oslo (2010).

[7] C. Prisacariu, Synchronous Kleene Algebra, The Journal of Logic and Algebraic Programming (JLAP) 79 (7) (2010) 608–635.

[8] G. H. Von Wright, An Essay in Deontic Logic and the General Theory of Action, North Holland Publishing Co., Amsterdam, 1968.

[9] J.-J. C. Meyer, A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic, Notre Dame Journal of Formal Logic 29 (1) (1988) 109–136.

[10] J. Broersen, R. Wieringa, J.-J. C. Meyer, A fixed-point characterization of a deontic logic of regular action, Fundam. Inf. 48 (2-3) (2001) 107–128.

[11] J.-J. C. Meyer, F. Dignum, R. Wieringa, The paradoxes of deontic logic revisited: A computer science perspective, Tech. Rep. UU-CS-1994-38, Department of Information and Computing Sciences, Utrecht University (1994).

[12] K. Segerberg, A deontic logic of action, Studia Logica 41 (2) (1982) 269–282.

[13] K. Segerberg, Getting started: Beginnings in the logic of action, Studia Logica 51 (3/4) (1992) 347–378.

[14] A. Z. Wyner, Sequences, obligations, and the contrary-to-duty paradox, in: L. Goble, J.-J. C. Meyer (Eds.), 8th International Workshop on Deontic Logic in Computer Science (DEON'06), Vol. 4048 of Lecture Notes in Computer Science, Springer, 2006, pp. 255–271.

[15] R. Van der Meyden, Dynamic logic of permission, the, in: J. Mitchell (Ed.), 5th Annual IEEE Symp. on Logic in Computer Science, (LICS'90), IEEE Computer Society Press, 1990, pp. 72–78.

[16] P. F. Castro, T. Maibaum, A complete and compact propositional deontic logic, in: C. B. Jones, Z. Liu, J. Woodcock (Eds.), ICTAC 2007, Vol. 4711 of LNCS, 2007, pp. 109–123.

[17] D. Harel, D. Kozen, J. Tiuryn, Dynamic Logic, MIT Press, 2000.

[18] C. Lutz, D. Walther, PDL with negation of atomic programs., in: D. A. Basin, M. Rusinowitch (Eds.), 2nd International Joint Conference on Automated Reasoning (IJCAR'04), Vol. 3097 of LNCS, Springer, 2004, pp. 259–273.

[19] J. Broersen, Modal action logics for reasoning about reactive systems, Ph.D. thesis, Vrije Universiteit Amsterdam (2003).

[20] M. Ben-Ari, J. Y. Halpern, A. Pnueli, Finite models for deterministic propositional dynamic logic., in: S. Even, O. Kariv (Eds.), 8th Colloquium On Automata, Languages and Programming (ICALP'81), Vol. 115 of LNCS, Springer, 1981, pp. 249–263.

[21] R. Milner, Calculi for synchrony and asynchrony., Theorethical Computer Science 25 (1983) 267–310.

[22] G. Berry, The foundations of Esterel, in: Proof, language, and interaction: essays in honour of Robin Milner, MIT Press, 2000, pp. 425–454.

[23] R. Milner, Communication and concurrency, Prentice Hall, 1995.

[24] C. A. R. Hoare, Communicating Sequential Processes, Prentice Hall, 1985.

[25] D. Harel, Recurring dominoes: Making the highly undecidable highly understandable (preliminary report), in: M. Karpinski (Ed.), Fundamentals of Computation Theory (FCT'83), Vol. 158 of LNCS, Springer, 1983, pp. 177–194.

[26] D. Peleg, Concurrent dynamic logic (extended abstract), in: 7th ACM Symposium on Theory of Computing (STOC'85), ACM, 1985, pp. 232–239.

[27] D. Peleg, Concurrent dynamic logic, Journal of ACM 34 (2) (1987) 450–479. doi:10.1145/23005.23008.

[28] A. K. Chandra, D. Kozen, L. J. Stockmeyer, Alternation, J. ACM 28 (1) (1981) 114–133.

[29] H. Prakken, M. Sergot, Dyadic deontic logic and contrary-to-duty obligation, in: D. Nute (Ed.), Defeasible Deontic Logic, Kluwer Academic Publishers, 1997, pp. 223–262.
URL `citeseer.ist.psu.edu/article/prakken97dyadic.html`

[30] J. Carmo, A. Jones, Deontic logic and contrary-to-duties, in: D. Gabbay, F. Guenthner (Eds.), Handbook of Philosophical Logic, Kluwer Academic Publishers, 2002, pp. 265–343.

[31] R. Trypuz, P. Kulicki, Towards Metalogical Systematisation of Deontic Action Logics Based on Boolean Algebra, in: G. Governatori, G. Sartor (Eds.), 10th International Conference on Deontic Logic in Computer Science (DEON'10), Vol. 6181 of Lecture Notes in Computer Science, Springer, 2010, pp. 132–147.

[32] F. Baader, T. Nipkow, Term Rewriting and All That, Cambridge University Press, 1998.

[33] V. R. Pratt, Process logic., in: 6th Symposium on Principles of Programming Languages (POPL'79), ACM, 1979, pp. 93–100.

[34] M. Y. Vardi, Why is modal logic so robustly decidable?, in: N. Immerman, P. G. Kolaitis (Eds.), Descriptive Complexity and Finite Models, Vol. 31 of DIMACS Series in Discrete Mathematics and Theoretical Computer Science, American Mathematical Society, 1996, pp. 149–184.

[35] E. Grädel, Why are modal logics so robustly decidable?, in: Current Trends in Theoretical Computer Science, 2001, pp. 393–408.

[36] P. Blackburn, M. de Rijke, Y. Venema, Modal Logic, Vol. 53 of Cambridge Tracts in Theoretical Computer Science, Cambridge Univ. Press, 2001.

[37] H. Sahlqvist, Correspondence and completeness in the first- and second-order semantics for modal logic, in: S. Kanger (Ed.), Proceedings of the Third Scandanavian Logic Symposium, North Holland, Amsterdam, 1975, pp. 110–143.

[38] S. Bursuc, C. Prisacariu, Unification and Matching in Separable Theories (extended abstract), in: M. Fernandez (Ed.), 24th International Workshop on Unification (UNIF10), Edimburg, UK, 2010.

[39] S. Bursuc, C. Prisacariu, Unification and Matching in Separable Theories - technicalities, Tech. Rep. 398, Department of Informatics, University of Oslo, Norway (July 2010).

[40] G. Pace, C. Prisacariu, G. Schneider, Model checking contracts - a case study, in: K. Namjoshi, T. Yoneda (Eds.), 5th International Symposium on Automated Technology for Verification and Analysis (ATVA'07), Vol. 4762 of LNCS, Springer, 2007, pp. 82–97.

[41] P. d'Altan, J.-J. C. Meyer, R. Wieringa, An integrated framework for ought-to-be and ought-to-do constraints, Artif. Intell. Law 4 (2) (1996) 77–111.

[42] D. Harel, R. Sherman, Propositional dynamic logic of flowcharts, in: M. Karpinski (Ed.), Fundamentals of Computation Theory (FCT'83), Vol. 158 of LNCS, Springer, 1983, pp. 195–206.

[43] D. Kozen, R. Parikh, A decision procedure for the propositional μ-calculus., in: E. M. Clarke, D. Kozen (Eds.), 4th Workshop on Logics of Programs, Vol. 164 of LNCS, Springer, 1983, pp. 313–325.

[44] R. S. Streett, E. A. Emerson, The Propositional Mu-Calculus is Elementary, in: 11^{th} International Colloquium on Automata, Languages and Programming (ICALP'84), Vol. 172 of LNCS, Springer, 1984, pp. 465–472.

[45] M. O. Rabin, Decidability of second-order theories and automata on infinite trees, Trans. Amer. Math. Soc. 141 (1969) 1–35.

[46] A. Ross, Imperatives and logic, Theoria 7 (1941) 53–71.

[47] A. N. Prior, Escapism: The logical basis of ethics, in: A. I. Melden (Ed.), Essays in Moral Philosophy, 1958, pp. 135–146.

[48] P. McNamara, Deontic logic, in: D. M. Gabbay, J. Woods (Eds.), Handbook of the History of Logic, Vol. 7, North-Holland Publishing, 2006, pp. 197–289.

[49] R. M. Chisholm, Supererogation and offence: A conceptual scheme for ethics, Ratio Juris 5 (1963) 1–14.

[50] S. Fenech, G. J. Pace, G. Schneider, Automatic Conflict Detection on Contracts, in: 6th International Colloquium on Theoretical Aspects of Computing (ICTAC'09), Vol. 5684 of LNCS, Springer, 2009, pp. 200–214.

[51] S. Fenech, G. J. Pace, G. Schneider, Clan: A tool for contract analysis and conflict discovery, in: 7th International Symposium on Automated Technology for Verification and Analysis (ATVA'09), Vol. 5799 of LNCS, Springer, 2009, pp. 90–96.

A. Additional Proofs

The following lemmas are helper results used in the proof of Theorem 3.21.

The following lemma guarantees that the conjunction of obligations implies equality between the structures of the conjuncts, or strict inclusion of one into the other.

Lemma A.1. *If $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then $K_{max}^{T(\alpha),i} = K_{max}^{T(\beta),i}$ otherwise $K_{max}^{T(\alpha),i} \subset K_{max}^{T(\beta),i}$ otherwise $K_{max}^{T(\alpha),i} \supset K_{max}^{T(\beta),i}$.*

Proof : Take an arbitrary pointed structure $K^{\mathcal{N}}, i$ and suppose $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$. The proof of this lemma uses *reductio ad absurdum* and is based on the fact that lines two and three in the semantics of obligation add $\circ$ markers to the states, and line four removes $\circ$ markers thus resulting in a contradiction.

If $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha)$ and $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\beta)$. From the first we have by the semantics that $T(\alpha)\ \mathcal{S}_i\ K^{\mathcal{N}}$ which means that there exists the maximal simulating structure $K_{max}^{T(\alpha),i}$. From the semantics of $O_{\mathcal{C}}(\beta)$ we obtain similarly $K_{max}^{T(\beta),i}$. Both maximal simulating structures are substructures of the same $K^{\mathcal{N}}$.

Suppose that there exists a transition $k \xrightarrow{\gamma} k' \in K_{max}^{T(\alpha),i}$ s.t. $k \xrightarrow{\gamma} k' \notin K_{max}^{T(\beta),i}$ and there is a transition $s \xrightarrow{\gamma'} s' \in K_{max}^{T(\beta),i}$ s.t. $s \xrightarrow{\gamma'} s' \notin K_{max}^{T(\alpha),i}$. Without loss of generality we will work with the transition $k \xrightarrow{\gamma} k'$ which from the semantics of $O_{\mathcal{C}}(\alpha)$ we have that $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \in \varrho(k')$. On the other hand the transition $k \xrightarrow{\gamma} k'$ is not part of $K_{max}^{T(\beta),i}$ and because $k \in K_{max}^{T(\beta),i}$ and we know that it exists at least one transition in $K_{max}^{T(\beta),i}$ (for example the transition $s \xrightarrow{\gamma'} s'$) then it means that $k \xrightarrow{\gamma} k' \in K_{rem}^{T(\beta),i}$. By the semantics of $O_{\mathcal{C}}(\beta)$ we know that $\forall a \in \mathcal{A}_B$ if $a \in \gamma$ then $\circ_a \notin \varrho(k')$. This results in a contradiction and therefore the initial supposition is wrong. $\square$

Corollary A.2.

a. If $K_{max}^{T(\alpha),i} = K_{max}^{T(\beta),i}$ then

(a) $TK_{max}^{T(\alpha),i} = TK_{max}^{T(\beta),i}$ and

(b) $K_{rem}^{T(\alpha),i} = K_{rem}^{T(\beta),i}$.

b. If $K_{max}^{T(\alpha),i} \subset K_{max}^{T(\beta),i}$ then

(a) $TK_{max}^{T(\alpha),i} \subset TK_{max}^{T(\beta),i}$ and

(b) $\forall k \xrightarrow{\gamma} k' \in K_{rem}^{T(\alpha),i}$ either $k \xrightarrow{\gamma} k' \in K_{rem}^{T(\beta),i}$ or $k \xrightarrow{\gamma} k' \in K_{max}^{T(\beta),i}$.

c. If $K_{max}^{T(\alpha),i} \supset K_{max}^{T(\beta),i}$ then

the same as before but interchange α with β.

Lemma A.3. *For any $\alpha, \beta, \gamma', \gamma'' \in \mathcal{A}^{\mathcal{D}}$ if $T(\alpha \times \gamma') = T(\beta \times \gamma'') = T$ then $\exists \gamma''' \in \mathcal{A}^{\mathcal{D}}$ s.t. $T = T(\alpha \times \beta \times \gamma''')$.*

Proof : From the completeness result of the algebra of actions we get that because $T(\alpha \times \gamma') = T(\beta \times \gamma'')$ we have $\alpha \times \gamma' = \beta \times \gamma'' = \theta$. We need to prove that $\exists \gamma''' \in \mathcal{A}^{\mathcal{D}}$

s.t. $\alpha\times\beta\times\gamma''' = \theta = \alpha\times\gamma' = \beta\times\gamma''$ which by the completeness results means that $T = T(\alpha\times\beta\times\gamma''')$.

The interpretation function I is applied to the canonical almost normal form, and therefore we consider the actions $\alpha\times\gamma'$ and $\alpha\times\beta\times\gamma'''$ to be in $canf$. Because the canonical form is defined inductively it is w.l.o.g. that we look only at the first levels of the actions (i.e., only at the $\times$-actions $\alpha^i_\times$ of the canonical form). For a simple notation we denote the $\times$-actions on the first level of α by $\alpha_1, \alpha_2, \ldots, \alpha_k$; note that there are k actions in total. For the action β we denote the $\times$-actions on the first level by $\beta_1, \beta_2, \ldots, \beta_l$. For the action θ we denote the $\times$-actions by τ_i.

To prove the lemma we use the proof principle *reductio ad absurdum* and suppose that $\alpha\times\beta\times\gamma''' \neq \theta$ is the case. According to the above this supposition is equivalent to saying that the $\times$-actions on the first level of θ are not constructed from the actions on the first level of $\alpha\times\beta$. This may be from several reasons.

First consider that a $\times$-action of $\alpha\times\beta$, say $\alpha_1\times\beta_1$ is not contained in any of the $\times$-actions τ_i on the first level of θ. Consider $\tau^i_{\alpha_1}$ to be those τ_i which contain α_1; and similarly consider $\tau^j_{\beta_1}$ those τ_i which contain β_1. From the supposition we know that β_1 does not appear in any of the $\tau^i_{\alpha_1}$; and similarly α_1 does not appear in any $\tau^j_{\beta_1}$. From the hypothesis $\theta = \beta\times\gamma''$ we know that in all τ it appears one of the β_j $\times$-actions. This means that in each of the $\tau^i_{\alpha_1}$ it appears one of the β_j where $j \neq 1$. Consider w.l.o.g. one of these actions $\tau^1_{\alpha_1} = \alpha_1\times\beta_2\times\gamma$ for some γ which may also be empty. From the same hypothesis $\theta = \beta\times\gamma''$ and knowing that $\alpha_1\times\beta_2\times\gamma$ is a $\times$-action on the first level of θ then it means that $\alpha_1\gamma$ is an action on the first level of γ''. This means that between the actions τ of the first level of θ there exists each of the actions $\alpha_1\times\gamma\times\beta_j$ with $j \neq 2$ (because we already have the index 2). In other words, the action $\alpha_1\times\gamma$ must be combined with any of the actions β_j including β_1.

We thus obtained the contradiction (i.e., there exists an action τ which contains $\alpha_1\times\beta_1$). Therefore, each of the $\alpha_i\times\beta_j$ of $\theta = \alpha\times\beta\times\gamma'''$ are contained in τ_i. In other words we have proven that all the $\times$-actions on the first level of the action $\alpha\times\beta$ are found among the $\times$-actions on the first level of θ. Moreover, the discussion above also proves that $\forall\tau\in\theta,\ \tau = \alpha_i\beta_j\gamma$; which says that there is no $\times$-action on the first level of θ which does not contain an action from the first level of $\alpha\times\beta$.

The only way to still have the (bad) supposition is to say that it is not the case that for all pairs $\alpha_i\beta_j$ there exits a same γ such that $\alpha_i\times\beta_j\times\gamma = \tau$ is a $\times$-action on the first level of θ. To explain it differently, this supposition wants to contradict the second $\times$ operator in the conclusion of the lemma $(\alpha\times\beta)\times\gamma'''$ which by the definition it must be that for each γ an action on the first level of γ''' it must be combined with each action $\alpha_i\times\beta_j$ of $\alpha\times\beta$.

We take an arbitrary pair $\alpha_i\times\beta_j$, say $\alpha_1\times\beta_1$ and w.l.o.g. suppose it has some extra action $\gamma_\times$ which may be also empty. Thus $\alpha_1\times\beta_1\times\gamma_\times$ is an action on the first level of θ. From the hypothesis $\beta\times\gamma'' = \theta$ and knowing that β_1 is combined with the action $\alpha_1\times\gamma_\times$ it implies that all other β_j with $j \neq 1$ must be combined with the same action. Therefore, the following are also actions τ: $\alpha_1\times\beta_2\times\gamma_\times, \ldots, \alpha_1\times\beta_{n''}\times\gamma_\times$. On the other hand, from the hypothesis $\alpha\times\gamma' = \theta$ and knowing that $\alpha_1\times\beta_1\times\gamma_\times$ is a τ action it means that all other α_i actions must be combined with $\beta_1\times\gamma_\times$. Therefore, we also have as τ actions: $\alpha_2\times\beta_1\times\gamma_\times, \ldots, \alpha_k\times\beta_1\times\gamma_\times$.

We continue to apply recursively the same reasoning on the new deduced actions like $\alpha_2 \times \beta_1 \times \gamma$ and we obtain in the end that all the actions $\alpha_k \times \beta_l$ appear among the actions τ on the first level of θ combined with the same action $\gamma_\times$. Thus, the second false supposition is contradicted.

The last way of contradicting the lemma is trivial and it supposes that it is not the case that all the τ actions of θ come from combination by $\times$ with the actions $\alpha_i \times \beta_j$. More clearly this tries to say that there exit other τ actions that do not follow the pattern deduced by the first two reasonings we had before. This cannot be as if there were another action besides $\alpha_i \times \beta_j \times \gamma_\times$, say τ' we have proven by contradicting the first supposition that this must be of the form $\alpha_i \times \beta_j \times \gamma'_\times$ and by the second supposition we again get that there exist all the $\alpha_i \times \beta_j \times \gamma'_\times$ as actions τ on the first level of θ.

The proof of the lemma is finished, as the bad supposition is always contradicted. $\Box$

Lemma A.4. *For any $K^{\mathcal{N}}$ a normative structure and α, β two distinct actions we have that if $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then $T(\alpha \times \beta)\ \mathcal{S}_i\ K^{\mathcal{N}}$.*

Proof: We use Lemma A.3 and mainly the naturalness constraint on obligations from Definition 3.19.

From the statement of the lemma $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ by applying the Lemma A.1 we get that $K_{max}^{T(\alpha),i} = K_{max}^{T(\beta),i}$ (we treat the two cases with strict inclusion at the end). This implies (see Corollary A.2) that the corresponding trees which unfold these maximal substructures are the same; i.e., $TK_{max}^{T(\alpha),i} = TK_{max}^{T(\beta),i} = TK_{max}^{\mathcal{N}}$.

Moreover, from the hypothesis of the lemma we get that $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha)$ and $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\beta)$. Considering the *naturalness* constraint it implies that:

$$\exists \gamma' \text{ s.t. } T(\alpha \times \gamma') = TK_{max}^{T(\alpha),i}$$
$$\exists \gamma'' \text{ s.t. } T(\beta \times \gamma'') = TK_{max}^{T(\beta),i}$$

From these and knowing that the maximal simulating structures are the same we get that $T(\alpha \times \gamma') = T(\beta \times \gamma'') = TK_{max}$. By applying the Lemma A.3 we get that $TK_{max} = T(\alpha \times \beta \times \gamma''')$.

Following the Definition 3.3 of the simulation relation $\mathcal{S}_i$, in order to prove the conclusion $T(\alpha \times \beta)\ \mathcal{S}_i\ K^{\mathcal{N}}$ we need to prove that:

(1) $\forall r \xrightarrow{\gamma} t' \in T(\alpha \times \beta), \exists i \xrightarrow{\gamma'} k' \in K^{\mathcal{N}}$ s.t. $\gamma \subseteq \gamma'$ and $t'\ \mathcal{S}\ k'$,

(2) $\forall i \xrightarrow{\gamma'} k' \in K^{\mathcal{N}}$ with $\gamma \subseteq \gamma'$ then $t'\ \mathcal{S}\ k'$.

Using the results of the previous lemmas the proofs of (1) and (2) become simple. As $T(\alpha \times \beta \times \gamma''') = TK_{max}^{\mathcal{N}}$ which is the tree unfolding of the substructure $K_{max}^{\mathcal{N}} = K_{max}^{T(\alpha),i} = K_{max}^{T(\beta),i}$ of $K^{\mathcal{N}}$, then it is simple to see that for any edge $r \xrightarrow{\gamma} t' \in T(\alpha \times \beta)$ there is a transition $i \xrightarrow{\gamma'} k' \in TK_{max}^{\mathcal{N}}$ which clearly $\gamma \subseteq \gamma'$ depending on γ'''. Therefore, $i \xrightarrow{\gamma'} k' \in K_{max}^{\mathcal{N}}$ and thus $i \xrightarrow{\gamma'} k' \in K^{\mathcal{N}}$. The fact that $t'\ \mathcal{S}\ k'$ is true is obvious by applying a similar recursive reasoning and descending one level in the tree. Note that the recursive reasoning stops when the tree node t' has no more children (i.e., no more edges $t' \xrightarrow{\gamma} t''$ exist in $T(\alpha\times\beta)$); and this is always the case as the tree is finite.

For proving (2) we use a similar recursive reasoning as before. From the condition $\gamma \subseteq \gamma'$ it implies that $\gamma' = \gamma\times\gamma''$. Because γ' is a label of a transition in $T(\alpha\times\beta\times\gamma''')$ then $\gamma' = \alpha\times\beta\times\gamma'''_{\times}\times\gamma''$ which because it contains α it enters under the application of the hypothesis $T(\alpha)\ \mathcal{S}_i\ K^{\mathcal{N}}$ (and similarly because it contains β we can apply $T(\beta)\ \mathcal{S}_i\ K^{\mathcal{N}}$). Applying the hypothesis leads to the fact there there are the edges $r \xrightarrow{\gamma} t'_\alpha \in T(\alpha)$ and $r \xrightarrow{\gamma} t'_\beta \in T(\beta)$ with $t'_\alpha\,\mathcal{S}\,k'$ and $t'_\beta\,\mathcal{S}\,k'$. On the other hand t' comes from the combination of the two t'_α and t'_β and thus a simple recursive reasoning gives $t'\,\mathcal{S}\,k'$. The recursive reasoning stops again when the node t' has no more children.

Note that if we consider inclusion among the maximal simulating structures (instead of the equality as we did) then the discussion above does not change. The $TK^{T(\alpha\times\beta),i}_{max}$ is the same as the interpretation $T(\alpha\times\beta\times\gamma''')$. □

In what follows we present two corollaries of Lemmas A.1 and A.4: the first shows what is the maximal simulating structure with respect to $T(\alpha\times\beta)$; and the second states that the obligation of $\alpha\times\beta$ respects the naturalness constraint. Corollary A.5 is used in the proofs of both Lemma A.7 and Lemma A.8.

Corollary A.5. *For any $K^{\mathcal{N}}$ a normative structure and α, β two distinct actions we have that if $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then either*
$K^{T(\alpha),i}_{max} = K^{T(\beta),i}_{max} = K^{T(\alpha\times\beta),i}_{max}$ *or*
$K^{T(\alpha),i}_{max} \subset K^{T(\beta),i}_{max} = K^{T(\alpha\times\beta),i}_{max}$ *or*
$K^{T(\beta),i}_{max} \subset K^{T(\alpha),i}_{max} = K^{T(\alpha\times\beta),i}_{max}$.

Corollary A.6. *If $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then $O(\alpha\times\beta)$ is a natural obligation.*

Lemma A.7. *If $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ then*
$\forall t \xrightarrow{\gamma} t' \in T(\alpha\times\beta)$ *and* $\forall s \xrightarrow{\gamma'} s' \in K^{\mathcal{N}}$ *s.t.* $t\,\mathcal{S}\,s \wedge \gamma\subseteq\gamma'$ *is the case that*
$\forall a \in \mathcal{A}_B$ *if* $a\in\gamma$ *then* $\circ_a \in \varrho(s')$.

Proof: It is simple to see, by looking at Definition 3.4, that all transitions $s \xrightarrow{\gamma'} s'$ mentioned in the lemma make up exactly the maximal simulating structure $K^{T(\alpha\times\beta),i}_{max}$. By Corollary A.5 this is the same as the maximal simulating structures for $T(\alpha)$ and $T(\beta)$.

To finish the proof we take one arbitrary edge $t \xrightarrow{\alpha_\times\times\beta_\times} t' \in T(\alpha\times\beta)$ and one arbitrary transition $s \xrightarrow{\gamma} s' \in K^{T(\alpha\times\beta),i}_{max}$ s.t. $t\,\mathcal{S}\,s$ and $\gamma = \alpha_\times\times\beta_\times\times\gamma'$ where γ' may also be $\mathbf{1}$. These satisfy the conditions in the lemma. The edge $t \xrightarrow{\alpha_\times\times\beta_\times} t'$ comes from the combination of two edges $t \xrightarrow{\alpha_\times} t' \in T(\alpha)$ and $t \xrightarrow{\beta_\times} t' \in T(\beta)$. On the other hand we have for the transition $s \xrightarrow{\gamma} s'$ that both $\alpha_\times \subseteq \gamma$ and $\beta_\times \subseteq \gamma$ hold. This means that we can apply the hypothesis of the lemma (i.e., apply the definition for $O_{\mathcal{C}}$ to both

$O_{\mathcal{C}}(\alpha)$ and $O_{\mathcal{C}}(\beta)$) to get that $\circ_a \in \varrho(s'), \forall a \in \alpha_\times$ and $\circ_a \in \varrho(s'), \forall a \in \beta_\times$ (because the definition says that for all transitions this happens). This implies the result of the lemma, i.e., $\circ_a \in \varrho(s'), \forall a \in \alpha_\times \times \beta_\times$. □

Lemma A.8. *If* $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ *then*
$\forall s \xrightarrow{\gamma'} s' \in K_{rem}^{T(\alpha\times\beta),i}$ *then* $\forall a \in \mathcal{A}_B$ *if* $a \in \gamma'$ *then* $\circ_a \notin \varrho(s')$.

Proof: Following from Corollary A.5 is that $K_{rem}^{T(\alpha\times\beta),i} = K_{rem}^{T(\alpha),i} = K_{rem}^{T(\beta),i}$ (the cases for inclusion are treated at the end). From the hypothesis $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha)$ we have that $\forall s \xrightarrow{\gamma'} s' \in K_{rem}^{T(\alpha),i}$ then $\forall a \in \mathcal{A}_B$ if $a \in \gamma'$ then $\circ_a \notin \varrho(s')$ which makes our proof goal also true by replacing $K_{rem}^{T(\alpha),i}$ with its equal $K_{rem}^{T(\alpha\times\beta),i}$.

In the case when $K_{max}^{T(\alpha),i} \subset K_{max}^{T(\beta),i} = K_{max}^{T(\alpha\times\beta),i}$ then we work as before but consider the structure for β instead. □

Lemma A.9. *If* $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha) \wedge O_{\mathcal{C}}(\beta)$ *then*
$K^{\mathcal{N}}, s \models \mathcal{C} \quad \forall s \in N$ *with* $t\,\mathcal{S}^s\, s \wedge t \in$ *leafs*$(T(\overline{\alpha\times\beta}))$.

Proof: The conclusion of the lemma should be read as: the formula $\mathcal{C}$ holds in all those states $s \in K^{\mathcal{N}}$ which can be reached by "following" the tree interpretation of the action complement $\overline{\alpha\times\beta}$ to the leafs. By "to follow" we mean that the normative structure simulates *strictly* the tree $T(\overline{\alpha\times\beta})$. The simulation must be strict so that we follow *exactly* the tree.

Recall the Definition 2.7 of the action complement. The complement of a compound action $\overline{\alpha}$ works on each level of the complemented action α. For the proof of this lemma it is enough to look at the behavior for only the first level, and for the rest we apply a similar recursive reasoning. Moreover, note that we need to look only at the leafs of the trees (i.e., at the states from the end of the final paths of the tree interpretation of the complemented action). Thus, the first level in the complement contains the choice $+_{\gamma\in\overline{R}}\gamma$ (defining the full branches; we look at the other full branches when we reason recursively at lower levels of the tree).

Thus, we need to prove that $\forall t \xrightarrow{\gamma} t' \in T(+_{\gamma\in\overline{R}}\gamma)$ with $\gamma \in \mathcal{A}_B^\times$ a $\times$-action s.t. $\forall \alpha_\times^i \times \beta_\times^j$ a $\times$-action on the first level of the tree of the complemented action $\alpha\times\beta$ we have that $\alpha_\times^i \times \beta_\times^j \not\subseteq \gamma$ then it is the case that if $\exists s \xrightarrow{\gamma} s' \in K^{\mathcal{N}}$ then $K^{\mathcal{N}}, s' \models \mathcal{C}$. Take an arbitrary transition $t \xrightarrow{\gamma} t'$ for which the above hold and for which $\exists s \xrightarrow{\gamma} s' \in K^{\mathcal{N}}$ and we show that $K^{\mathcal{N}}, s' \models \mathcal{C}$.

From the condition $\forall \alpha_\times^i \times \beta_\times^j, \alpha_\times^i \times \beta_\times^j \not\subseteq \gamma$ we can conclude that either $\forall \alpha_\times^i, \alpha_\times^i \not\subseteq \gamma$ or $\forall \beta_\times^j, \beta_\times^j \not\subseteq \gamma$. This is done by using the proof principle *reductio ad absurdum* and we suppose that neither of the $\forall \alpha_\times^i, \alpha_\times^i \not\subseteq \gamma$ nor $\forall \beta_\times^j, \beta_\times^j \not\subseteq \gamma$ hold. This means that $\exists i', j'$ s.t. $\alpha_\times^{i'} \subseteq \gamma \wedge \beta_\times^{j'} \subseteq \gamma$ which implies that $\alpha_\times^{i'} \times \beta_\times^{j'} \subseteq \gamma$. By looking again at the definition of the $\times$ operation we see that $\alpha_\times^{i'} \times \beta_\times^{j'}$ must be an action among the $\alpha_\times^i \times \beta_\times^j$. Therefore, the conclusion that we have just drawn before enters into contradiction with the initial condition $\forall \alpha_\times^i \times \beta_\times^j, \alpha_\times^i \times \beta_\times^j \not\subseteq \gamma$.

By using one of the hypothesis of the lemma, say $K^{\mathcal{N}}, i \models O_{\mathcal{C}}(\alpha)$ we conclude from the definition of the semantics of $O_{\mathcal{C}}$ that the transition that we work with $t \xrightarrow{\gamma} t'$ respects the fact that $\forall \alpha^i_\times, \alpha^i_\times \not\subseteq \gamma$ and thus in the end state of the transition $s \xrightarrow{\gamma} s' \in K^{\mathcal{N}}$ we have $K^{\mathcal{N}}, s' \models \mathcal{C}$. This is the conclusion of the lemma. $\square$