ECAI 2016
G.A. Kaminka et al. (Eds.)

doi:10.3233/978-1-61499-672-9-886

A Dynamic Logic of Norm Change

Max Knobbout and **Mehdi Dastani** and **John-Jules Meyer** [1]

Abstract. Norms are effective and flexible means to control and regulate the behaviour of autonomous systems. Adding norms to a system changes its specification which may in turn ensure desirable system properties. As of yet, there is no generally agreed formal methodology to represent and reason about the dynamics of norms and their impacts on system specifications. In this paper, we introduce various types of norms, such as state-based or action-based norms, and gradually develop a dynamic modal logic to characterize the dynamics of such norms in a formal way. The logic can be used to prove various properties of norm dynamics and their impacts on system specification. Moreover, we show that this logic is sound and complete.

1 Introduction

Norms are widely proposed as an effective and flexible means to control and regulate the behaviour of autonomous systems. Generally, norms specify the standards of behaviours such as which actions or states should be achieved or avoided. Adding norms to a system changes its specification which may in turn incentivize/inhibit specific behaviours and thereby ensure some desirable system level properties. For example, consider the norm "individuals entering a train station should have valid tickets" being introduced in a train station. The addition of this norm incentivize having a valid ticket before entering the train station and ensures that the train station is not getting unnecessarily crowded. We assume that the addition of norms incentivizes/inhibits behaviours by various enforcement means such as regimentation (e.g. by placing ports at the entrance gates of the train station) or by means of sanctions (e.g. by random inspection of individuals at the train station and issuing fines for those who has no valid ticket). In this paper, we ignore the issue of norm enforcement and focus on how norms update system specifications, i.e., which system states or actions are considered as good/bad after the system is updated with norms. A system to which a set of norms is added, i.e., a system that is governed by a set of norms, is referred to as a normative system [15, 14, 1].

A lot of research has focused on deciding (or proving) correctness of a normative system. A normative system, i.e., a system with a set of norms, is correct if the objectives of the system designer are satisfied after the norms have been added to the system [15, 2, 12]. As of yet, there is no generally agreed formal methodology to represent and reason about the norm change in normative systems. Such a methodology allows us to formally investigate the dynamics of various types of norms, such as state-based or action-based norms, in normative systems and their impact on the system specification. For example, the methodology enables us to reason about the introduction of the above-mentioned norm in a train station before and after an individual has entered the train station, either with or without a valid ticket. It also enables us to reason about the impact of different order of norm change on system specifications, and the interaction between norm change and agents' behaviours.

One may identify two possible methodological approaches when dealing with norm change. On the one hand we can identify the syntactic approach [7], where norm change is considered as an operation on the underlying "code" that constitutes the system. On the other hand we can identify the semantic approach [4, 13] which aims to look at norm change as an update of the model. This work falls in the second category, but is novel because (1) instead of providing just a semantic analysis we provide a new dynamic logic to represent and reason about norm change, and (2) we provide an accompanying (sound and complete) proof system for the logic.

The view we adopt is that normative systems can be modelled by pointed labelled transition systems, which show which facts become true under execution of which actions. Moreover, we assume that updating a system with a norm modifies the system specification and thereby its behaviour. The specific problem we address in this paper is how to represent and reason about these norm updates. We introduce new types of norms that are expressive enough to model existing norm types. We propose a new dynamic norm logic with norm update operations and an accompanying proof system to reason about norm updates in normative systems. Inspired by dynamic logic [11, 16], the effect of a norm update operation is an update of the normative system. The kind of updates (norms applied to normative systems) and its effects (normative systems that are aligned with the norm) are completely novel. The contribution of this work is significant because it paves the way for the development of formal tools that can be used to prove correctness of a normative system. From a practical point of view such tools are essential to investigate the interaction between norm change and the system behaviour, in particular, whether or when the addition of some norms satisfy some desired system properties.

In this paper, we first introduce a formal framework to model normative systems and various types of norms. We then explore norms of the 'to-be' variant, which may forbid (or permit) certain states to occur. We devise a dynamic logic with update operation for 'to-be' norms. We then move to norms of the 'to-do' variant, which may forbid (or permit) certain actions to occur. For this norm type, we devise a dynamic logic with update operation for 'to-do' norms. For the proposed dynamic logics we provide accompanying proof systems.

2 Framework

This section defines the models we use for normative systems, and presents the syntax and semantics of the logic we use to express properties of these systems. We consider a normative system as a labelled transition system that gives us for each possible execution of actions the facts which become true. This is a standard way to model the be-

[1] Utrecht University, The Netherlands, email: {M.Knobbout,M.M.Dastani,J.J.C.Meyer}@uu.nl

haviour of a system, except that we additionally assume the existence of a set of violation atoms in order to model what is forbidden and permitted. This idea comes from Anderson's reduction [5], where bad state of affairs can be labelled by a violation atom. A state with a violation atom assigned to it is then interpreted as a forbidden state. We note that a violation atom can represent a fine, but it can also simply mean that something bad has happened. In this paper, we do not wish to model whether an action is produced by a single agent or a group of agents (e.g. Concurrent Game Structures). Because of this simplification, we do not have to represent the agents explicitly, and can just assume the existence of some action alphabet. In the remainder of this paper we use the term normative system to refer to such a structure.

Definition 1 *A normative system N is a tuple $(Q, Act, \rightarrow, \Pi, V, \mu)$ such that:*

- *Q is a non-empty finite set of states from the system.*
- *Act is a finite set of (domain) actions.*
- *$\rightarrow \subseteq Q \times Act \times Q$ is a relation between states with actions, such that for all $q \in Q$ and $\alpha \in Act$ there exists exactly one $q' \in Q$ such that $(q, \alpha, q') \in \rightarrow$. Whenever $(q, \alpha, q') \in \rightarrow$, we write $q(\alpha)$ to denote q'*
- *Π is a finite set of atomic propositions.*
- *$V \subseteq \Pi$ is a finite set of atomic violation propositions.*
- *μ is a valuation function mapping a state $q \in Q$ to an element from $\mathcal{P}(\Pi)$.*

A pointed normative system is a pair (N, q) such that N is a normative system, and $q \in Q$ a state from N. Given such a structure, we say that a state q is forbidden whenever there exists a violation $v \in V$ such that $v \in \mu(q)$. Similarly, we say that an action α is forbidden in state q whenever there exists a violation $v \in V$ such that $v \in \mu(q(\alpha))$. Note that this model assumes that actions are deterministic (e.g. each action leads to a unique next state) and are always enabled. To model that an action α has no effect on a state q, we can simply model it by assuming that $q(\alpha) = q$, i.e. action α leads to the same state.

The language of propositional logic with action modality, written in this paper as $\mathcal{L}_0$, consists of formulas φ built by the following grammar, where $p \in \Pi$ and $\alpha \in Act$:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid Do(\alpha)\varphi$$

Along a pointed normative system (N, q), we can evaluate formulas of $\mathcal{L}_0$ in the following way:

- $N, q \models p$ iff $p \in \mu(q)$
- $N, q \models \neg\varphi$ iff $N, q \not\models \varphi$
- $N, q \models \varphi_1 \vee \varphi_2$ iff $N, q \models \varphi_1$ or $N, q \models \varphi_2$.
- $N, q \models Do(\alpha)\varphi$ iff $N, q(\alpha) \models \varphi$

Given a pointed normative system (N, q), we say that a sequence of actions $\alpha_1 \dots \alpha_n$ brings about φ if and only if $N, q \models Do(\alpha_1) \dots Do(\alpha_n)\varphi$. As is standard, we say that $N \models \varphi$ holds whenever for all $q \in Q$ it holds that $N, q \models \varphi$, and that $\models \varphi$ holds (alternatively, "φ is valid") whenever for all normative systems N we have $N \models \varphi$. We have the following result from modal logic [9].

Theorem 1 (Sound and complete axiomatization of $\mathcal{L}_0$) *The logic $\mathcal{L}_0$ is soundly and completely axiomatized by modal system* **K** *(i.e. the system consisting of all propositional tautologies together with the necessitation rule and the distributive axiom) with the following 'Function' axiom scheme:*

$$Do(\alpha)\neg\varphi \leftrightarrow \neg Do(\alpha)\varphi$$

We refer to system **K** *with this additional axiom scheme as system* **L0***, and write $\vdash_{\mathcal{L}_0} \varphi$ whenever φ is provable in* **L0***. For all $\varphi \in \mathcal{L}_0$ we thus have:*

$$\models \varphi \quad \Leftrightarrow \quad \vdash_{\mathcal{L}_0} \varphi$$

This axiom represents that the arrows of a normative system are functional, i.e. a state and an action completely and uniquely define the next state. We will now introduce an example of a normative system. This example concerns a train station where individuals can enter and leave. Throughout this paper we use this example as a running example to demonstrate how various norms can alter the behaviour of the system.

2.1 Running Example

Consider the normative system N_{station} in Figure 1. This simple

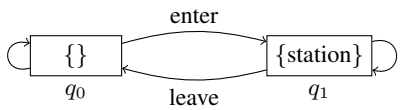

Figure 1. System N_{station}.

system models a train station where an individual/traveller can enter and leave. In particular, the system can either be in state q_0 or state q_1, and depending on the performed action by the individual can switch between these states. Here 'station' denotes the fact that the individual is in the station. The reflexive arrows denote that the performance of all the remaining actions that can occur will result in the same state (e.g. action 'enter' in state q_1 will result in q_1). An example of a formula that holds in this system is:

$$N_{\text{station}}, q_1 \models Do(\text{leave})\neg\text{station}$$

Observe that there are no violations in this system yet. Later on we will add norms to this system and see how the system can show more interesting and complex behaviour.

2.2 Norm Types

Before we formally introduce the language of norms and norm update, it is worthwhile to reflect on the kinds of norms we want to express and model. Broadly speaking, we consider the following two classes:

1. **State-based norms** A state-based norm refers to certain states that should be achieved or avoided. These kinds of norms are of the 'to-be' variant. To extend the expressiveness of these kinds of norms even further, we also optionally allow the addition of a repair action. For example, the station may add a norm which states that being at the station is forbidden until a valid subscription is bought. These state-based norms are thus conditional on a repair action. This class of norms is, to our knowledge, new and allows for expressive norms that are required to model practical scenarios.
2. **Action-based norms** An action-based norm refers to certain actions that should be performed or avoided. These kinds of norms are of the 'to-do' variant. Again, we optionally allow the performance of a repair actions, which allows for expressive norms.

More elaborate norms pertaining to complex behaviours of the system can be acquired by combining norms in various ways. It is important to note that the addition of these norms are not 'physical' actions of the system. They come from outside the system (i.e. a designer) and are not triggered by the actions of the system.

3 Language for Norms and Norm Updates

In this section we begin our first step into developing our language for norms and norm updates. The kinds of norms we consider in this section are of the 'to-be' variant.

3.1 Language and Update

Given a normative system, we construct the norm language $\mathcal{N}_1$ in the following way, where $\varphi \in \mathcal{L}_0$, $v \in V$ and $Act_R \subseteq Act$:

$$\mathsf{n} ::= (\varphi, +v, Act_R) | (\varphi, -v, Act_R)$$

Whenever we have a norm $(\varphi, \pm v, Act_R)$ (where $\pm v$ can either be $+v$ or $-v$), we refer to φ as the norm condition, $\pm v$ as the norm effect and Act_R as the set of actions from the system that will count as repair actions. These constructs relate to the kinds of norms we described earlier. Adding a norm $(\varphi, +v, Act_R)$ implies that for every φ-state the violation v will hold, until a repair action from the set Act_R occurs, at which point the behaviour of the system will revert back to what it was before the norm was added to the system. Whenever such a repair action occurs, we say that the norm effect is repaired. Note that we use the notion of repair in a rather liberal way, since a repair action may very well occur before a φ-state is ever encountered. In this case, we still say that a norm effect is repaired, even though it was never the case that we were in a state in which the norm effect $\pm v$ was realized. We note that the set Act_R can be empty, which implies that the norm is permanent. For example, the norm $(\varphi, +v, \emptyset)$ states that all φ-states are now permanently forbidden. In this way, we acquire the 'classical' interpretation of a state-based norm. The norm $(\varphi, -v, Act_R)$ behaves in a similar fashion, except in this case the proposition v will stop to hold until this norm effect is repaired. Both norms can be used to update pointed normative systems. That is, they transform a pointed normative system to a new pointed normative system. In this paper we want to give clear semantics to how a pointed normative system should behave when a norm from our language is added to the system. This is why we introduce the notion of *norm-aligned*. We say that an updated system is norm-aligned if the system implements the new restrictions of the added norm.

Definition 2 (Norm-Aligned ($\mathcal{N}_1$)) *Let (N, q) be a pointed normative system, $\mathsf{n} = (\varphi, +v, Act_R) \in \mathcal{N}_1$ be a norm, and $(N, q)'$ be a possible update of (N, q) with n. We say that $(N, q)'$ is* norm-aligned *with n if for every proposition $p \in \Pi$ and every (possibly empty) sequence of actions $\alpha_1 \dots \alpha_n \in Act^*$, we have that $(N, q)' \models Do(\alpha_1) \dots Do(\alpha_n)p$ if and only if*

1. $N, q \models Do(\alpha_1) \dots Do(\alpha_n)p$, ***or***
2. $p = v$, $N, q \models Do(\alpha_1) \dots Do(\alpha_n)\varphi$, *and,* $\alpha_1, \dots, \alpha_n \notin Act_R$.

In words, a system updated with a norm $(\varphi, +v, Act_R)$ is norm-aligned with the norm if (1) all propositions that were true before remain true, plus if (2) the norm condition φ is true and no repair action from Act_R has been performed, then v should be true as well. This notion captures both a property of success and a property of minimal change. On the one hand, it states that the norm effect should hold under the right conditions (property of success; the norm condition is true and no repair action has been performed yet), and on the other hand it states that everything else should remain unchanged (property of minimal change). We can in a similar manner define the notion of norm-aligned for an update of the form $(\varphi, -v, Act_R)$, which should reflect that such an update removes v. This leads us to the following postulate.

Postulate 1 (Norm-Aligned ($\mathcal{N}_1$)) *Any normative system updated with a norm $\mathsf{n} \in \mathcal{N}_1$ should be norm-aligned.*

This is inspired by the kind of postulates we can find in the AGM framework, which state how updates should behave [3]. A natural question that arises is how we can define updates that are norm-aligned. That is, given a pointed system (N, q) and norm $\mathsf{n} \in \mathcal{N}_1$, how can we define $(N, q)'$ such that it is norm-aligned? The next section aims to answer this question.

3.1.1 Defining Norm-Aligned Updates

We will now show how we can update a pointed normative system to a new pointed normative system which is norm-aligned. Given a pointed normative system (N, q) and a norm n, we write $(N, q)[\mathsf{n}]$ to denote the updated system, and we define this update as follows.

Definition 3 *Given $N = (Q, Act, \to, \Pi, V, \mu)$, $q \in Q$ and $\mathsf{n} = (\varphi, +v, Act_R) \in \mathcal{N}_1$, we let $(N, q)[\mathsf{n}] = (N[\mathsf{n}], q[\mathsf{n}])$ such that:*

- *System $N[\mathsf{n}] = (Q', Act, \to', \Pi, V, \mu')$, where:*
  - $Q' = \{q^r, q^a \mid q \in Q\}$
  - $\to' = \{(q_i^a, \alpha, q_j^a) \mid q_i(\alpha) = q_j \text{ and } \alpha \notin Act_R\} \bigcup \{(q_i^a, \alpha, q_j^r) \mid q_i(\alpha) = q_j \text{ and } \alpha \in Act_R\} \bigcup \{(q_i^r, \alpha, q_j^r) \mid q_i(\alpha) = q_j\}$
  - *For every state $q \in Q$:*
    $\mu'(q^r) = \mu(q)$ *and*
    $\mu'(q^a) = \begin{cases} \mu(q) \cup \{v\} & \text{if } N, q \models \varphi \\ \mu(q) & \text{otherwise} \end{cases}$
- *State $q[\mathsf{n}] = q^a$*

That is, for every state $q \in Q$ we create two copies in the updated system; one in which the norm effect is active (q^a) and one in which it is repaired (q^r). Thus, for each update the states of the system are duplicated. The transitions between active and repaired states are analogous to the transitions of the original system, except whenever we are in an active state and a repair action occurs, in which case we go to a repaired state. Note that we have written q_i and q_j in this definition to simply denote that these states might possibly be different; we attach no further meaning to this indexing. Alternatively, for an update with $-v$, the updated valuation function μ' becomes:

$$\mu'(q^r) = \mu(q) \text{ and } \mu'(q^a) = \begin{cases} \mu(q) \backslash \{v\} & \text{if } N, q \models \varphi \\ \mu(q) & \text{otherwise} \end{cases}$$

The definition of this update reflects the fact that if no repair action has been performed yet (meaning we are in an active state) and φ is true at this state, then v should also be true for $+v$ and false for $-v$. This leads us to the following result.

Proposition 1 *Given an arbitrary N, $q \in Q$ and $\mathsf{n} \in \mathcal{N}_1$, the pointed system $(N, q)[\mathsf{n}]$ is norm-aligned with n.*

With our running example, we can visualize how this update changes the behaviour of a normative system.

3.1.2 Running Example

We will return to the station system found in Figure 1. Assume that the station wants to impose a policy such that each traveller needs to have a valid travel subscription to be in the station, otherwise he is in violation. Formally, the system is updated with the norm $\mathsf{n}_0 = (\text{station}, +v, \{\text{buy\_sub}\})$, where buy_sub reflects the action of buying a subscription (which was not explicitly drawn in Figure 1): to be in a 'station'-state causes violation v, unless this effect is repaired by buying a subscription. In Figure 2 we see how we update the normative system N_{station} to system $N_{\text{station}}[\mathsf{n}_0]$, and states q_0 and q_1 to q_0^a and q_1^a respectively. It is important to note that the update

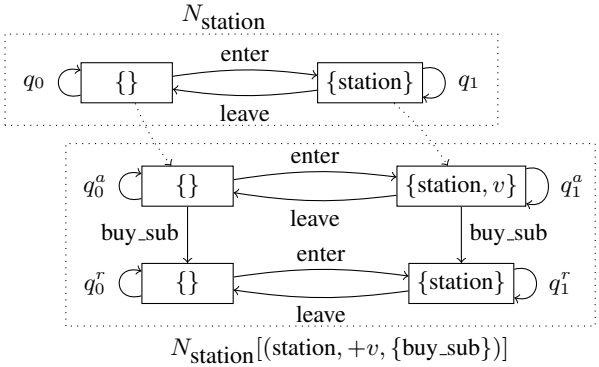

Figure 2. Above system N_{station}, and below the updated system. The dotted transitions denote the state-updates.

can affect the current state; if we would be in state q_1 in the original system, it could be that an update causes us to be in a violation-state, particularly we have:

$$(N_0, q_1)[\mathsf{n}_0] \models v$$

If the 'enter' action is performed before the 'buy_sub' action in the updated system, we would also be in a violation-state. We thus have that:

$$(N_{\text{station}}, q_0)[\mathsf{n}_0] \models Do(\text{enter})v$$

Otherwise this would not be the case. We have:

$$(N_{\text{station}}, q_0)[\mathsf{n}_0] \models Do(\text{buy\_sub})Do(\text{enter})\neg v$$

Thus, we see that this relatively simple norm already makes the behaviour of the system more interesting and complex. In the next section we will extend our language to reason about these dynamic updates.

3.2 Logic and Axiomatization

We can now extend modal language $\mathcal{L}_0$ to dynamic modal language $\mathcal{L}_1$ by adding norm update operation $[\mathsf{n}]$ to $\mathcal{L}_0$, where $p \in \Pi$, $\alpha \in Act$, and $\mathsf{n} \in \mathcal{N}_1$:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid Do(\alpha)\varphi \mid [\mathsf{n}]\varphi$$

Formulas of $\mathcal{L}_1$ are evaluated along pointed normative systems. This is done by adding the following rule to the satisfaction relation of $\mathcal{L}_0$:

$$N, q \models [\mathsf{n}]\varphi \quad \text{iff} \quad (N, q)[\mathsf{n}] \models \varphi$$

We reiterate that although this follows the definition of an update we can find in dynamic logic, the difference here is that we update with a norm consisting of a condition, effect and repair actions. A formula $[\mathsf{n}]\varphi$ should be read as: "after adding norm n to the normative system it is the case that φ holds".

3.2.1 Examples

Before we go to an axiomatization of this logic, let us look at some specific (non-)validities of this dynamic logic. First and foremost, if φ holds at a specific state, and we update with $(\varphi, +v, Act_R)$, we expect that the violation v holds. This is reflected by the following validity:

$$\models \varphi \rightarrow [(\varphi, +v, Act_R)]v$$

We expect that the other way around is not necessarily the case (the right side implying the left side), since it might be that v was already the case in a state. Again, this is reflected by the following non-validity:

$$\not\models ([(\varphi, +v, Act_R)]v) \rightarrow \varphi$$

However, if we know that $\neg v$ is the case and we know that after the norm update operation v holds, it can only be the case that φ holds. Thus:

$$\models \neg v \wedge ([(\varphi, +v, Act_R)]v) \rightarrow \varphi$$

For any action $\alpha \in Act$ such that $\alpha \in Act_R$, we have the following expected validity:

$$\models ([(\varphi, \pm v, Act_R)](Do(\alpha)\psi)) \leftrightarrow Do(\alpha)\psi$$

That is to say, whenever a norm effect is repaired, the truth of a formula ψ depends merely on whether it was true before the update; i.e. nothing changes. This follows from the fact that these updates are norm-aligned. However, when the norm effect is not repaired, we can infer the truth of ψ by *first* performing the action α and *then* updating the system. Or in short, it does not matter at what moment the update is performed. Thus, for any action $\alpha \in Act$ such that $\alpha \notin Act_R$ we have:

$$\models [(\varphi, \pm v, Act_R)](Do(\alpha)\psi) \leftrightarrow (Do(\alpha)[(\varphi, \pm v, Act_R)]\psi)$$

We also have the important property that order matters. First updating with norm n_0 and then with n_1 may have different results than first updating with n_1 and then with n_0. For example, we have:

$$\models [(\top, +v, \emptyset)]([(v, +v', \emptyset)]v') \text{ , and,}$$
$$\not\models [(v, +v', \emptyset)]([(\top, +v, \emptyset)]v')$$

Lastly, we can also have updates without any effects, since for example the update $(\bot, +v, Act_R)$ leaves the truth of any formula unchanged:

$$\models ([(\bot, +v, Act_R)]\psi) \leftrightarrow \psi$$

In the next section we will provide an axiomatization of our logic.

3.2.2 Axiomatization

We will now show how these properties lead to a non-trivial axiomatization of our logic. We have the following result.

Theorem 2 *The logic $\mathcal{L}_1$ is (soundly and completely) axiomatized by system* **L1***, which consists of adding the following reduction axiom schemes to system* **L0** *together with the rule of replacement of equivalent formulas, where* n *is* $(\varphi, \pm v, Act_R)$*:*

1. *(a)* $([(\varphi, +v, Act_R)]v) \leftrightarrow (\varphi \vee v)$
   (b) $([(\varphi, -v, Act_R)]v) \leftrightarrow (\neg\varphi \wedge v)$
2. $([(\varphi, \pm v, Act_R)]p) \leftrightarrow p$ *(if* $v \neq p$*)*
3. $([\mathsf{n}]\neg\psi) \leftrightarrow (\neg[\mathsf{n}]\psi)$
4. $([\mathsf{n}](\psi_1 \vee \psi_2)) \leftrightarrow (([\mathsf{n}]\psi_1) \vee ([\mathsf{n}]\psi_2))$
5. $([\mathsf{n}]Do(\alpha)\psi) \leftrightarrow (Do(\alpha)\psi)$ *(if* $\alpha \in Act_R$*)*
6. $([\mathsf{n}]Do(\alpha)\psi) \leftrightarrow (Do(\alpha)[\mathsf{n}]\psi)$ *(if* $\alpha \notin Act_R$*)*

Proof. **(Soundness)** Soundness can be proven by proving soundness for every axiom independently. Soundness of each axiom can be shown by following the definitions found in the paper except for axiom 5, which requires the result that for every (N, q) we have $N[\varphi, \pm p, Act_R], q^r \models \psi$ iff $N, q \models \psi$, which can be shown by showing the existence of bi-simulation relation between the two models and states. We show soundness of axiom 6 as example, and omit the other ones due to space constraints:

$$\begin{array}{rl} N, q \models [\mathsf{n}]Do(\alpha)\psi & \Leftrightarrow \\ N[\mathsf{n}], q^r \models Do(\alpha)\psi & \Leftrightarrow \\ N[\mathsf{n}], q^r(\alpha) \models \psi & \Leftrightarrow \\ N[\mathsf{n}], q(\alpha)^r \models \psi \ \text{(if } \alpha \notin Act_T\text{)} & \Leftrightarrow \\ N, q(\alpha) \models [\mathsf{n}]\psi \ \text{(if } \alpha \notin Act_T\text{)} & \Leftrightarrow \\ N, q \models Do(\alpha)[\mathsf{n}]\psi \ \text{(if } \alpha \notin Act_T\text{)} & \end{array}$$

(Completeness) In order to show completeness, we define the following translation function $\tau_1 : \mathcal{L}_1 \to \mathcal{L}_0$:

$$\begin{array}{rcll} \tau_1(p) & = & p & \\ \tau_1([(\varphi, +v, Act_R)]v) & = & \varphi \vee v & \\ \tau_1([(\varphi, -v, Act_R)]v) & = & \neg\varphi \wedge v & \\ \tau_1(\neg\psi) & = & \neg\tau_1(\psi) & \\ \tau_1(\psi_1 \vee \psi_2) & = & \tau_1(\psi_1) \vee \tau_1(\psi_2) & \\ \tau_1(Do(\alpha)\psi) & = & Do(\alpha)\tau_1(\psi) & \\ \tau_1([\mathsf{n}]\neg\psi) & = & \tau_1(\neg[\mathsf{n}]\psi) & \\ \tau_1([\mathsf{n}](\psi_1 \vee \psi_2)) & = & \tau_1(([\mathsf{n}]\psi_1) \vee ([\mathsf{n}]\psi_2)) & \\ \tau_1([\mathsf{n}]Do(\alpha)\psi) & = & \tau_1(Do(\alpha)\psi) & (\text{if } \alpha \in Act_R) \\ \tau_1([\mathsf{n}]Do(\alpha)\psi) & = & \tau_1(Do(\alpha)[\mathsf{n}]\psi) & (\text{if } \alpha \notin Act_R) \\ \tau_1([\mathsf{n}_1]([\mathsf{n}_2]\psi)) & = & \tau_1([\mathsf{n}_1]\tau_1([\mathsf{n}_2]\psi)) & \end{array}$$

The first observation we can make is that this function is well-defined, i.e. for any formula $\varphi \in \mathcal{L}_1$ this function returns a formula $\tau_1(\varphi) \in \mathcal{L}_0$. The reason that this function is well-defined (i.e. always gives an answer) is that we can show under a suitable definition of the complexity of formulas (see [16]) that this translation always reduces the complexity. Using this translation, we can show that for any $\varphi \in \mathcal{L}_1$ we have $\vdash_{\mathrm{L}_1} \varphi \leftrightarrow \tau_1(\varphi)$ (where $\vdash_{\mathrm{L}_1} \varphi$ stands for "φ is provably in system **L1**"), which due to space constraints we omit in this paper. By soundness of $\mathcal{L}_1$, we have from $\vdash_{\mathrm{L}_1} \varphi \leftrightarrow \tau_1(\varphi)$ that $\models \varphi \leftrightarrow \tau_1(\varphi)$. Thus, from our assumption that $\models \varphi$, we have $\models \tau_1(\varphi)$. By the completeness of system **L0** (Theorem 1), we have $\vdash_{\mathrm{L}_0} \tau_1(\varphi)$. This implies that also $\vdash_{\mathrm{L}_1} \tau_1(\varphi)$, since **L1** contains all the rules and axioms of **L0**. Since we have shown that $\vdash_{\mathrm{L}_1} \varphi \leftrightarrow \tau_1(\varphi)$ and since $\vdash_{\mathrm{L}_1} \tau_1(\varphi)$, we can apply modus ponens to acquire $\vdash_{\mathrm{L}_1} \varphi$.

These axioms are inspired by the type of reduction axioms we may find in [16]. Axiom 1a states that if we update every φ-state with $+v$, in order for v to be true it either had to be true before the update, or φ is the case. Axiom 1b states that if we update every φ-state with $-v$, in order for v to be true it has to be both true before the update and the state itself was not updated, i.e. $\neg\varphi$ is the case. Axiom 2 states that any other proposition for which the update does not apply behave invariantly: i.e. the truth condition remains as what it was before the update. Axiom 3 and 4 both state that the dynamic update is a *function*: a normative system N together with an update n uniquely defines an updated system $N[\mathsf{n}]$. Axiom 5 states that when a repair action is performed, the norm is no longer in effect. Finally, axiom 6 states that if a certain action does not repair the norm effect, it does not matter if we perform the update before or after this action in order to determine the state of affairs caused by this action.

We can now show how we can derive the validity of assertions from language $\mathcal{L}_1$. Previously in the paper, we already briefly mentioned without proof that the following assertion is valid:

$$\models [(\top, +v, \emptyset)]([(v, +v', \emptyset)]v')$$

Below, we prove it formally by providing a derivation from system **L1**. We use notation 'RE(x,y)' to denote that we replaced a subformula from line x by using an acquired equivalence from line y, which is short-hand notation for applying the rule of replacement of equivalence formulas (RE) and then applying modus ponens (MP) on the result. Moreover, we use the notation 'MP(x,y)' to denote that we apply modus ponens (MP) with the implication from line x and the precedent from line y. In the proof, we assume that $v \neq v'$:

$([(v, +v', \emptyset)]v') \leftrightarrow (v \vee v')$	Ax.1(a)
$([(\top, +v, \emptyset)](v \vee v')) \leftrightarrow ([(\top, +v, \emptyset)]v \vee [(\top, +v, \emptyset)]v')$	Ax.4
$([(\top, +v, \emptyset)]v) \leftrightarrow (\top \vee v)$	Ax.1(a)
$([(\top, +v, \emptyset)]v') \leftrightarrow v'$	Ax.2
$([(\top, +v, \emptyset)](v \vee v')) \leftrightarrow ((\top \vee v) \vee [(\top, +v, \emptyset)]v')$	RE(2,3)
$([(\top, +v, \emptyset)](v \vee v')) \leftrightarrow ((\top \vee v) \vee v')$	RE(5,4)
$([(\top, +v, \emptyset)]([(v, +v', \emptyset)]v')) \leftrightarrow ((\top \vee v) \vee v')$	RE(6,1)
$((\top \vee v) \vee v')$	Taut.
$[(\top, +v, \emptyset)]([(v, +v', \emptyset)]v')$	MP(7,8)

Thus, we have:

$$\vdash_{\mathrm{L}_1} [(\top, +v, \emptyset)]([(v, +v', \emptyset)]v')$$

And by soundness of **L1** we have:

$$\models [(\top, +v, \emptyset)]([(v, +v', \emptyset)]v')$$

A natural question we may ask is whether logic $\mathcal{L}_1$ with corresponding system **L1** is *decidable*, i.e. whether there exists an effective procedure that tells us whether an arbitrary formula $\varphi \in \mathcal{L}_1$ is valid. We have the following result.

Theorem 3 *The logical language* $\mathcal{L}_1$ *with corresponding system* **L1** *is decidable.*

Proof. We use the translation function $\tau_1 : \mathcal{L}_1 \to \mathcal{L}_0$ from our completeness proof in Theorem 2 to establish the following correspondence:

$$\vdash_{\mathrm{L}_1} \varphi \quad \Leftrightarrow \quad \vdash_{\mathrm{L}_1} \tau_1(\varphi) \quad \Leftrightarrow \quad \vdash_{\mathrm{L}_0} \tau_1(\varphi)$$

Thus, in order to determine whether $\vdash_{\mathrm{L}_1} \varphi$, it is both necessary and sufficient to show that $\vdash_{\mathrm{L}_0} \tau_1(\varphi)$. However, since logic $\mathcal{L}_0$ with corresponding system **L0** is decidable and since τ_1 is well defined and can be effectively computed, decidability immediately transfers to our logic and system.

This concludes our work on the logical language $\mathcal{L}_1$ and **L1** allowing us to reason about normative update of the 'to-be' variant. In the next section we take a look at normative update of the 'to-do' variant, leading to another dynamic logic which, as we will see later, will extend this logic.

4 Extended Norm Language and Update

In the previous section, we developed a basic logical language of normative update. In this section we consider a different kind of normative update, which corresponds to norms of the 'to-do' variant.

4.1 Language and Update

In the remainder of this section, we give a language to construct these kinds of norms, we show how we can update a normative system using these norms and finally we show how we can add these constructs to our logical language. We construct the norm language $\mathcal{N}_2$ in the following way, where $\varphi \in \mathcal{L}_0$, $v \in V$, $Act_T \subseteq Act$ and $Act_R \subseteq Act$:

$$\mathsf{n} ::= (Act_T, \varphi, +v, Act_R) \mid (Act_T, \varphi, -v, Act_R)$$

We call the set Act_T the set of trigger actions. Adding a norm $(Act_T, \varphi, +v, Act_R)$ implies that whenever a trigger action from Act_T occurs, from that point on for every φ-state the violation v will start to hold until a repair action from the set Act_R occurs. An example norm might be that speeding will result in a violation v which can be repaired by paying a fine, which can be modelled by $(\{\texttt{speeding}\}, \top, +v, \{\texttt{pay}\})$. Again, since we want to give a clear semantic interpretation to how a pointed normative system should behave when a norm is added to the system, we extend the notion of *norm-aligned* to updates from this extended language.

Definition 4 (Norm-Aligned ($\mathcal{N}_2$)) *Given a pointed normative system (N, q), a norm $\mathsf{n} = (Act_T, \varphi, +v, Act_R) \in \mathcal{N}_2$, we say that $(N, q)'$ is* norm-aligned *with* n *if for every atomic proposition p and every (possibly empty) sequence of actions $\alpha_1 \dots \alpha_n \in Act^*$, we have that $(N, q)' \models Do(\alpha_1) \dots Do(\alpha_n)p$ if and only if*

1. $N, q \models Do(\alpha_1) \dots Do(\alpha_n)p$, ***or;***
2. $p = v$, $N, q \models Do(\alpha_1) \dots Do(\alpha_n)\varphi$, *and* $\exists i : \alpha_i \in Act_T, \forall j, i < j \leq n : \alpha_j \notin Act_R$.

In words, a system updated with a norm $(Act_T, \varphi, +v, Act_R)$ is norm-aligned the norm if (1) all propositions that were true before remain true, plus (2) if the norm condition φ is true, a trigger action from Act_T has been performed and no repair action from Act_R has been performed yet, then v should be true as well. Again, this notion captures both a property of success and a property of minimal change. On the one hand, it states that the norm effect should hold under the right conditions (property of success; a trigger action has been performed, the norm condition is true and no repair action has been performed yet), and on the other hand it states that everything else should remain unchanged (property of minimal change). Notice that by the above interpretation of norm-aligned, whenever we have a norm $(Act_T, \varphi, \pm v, Act_R)$ and an action α such that $\alpha \in Act_T$ *and* $\alpha \in Act_R$, it is always the case that action α triggers and not repairs the norm effect, even though this action is both a trigger and a repair action. In other words, the trigger actions take priority over the repair actions. Again, more complex norms can be encoded by combining a multitude of these simpler norms. We again have that every update should be norm-aligned, as given by the following postulate.

Postulate 2 (Norm-Aligned ($\mathcal{N}_2$)) *Any normative system updated with a norm $\mathsf{n} \in \mathcal{N}_2$ should be norm-aligned.*

We will now show how we can perform a norm update which results in a norm-aligned normative system. This norm update is very similar to the previous norm update we saw in this paper, the main difference lying in the updated accessibility relation of the model. Formally:

Definition 5 *Given $N = (Q, Act, \to, \Pi, V, \mu)$, $q \in Q$ and $\mathsf{n} = (Act_T, \varphi, +v, Act_R) \in \mathcal{N}_2$, we let $(N, q)[\mathsf{n}] = (N[\mathsf{n}], q[\mathsf{n}])$ such that:*

- *System $N[\mathsf{n}] = (Q', Act, \to', \Pi, V, \mu')$, where:*
  - $Q' = \{q^r, q^a \mid q \in Q\}$
  - $\to' =$
    $\{(q_i^r, \alpha, q_j^r) \mid q_i(\alpha) = q_j \text{ and } \alpha \notin Act_T\} \bigcup$
    $\{(q_i^r, \alpha, q_j^a) \mid q_i(\alpha) = q_j \text{ and } \alpha \in Act_T\} \bigcup$
    $\{(q_i^a, \alpha, q_j^a) \mid q_i(\alpha) = q_j \text{ and } \alpha \notin (Act_R \backslash Act_T)\} \bigcup$
    $\{(q_i^a, \alpha, q_j^r) \mid q_i(\alpha) = q_j \text{ and } \alpha \in (Act_R \backslash Act_T)\}$
  - *For every state $q \in Q$:*
    $\mu'(q^r) = \mu(q)$ *and*
    $\mu'(q^a) = \begin{cases} \mu(q) \cup \{v\} & \text{if } N, q \models \varphi \\ \mu(q) & \text{otherwise} \end{cases}$
- *State $q[\mathsf{n}] = q^r$*

The norm update with $-v$ works analogously, except we remove this atomic proposition from the norm updated valuation function μ' for every active state. Again, for every state $q \in Q$ we create two copies in the updated system; one in which the norm effect is active and one in which it is repaired. There are two important differences in norm updates from $\mathcal{N}_2$ in comparison with norm updates from $\mathcal{N}_1$. They are the following:

1. The updated accessibility relation is different. It is still the case that whenever we are in an active state and a repair action has been performed (which is not also a trigger action), we go to a repaired state. However, whenever we are in a repaired state and a trigger action has been performed, we go to an active state. In other words, with this norm update we are both able to go from active states to repaired states and vice-versa. If no trigger or repair action is performed, we remain in an active (or repaired) state.
2. With the update with norms from $\mathcal{N}_2$ the current state is updated to a repaired state, while with updates with norms from $\mathcal{N}_1$ the current state was updated to an active state.

The following proposition shows that these norm updates again result in a norm-aligned system:

Proposition 2 *Given an arbitrary N, $q \in Q$ and $\mathsf{n} \in \mathcal{N}_2$, the pointed system $(N, q)[\mathsf{n}]$ is norm-aligned with* n.

We return to our running example of the station to visualize how this norm update works.

4.1.1 Running Example

Returning to the example from Figure 1, we now want to update this system with the norm which states that if we do not check-in and enter the station, we are in violation until we leave. We assume that entering through this method is encoded by an action 'unchecked', which encodes the action of not checking in. We can now encode this norm by the norm $\mathsf{n} = (\{\text{unchecked}\}, station, +v, \{\text{leave}\})$, which states that there is a violation v if a traveller is not checked in, and this violation remains as long as the traveller is in the station. By following the rules of the norm update, we see in Figure 3 how we

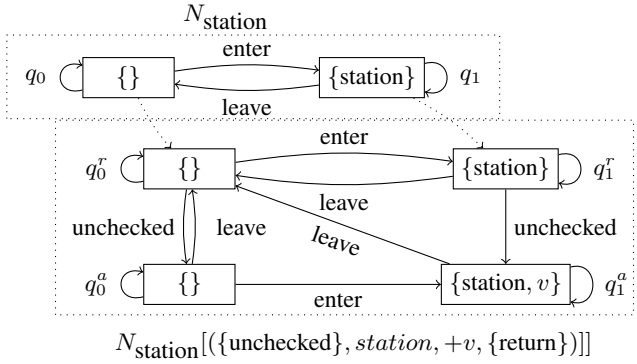

Figure 3. Above system N_{station}, and below the updated system. The dotted transitions denote the state-updates.

go from system N_{station} to $N_{\text{station}}[\mathsf{n}]$, and states q_0 and q_1 to q_0^r and q_1^r respectively. We now have the validity that:

$$(N_{\text{station}}, q_0)[\mathsf{n}] \models Do(\text{unchecked})Do(\text{enter})v$$

Or in words, not checking in and entering brings about a violation. Note that in principle, norms from $\mathcal{N}_1$ and $\mathcal{N}_2$ can be combined in arbitrary ways, even though in these examples we focussed on them separately.

4.2 Logic and Axiomatization

We can now add these norm update operations to our language in the same way we did to $\mathcal{L}_1$ to acquire $\mathcal{L}_2$, which contains formulas of the form $[\mathsf{n}]\varphi$ where $\mathsf{n} \in \mathcal{N}_2$.

4.2.1 Examples

Before we go to an axiomatization of this logic, let us look at some specific (non-)validities of this extended dynamic logic. First, we expect that updating with $(Act_T, \varphi, +v, Act_R)$ does not immediately change the valuation of the current state. This is reflected by the following validity, which holds for any proposition p':

$$\models ([(Act_T, \varphi, +v, Act_R)]p') \leftrightarrow p'$$

In other words, the truth of p' in the current state after an update depends merely on the truth of p' before the update. Moreover, we expect whenever a norm effect is not triggered by a certain action, it does not matter whether this update is performed before or after this action. That is, whenever $\alpha_1, \ldots, \alpha_n \notin Act_T$, we have that:

$$\begin{aligned}\models\ & ([(Act_T, \varphi, +v, Act_R)](Do(\alpha_1)\ldots Do(\alpha_n)\psi)) \leftrightarrow \\ & (Do(\alpha_1)\ldots Do(\alpha_n)[(Act_T, \varphi, +v, Act_R)]\psi)\end{aligned}$$

Of course, a more interesting scenario happens when a trigger action is performed. If we have that $\alpha \in Act_T$, we have the following validity, which elegantly showcases the connection between updates from $\mathcal{N}_1$ and $\mathcal{N}_2$:

$$\begin{aligned}\models\ & ([(Act_T, \varphi, +v, Act_R)]Do(\alpha)\psi) \leftrightarrow \\ & (Do(\alpha)[\varphi, +v, Act_T \cup Act_R]([Act_T, \varphi, +v, Act_R]\psi))\end{aligned}$$

The truth of this validity is not at all apparent. In words, updating a system with $(Act_T, \varphi, +v, Act_R)$ and then performing a trigger action is equivalent to first performing the trigger action, then updating the system with $(\varphi, +v, Act_T \cup Act_R)$ (note that this norm is in $\mathcal{N}_1$, and that the repair actions are $Act_T \cup Act_R$ instead of just Act_R) and finally updating again with $(Act_T, \varphi, +v, Act_R)$. The reason the norm $(Act_T, \varphi, +v, Act_R)$ does not disappear on the right hand side is because it remains in effect, i.e. it can be triggered by an action at a later stage.

4.2.2 Axiomatization

In the previous section, we saw some example validities which we can use in this section to provide an axiomatization of our logic. We have the following result:

Theorem 4 *The logic $\mathcal{L}_2$ is axiomatized by system* **L2***, which contains all the axioms and rules from* **L1***, but with the following additional reduction axiom schemes together with the rule of replacement of equivalent formulas, where $\mathsf{n} = (Act_T, \varphi, \pm v, Act_R)$:*

1. $([\mathsf{n}]p) \leftrightarrow p$
2. $([\mathsf{n}]\neg\psi) \leftrightarrow (\neg[\mathsf{n}]\psi)$
3. $([\mathsf{n}](\psi_1 \vee \psi_2)) \leftrightarrow (([\mathsf{n}]\psi_1) \vee ([\mathsf{n}]\psi_2))$
4. $([\mathsf{n}]Do(\alpha)\psi) \leftrightarrow$
   $(Do(\alpha)[(\varphi, \pm v, Act_R \cup Act_T)]([\mathsf{n}]\psi))$ *(if* $\alpha \in Act_T$*)*
5. $([\mathsf{n}]Do(\alpha)\psi) \leftrightarrow (Do(\alpha)[\mathsf{n}]\psi)$ *(if* $\alpha \notin Act_T$*)*

Proof. **(Soundness)** Soundness can again be proven by individual soundness for each axiom. Special notice should be laid on axiom 4, which contains a norm from $\mathcal{N}_1$ with $Act_R \cup Act_T$ as the repair actions. This is acquired from the result that for an arbitrary pointed normative system (N, q), norm $\mathsf{n} = (Act_T, \varphi, \pm v, Act_R) \in \mathcal{N}_2$ and formula $\psi \in \mathcal{L}_2$, we have $N[\mathsf{n}], q^a \models \psi$ if and only if $(N[(\varphi, \pm v, Act_R \cup Act_T)], q^a)[\mathsf{n}] \models \psi$. This result can be shown by showing the existence of bi-simulation relation between the two models and states. Intuitively, this axiom states that when a trigger action has been performed, it would be equivalent to first performing this action, then adding the norm in which the norm effect is immediately active, and finally adding the norm again. The intuitive reason for using $Act_R \cup Act_T$ instead of Act_R as the repair actions is to avoid interference with n when at a later moment a trigger action occurs again, i.e. if we would use Act_R the axiom would not be sound.

(Completeness) In order to show completeness, we define the following translation function $\tau_2 : \mathcal{L}_2 \to \mathcal{L}_0$, where below the function $\tau_1 : \mathcal{L}_1 \to \mathcal{L}_0$ is the function as defined in the proof of Theorem 2, which we use to reduce formulas of the form $[\mathsf{n}]p$ (if $\mathsf{n} \in \mathcal{N}_1$). Moreover, we define $f : \mathcal{N}_2 \to \mathcal{N}_1$ as:

$$f((Act_T, \varphi, \pm v, Act_R)) := (\varphi, \pm v, Act_R \cup Act_T)$$

The reduction is defined as follows:

$$\begin{aligned}
\tau_2(p) &= p \\
\tau_2([\mathsf{n}]p) &= \tau_1([\mathsf{n}]p) && (\mathsf{n} \in \mathcal{N}_1) \\
\tau_2([\mathsf{n}]p) &= p && (\mathsf{n} \in \mathcal{N}_2) \\
\tau_2(\neg\psi) &= \neg\tau_2(\psi) \\
\tau_2(\psi_1 \vee \psi_2) &= \tau_2(\psi_1) \vee \tau_2(\psi_2) \\
\tau_2(Do(\alpha)\psi) &= Do(\alpha)\tau_2(\psi) \\
\tau_2([\mathsf{n}]\neg\psi) &= \tau_2(\neg[\mathsf{n}]\psi) \\
\tau_2([\mathsf{n}](\psi_1 \vee \psi_2)) &= \tau_2(([\mathsf{n}]\psi_1) \vee ([\mathsf{n}]\psi_2)) \\
\tau_2([\mathsf{n}]Do(\alpha)\psi) &= \tau_2(Do(\alpha)\psi) && (\mathsf{n} \in \mathcal{N}_1 \,\&\, \alpha \in Act_R) \\
\tau_2([\mathsf{n}]Do(\alpha)\psi) &= \tau_2(Do(\alpha)[\mathsf{n}]\psi) && (\mathsf{n} \in \mathcal{N}_1 \,\&\, \alpha \notin Act_R) \\
\tau_2([\mathsf{n}]Do(\alpha)\psi) &= \tau_2(Do(\alpha)[f(\mathsf{n})]([\mathsf{n}]\psi)) && (\mathsf{n} \in \mathcal{N}_2 \,\&\, \alpha \in Act_T) \\
\tau_2([\mathsf{n}]Do(\alpha)\psi) &= \tau_2(Do(\alpha)[\mathsf{n}]\psi) && (\mathsf{n} \in \mathcal{N}_2 \,\&\, \alpha \notin Act_T) \\
\tau_2([\mathsf{n}_1]([\mathsf{n}_2]\psi)) &= \tau_2([\mathsf{n}_1]\tau_2([\mathsf{n}_2]\psi))
\end{aligned}$$

This time around, it is slightly harder to show that τ_2 is well-defined. This is because τ_2 on input $[\mathsf{n}]Do(\alpha)\psi$ for which $\mathsf{n} \in \mathcal{N}_2$ and $\alpha \in Act_T$ 'generates' an extra norm $f(\mathsf{n})$, i.e., it can occur that on a certain input, the number of dynamic operators increases. However, since the number of norms from the set $\mathcal{N}_2$ in the scope of a norm from $\mathcal{N}_1$ never increases and always eventually decreases, we know that the translation will eventually return a well-formed formula from $\mathcal{L}_0$. It is possible to show that for any $\varphi \in \mathcal{L}_2$ we have that $\vdash_{\mathrm{L}_2} \varphi \leftrightarrow \tau_2(\varphi)$ (where again $\vdash_{\mathrm{L}_2} \varphi$ stands for "φ is provably in **L2**"), which we omit due to space constraints. We can now show that from the assumption that $\models \varphi$ we have $\vdash_{\mathrm{L}_2} \varphi$. By soundness of $\mathcal{L}_2$, we have from $\vdash_{\mathrm{L}_2} \varphi \leftrightarrow \tau_2(\varphi)$ that $\models \varphi \leftrightarrow \tau_2(\varphi)$. Thus, from our assumption that $\models \varphi$, we have $\models \tau_2(\varphi)$. By the completeness of system **L0**, we have $\vdash_{\mathrm{L}_0} \tau_2(\varphi)$. This implies that also $\vdash_{\mathrm{L}_2} \tau_2(\varphi)$, since **L2** contains all the rules and axioms of **L0**. Since we have shown that $\vdash_{\mathrm{L}_2} \varphi \leftrightarrow \tau_2(\varphi)$ and since $\vdash_{\mathrm{L}_2} \tau_2(\varphi)$, we can apply rule modus ponens to acquire $\vdash_{\mathrm{L}_2} \varphi$.

In words, axiom 1 states that an update does not modify the state of affairs of the current state. Axiom 2 and 3 both state that the dynamic update is a *function*: given a normative system and an update, a new normative system is uniquely determined. Axiom 4 is especially noteworthy. This axiom states that when a trigger action has been performed, it would be equivalent to first performing this action, then adding the norm in which the norm effect is immediately active, and finally adding the norm again. This axiom elegantly showcases the connection between our two types of norms, namely it both contains constructs from language $\mathcal{N}_1$ and $\mathcal{N}_2$. Axiom 5 states the property that if the norm is not triggered by a certain action, it does not matter if we perform the update before or after this action. All these axioms are only concerned with the trigger actions, while the axioms of $\mathcal{L}_1$ are only concerned with the repair actions. Together they form a complete axiomatization which considers both the trigger and repair actions.

We again have a similar result about decidability.

Theorem 5 *The logical language $\mathcal{L}_2$ with corresponding system* **L2** *is decidable.*

Proof. We use the translation function $\tau_2 : \mathcal{L}_2 \to \mathcal{L}_0$ from our completeness proof in Theorem 4 to establish the following correspondence:

$$\vdash_{\mathrm{L}_2} \varphi \quad \Leftrightarrow \quad \vdash_{\mathrm{L}_2} \tau_2(\varphi) \quad \Leftrightarrow \quad \vdash_{\mathrm{L}_0} \tau_2(\varphi)$$

Thus, in order to determine whether $\vdash_{\mathrm{L}_2} \varphi$, it is both necessary and sufficient to show that $\vdash_{\mathrm{L}_0} \tau_2(\varphi)$. However, since logic $\mathcal{L}_0$ with corresponding system **L0** is decidable and since τ_2 is well defined and can be effectively computed, decidability again transfers to our logic and system.

Note that this proof highlights an easier way to establish the validity of a sentence. To show for example that $\vdash_{\mathrm{L}_2} [\mathsf{n}](Do(\alpha_2)Do(\alpha_1)v)$, where $\mathsf{n} = (\{1\}, \top, +v, \{2\})$, we can perform the following translation:

$$\begin{aligned}
\tau_2([\mathsf{n}](Do(\alpha_2)Do(\alpha_1)v)) &= \\
\tau_2(Do(\alpha_2)[\mathsf{n}](Do(\alpha_1)v)) &= \\
Do(\alpha_2)\tau_2([\mathsf{n}](Do(\alpha_1)v)) &= \\
Do(\alpha_2)\tau_2(Do(\alpha_1)[f(\mathsf{n})]([\mathsf{n}]v)) &= \\
Do(\alpha_2)Do(\alpha_1)\tau_2([f(\mathsf{n})]([\mathsf{n}]v)) &= \\
Do(\alpha_2)Do(\alpha_1)\tau_2([f(\mathsf{n})]\tau_2([\mathsf{n}]v)) &= \\
Do(\alpha_2)Do(\alpha_1)\tau_2([f(\mathsf{n})]v) &= \\
Do(\alpha_2)Do(\alpha_1)(\top \vee v)
\end{aligned}$$

To easily establish that:

$$\vdash_{\mathrm{L}_0} Do(\alpha_2)Do(\alpha_1)(\top \vee v)$$

This concludes our work on the logical language $\mathcal{L}_2$ and **L2**, extending $\mathcal{L}_1$ and **L1**. We saw that even though our second logic extends the first, decidability is still guaranteed.

5 Related work and Conclusions

In this paper we introduced various expressive types of norms and devised new dynamic modal logics that are able to characterize the dynamics of such norms. We have shown that these logics are sound, complete and decidable, which means that we can use automated theorem provers to prove properties of norm dynamics and their effects on normative systems. The new forms of norms may have various effects and non-effects on the behaviour of a system which might initially be overlooked. With a formal and rigid framework like the one provided in this paper, we can automatically verify these effects.

The updates we consider in this paper are inspired by the kind of updates we may see in dynamic epistemic logic such as Public Announcement Logic [16], but note that the argument (a norm, which is not just a logical formula but a complex structure consisting of a condition, effect and repair actions) and result (a norm-aligned system) of our norm update is something completely different from what we may find there. Another important difference is the fact that these updates are performed on *pointed* Kripke structures, while updates from [16] are performed on just Kripke structures. Lastly, as an important note, we want to express that although these norm update operations may at first glance be related to Propositional Dynamic Logic (PDL) [11], the updates we consider are performed on Kripke structures with duplicated states and not just on valuations.

This research is related to a multitude of different papers concerned with norm change. The work presented in [7] uses the syntactic approach, where norm change is considered as an operation on the underlying "code" that constitutes the normative system. The work in [10] shows a way to program norm change, but not how to reason about this. In [4], the authors show, given a conditional norm, how a transition system can be unfolded to a tree in which the norm is into effect. The work in [13] considers norm-updates performed on normative systems, but does not consider an axiomatized logic to reason about these updates. In [6], Aucher et al. give a logical account of ought-to-be norm change via the notions of context expansion and contraction. In our work, we perform norm updates on pointed labelled transition systems to give a logical account of a much more general form of norm change via updates of the whole system at hand. To capture the dynamics of norms we have employed ideas from Dynamic Epistemic Logic, [16], where they use dynamic modal logic to characterise the dynamics of Kripke structures, albeit in a totally different (viz. epistemic) setting.

For future research there are various directions. Firstly, we can apply this framework to other well known frameworks of normative systems, such as the coloured systems considered in [14]. Secondly, we can look at other types of norms, such as norms with deadlines, and see how we can relate these to the current model. Lastly, the relation between the current framework and other frameworks of (dynamic) normative theory (e.g., [4, 8, 2, 7]) needs to be explored further.

REFERENCES

[1] Thomas Ågotnes, Wiebe van der Hoek, Juan A. Rodríguez-Aguilar, Carles Sierra, and Michael Wooldridge, 'On the logic of normative systems', in *Proceedings of the twentieth International Joint Conference on Artificial Intelligence (IJCAI 2007)*, pp. 1175–1180, (2007).

[2] Thomas Ågotnes, Wiebe van der Hoek, and Michael Wooldridge, 'Normative system games', in *Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2007)*, pp. 881–888, (2007).

[3] Carlos E. Alchourrón, Peter Gärdenfors, and David Makinson, 'On the logic of theory change: Partial meet contraction and revision functions', *Journal of Symbolic Logic*, **50**(2), 510–530, (1985).

[4] Natasha Alechina, Mehdi Dastani, and Brian Logan, 'Reasoning about normative update', in *Proceedings of the Twenty-Third International Joint Conference on Artificial Intelligence (IJCAI 2013)*, pp. 20–26, (2013).

[5] Alan Ross Anderson, 'A reduction of deontic logic to alethic modal logic', *Mind*, **67**(265), 100–103, (1958).

[6] Guillaume Aucher, Davide Grossi, Andreas Herzig, and Emiliano Lorini, 'Dynamic context logic', in *Proceedings of Logic, Rationality and Interaction (LORI 2009)*, pp. 15–26, (2009).

[7] Guido Boella, Gabriella Pigozzi, and Leendert van der Torre, 'Normative framework for normative system change', in *Proceedings of the 8th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2009)*, pp. 169–176, (2009).

[8] Nils Bulling and Mehdi Dastani, 'Verifying normative behaviour via normative mechanism design', in *Proceedings of the Twenty-Second International Joint Conference on Artificial Intelligence (IJCAI 2011)*, pp. 103–108, (2011).

[9] Brian F. Chellas, *Modal Logic: An Introduction*, Cambridge University Press, 1980.

[10] Mehdi Dastani, John-Jules Ch. Meyer, and Nick A. M. Tinnemeier, 'Programming norm change', *Journal of Applied Non-Classical Logics*, 151–180, (2012).

[11] Michael J. Fischer and Richard E. Ladner, 'Propositional modal logic of programs', in *Proceedings of the Ninth Annual ACM Symposium on Theory of Computing*, pp. 286–294, (1977).

[12] Max Knobbout and Mehdi Dastani, 'Reasoning under compliance assumptions in normative multiagent systems', in *Proceedings of the 11th International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, pp. 331–340, (2012).

[13] Max Knobbout, Mehdi Dastani, and John-Jules Ch. Meyer, 'Reasoning about dynamic normative systems', in *Logics in Artificial Intelligence - 14th European Conference (JELIA 2014)*, pp. 628–636, (2014).

[14] Marek J. Sergot, 'Action and agency in norm-governed multi-agent systems', in *Proceedings of the Engineering Societies in the Agents World VIII (ESAW 2007)*, pp. 1–54, (2007).

[15] Yoav Shoham and Moshe Tennenholtz, 'On the synthesis of useful social laws for artificial agent societies', in *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI 1992)*, pp. 276–281, (1992).

[16] Hans van Ditmarsch, Wiebe van der Hoek, and Barteld Kooi, *Dynamic Epistemic Logic, volume 337 of Synthese Library Series*, Springer, 2007.