An Arrow-based Dynamic Logic of Normative Systems and Its Decidability

Hans van Ditmarsch[1], Louwe Kuijer[2], and Mo Liu[3]

[1] University of Toulouse, CNRS, IRIT hans.van-ditmarsch@irit.fr
[2] University of Liverpool louwe.kuijer@liverpool.ac.uk
[3] University of Lorraine, LORIA mo.liu@loria.fr

Abstract. Normative arrow update logic (NAUL) is a logic that combines normative temporal logic (NTL) and arrow update logic (AUL). In NAUL, norms are interpreted as arrow updates on labeled transition systems with a CTL-like logic. We show that the satisfiability problem of NAUL is decidable with a tableau method and it is in EXPSPACE.

Keywords: normative system · arrow update logic · tableau method

1 Introduction

Deontic logic is the study of rules, norms, obligations and permissions, through logical means [17,5,7,10,13], and this has also been extensive investigated in dynamic modal logics [16,6,20,12,11]. In the field of deontic logic, there is a sub-field that studies rules or norms by comparing the situation where a rule is not in effect, or not being followed, to the situation where the rule/norm is obeyed. There is no universally accepted name for this sub-field, but "social laws" [18,19,9] and "normative systems" [1,3] are often used. We will use the term *normative systems*, and refer to the behavioural restrictions under consideration as *norms*.

A logic of normative systems is concerned with what things agents are capable of doing, and what they are allowed to do if a norm is enacted. It therefore requires a model of agency at its core. Any model of agency will do, but the most commonly used choices are labeled transition systems with a CTL-like logic of agency [8] and outcome function transition systems with ATL-like logic [4]. Here, we will follow the CTL-style approach as *normative temporal logic* [2]. This means that a model is a labeled transition system, i.e., it contains a set S of states and a set $\{R(a) \mid a \in \mathcal{A}\}$ of accessibility relations, where $R(a) \subseteq S \times S$. A transition $(s_1, s_2) \in R(a)$, is an action or an agent that changes the state of the world from s_1 to s_2.

In order to choose a course of action, we need to decide whether we should adopt a norm and then check if an action is allowed by the norm. Whether an action a is allowed may depend on a logical condition φ before the action takes place, so on the situation in s_1, and also may depend on a logical condition ψ after the action took place, so on a condition satisfied in s_2. We refer to s_1 as the *source* of the action, to φ as a *source condition*, to s_2 as the *target*, and to ψ as a *target condition*. For norms with both source and target conditions one cannot reduce multiple source conditions to one (for example by taking the disjunction), nor multiple target condition to one. A norm in our formalism will be therefore represented by a list of clauses, each with a source condition and a target condition. This is as in *arrow update logic* [14,21]. The arrow eliminating updates in arrow update logic now correspond to adherence to norms.

We will also introduce more complex ways to describe norms, so we will refer to such a list of clauses as an *atomic norm*. We distinguish four ways to combine norms. If N_1 and N_2 are norms, then

- $-N_1$ is the negation of N_1, and allows exactly those actions that are disallowed by N_1,
- $N_1 + N_2$ is the additive combination of N_1 and N_2, and allows exactly those actions that are allowed by N_1 or N_2,
- $N_1 \times N_2$ is the multiplicative combination of N_1 and N_2, and allows exactly those actions that are allowed by both N_1 and N_2.
- $N_1 \circ N_2$ is the sequential composition of N_1 and N_2, and allows exactly those actions that are allowed by N_2 in the transition system restricted to those actions that are allowed by N_1.

We further distinguish *static* from *dynamic* applications of norms. A liveness condition such as "if the norm N is obeyed, then φ is guaranteed to be true at every time in the future" can be formalized in two ways, which we denote $[N]G\varphi$ (dynamic) and $G_N\varphi$ (static). The difference lies in whether the norm N is assumed to hold during the evaluation of φ: when evaluating $[N]G\varphi$, everything inside the scope of $[N]$ is considered in the transition system restricted to the actions allowed by N. When evaluating $G_N\varphi$, on the other hand, the "forever in the future" operator G is evaluated in the system restricted to N-allowed actions, but φ is evaluated in the non-restricted system.

The dynamic operator $[N]$ can be expressed using only the static operators, and the combined norms can be expressed using only atomic norms. They do not affect the expressivity. However, the combined and dynamic norms affect the succinctness of the language, and thus the complexity of decision problems. The logic will be called NAUL, Normative Arrow Update Logic (preliminarily presented as [15]). We will now formally define its syntax and semantics and then investigate the complexity of satisfiability with a tableau method.

2 Language and Semantics

Let $\mathcal{A}$ be a finite set of agents and $\mathcal{P}$ a countably infinite set of propositional variables.

Definition 1. *The formulas of $\mathcal{L}_{NAUL}$ are given by*

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid [N]\varphi \mid \Box_N\varphi \mid G_N\varphi \mid F_N\varphi$$
$$\mathcal{N} ::= (\varphi, B, \varphi) \mid \overline{(\varphi, B, \varphi)} \mid \mathcal{N}, (\varphi, B, \varphi) \mid \mathcal{N}, \overline{(\varphi, B, \varphi)}$$
$$N ::= \mathcal{N} \mid -N \mid N + N \mid N \times N \mid N \circ N$$

where $p, \in \mathcal{P}$ and $B \subseteq \mathcal{A}$.

Remark 1. In NAUL we use only three temporal operators: $\Box_N$, G_N and F_N. These temporal operators include an implicit universal quantification over all paths, so we could have denoted them in a more CTL-like fashion as AX_N, AG_N and AF_N. Operators corresponding to the other temporal operators from CTL can be defined in NAUL. For example, $E(\varphi_1 U_N \varphi_2)$ can be defined as $\neg G_{(\varphi_1, \mathcal{A}, \top)\times N}\neg\varphi_2$.

In NAUL, the set of subformulas ($SubF$) or subnorms ($SubN$) of a formula φ (or a norm N) includes all formulas or norms *occur* in φ (or N).

Strictly speaking a norm of type $\mathcal{N}$ is a list of clauses, but we abuse notation by identifying it with the set of its clauses. Additionally, we use a number of abbreviations. We refer to norms of type $\mathcal{N}$ as *atomic norms* and norms of type N simply as *norms*. Note that every atomic norm is also a norm.

Definition 2. *We use $\wedge, \rightarrow, \leftrightarrow, \bigwedge, \bigvee$ and $\Diamond_N$ in the usual way as abbreviations. Furthermore, we use $\hat{G}_N$ and $\hat{F}_N$ as abbreviations for $\neg G_N \neg$ and $\neg F_N \neg$. We write $\Box_B$ for $\Box_{(\top,B,\top)}$, G_B for $G_{(\top,B,\top)}$ and F_B for $F_{(\top,B,\top)}$. Finally, we use $\Box, G$ and F for $\Box_{\mathcal{A}}$, $G_{\mathcal{A}}$ and $F_{\mathcal{A}}$.*

Definition 3. *A* model $\mathcal{M}$ *is a triple $\mathcal{M} = (S, R, v)$ where S is a set of states, $R : \mathcal{A} \rightarrow 2^{S\times S}$ maps each agent to an accessibility relation on S, and $v : \mathcal{P} \rightarrow 2^S$ is a valuation. A* pointed model *is a pair $(\mathcal{M}, s)$ where $\mathcal{M} = (S, R, v)$ is a model and $s \in S$.*

A pair $(s_1, s_2) \in R(a)$ is also called *transition* in $\mathcal{M}$. It is denoted $s_1 \xmapsto{a_1} s_2$. A *path* in $\mathcal{M}$ is a (possibly infinite) sequence $s_1 \xmapsto{a_1} s_2, s_2 \xmapsto{a_2} s_3, \cdots$ of transitions in $\mathcal{M}$. A path P' *extends* a path P if P is an initial segment of P'. The semantics of $\mathcal{L}_{NAUL}$ are given by the following two interdependent definitions.

Definition 4. *Let $\mathcal{M} = (S, R, v)$ be a relational model and N a norm. A transition $s_1 \xmapsto{a} s_2$* satisfies N in $\mathcal{M}$ *if one of the following is holds:*

1. *N is an atomic norm, there is a positive clause $(\varphi, B, \psi) \in N$ such that $\mathcal{M}, s_1 \models \varphi$, $a \in B$ and $\mathcal{M}, s_2 \models \psi$. Furthermore, there is no negative clause $\overline{(\varphi, B, \psi)} \in N$ such that $\mathcal{M}, s_1 \models \varphi$, $a \in B$ and $\mathcal{M}, s_2 \models \psi$,*
2. *N is of the form $-N_1$ and $s_1 \xmapsto{a} s_2$ does not satisfy N_1,*
3. *N is of the form $N_1 + N_2$ and $s_1 \xmapsto{a} s_2$ satisfies N_1 or N_2 in $\mathcal{M}$,*
4. *N is of the form $N_1 \times N_2$ and $s_1 \xmapsto{a} s_2$ satisfies N_1 and N_2 in $\mathcal{M}$,*
5. *N is of the form $N_1 \circ N_2$, $s_1 \xmapsto{a} s_2$ satisfies N_1 in $\mathcal{M}$ and the transition $s_1 \xmapsto{a} s_2$ satisfies N_2 in $\mathcal{M} * N_1$.*

A path $s_1 \overset{a_1}{\longmapsto} s_2 \overset{a_2}{\longmapsto} s_3 \cdots$ *is an* N-path in $\mathcal{M}$ *if every transition* $s_i \overset{a_i}{\longmapsto} s_{i+1}$ *in the path satisfies* N *in* $\mathcal{M}$. *An* N*-path is* full in $\mathcal{M}$ *if there is no* N*-path in* $\mathcal{M}$ *that extends it.*

When the model $\mathcal{M}$ is clear from context, we say simply that a transition satisfies N or that a path is an N-path.

Definition 5. *Let* $\mathcal{M} = (S, R, v)$ *be a transition system and* $s \in S$. *The relation* $\models$ *is given as follows.*

$$\begin{array}{lll}
\mathcal{M}, s \models p & \Leftrightarrow s \in v(p) \textit{ for } p \in \mathcal{P} \\
\mathcal{M}, s \models \neg\varphi & \Leftrightarrow \mathcal{M}, s \not\models \varphi \\
\mathcal{M}, s \models \varphi_1 \vee \varphi_2 & \Leftrightarrow \mathcal{M}, s \models \varphi_1 \textit{ or } \mathcal{M}, s \models \varphi_2 \\
\mathcal{M}, s \models \Box_N \varphi & \Leftrightarrow \mathcal{M}, s' \models \textit{for every transition } s \longmapsto s' \textit{ that satisfies } N \\
\mathcal{M}, s \models G_N \varphi & \Leftrightarrow \textit{for every } N\textit{-path } P \textit{ starting in } s \textit{ and} \\
& \quad \textit{every } s' \in P \textit{ we have } \mathcal{M}, s' \models \varphi \\
\mathcal{M}, s \models F_N \varphi & \Leftrightarrow \textit{for every full } N\textit{-path } P \textit{ starting in } s \textit{ there is} \\
& \quad \textit{some } s' \in P \textit{ such that } \mathcal{M}, s' \models \varphi \\
\mathcal{M}, s \models [N]\varphi & \Leftrightarrow \mathcal{M} * N, s \models \varphi
\end{array}$$

where $\mathcal{M} * N = (S, R * N, v)$ *and, for every* $a \in \mathcal{A}$,

$$R * N(a) = \{(s, s') \in R(a) \mid s \overset{a}{\longmapsto} s' \textit{ satisfies } N\}.$$

Recall that the single state s is a degenerate path with no transitions. So every transition in s satisfies every norm N, so it is an N-path. As a result, $\mathcal{M}, s \models G_N\varphi$ implies $\mathcal{M}, s \models \varphi$.

3 Example: Self-driving Cars

We will give a simple example of NAUL. Suppose we have a racetrack where a number of self-driving cars operate. We want to equip cars with norms that will guarantee that they avoid

(a) collisions with each other and stationary objects;
(b) "deadlock" situations where no one can act.

Let *coll* be the proposition variable that represents "a collision happens". Note that situations where no one can act are represented by $\Box\bot$.

For (a), we create a norm N_c such that if no collision has occurred then it should prevent collisions for every point in the future. N_c is therefore successful if we have $\neg coll \to [N_c]G\neg coll$. The simplest way is to disallow any action, then N_c is $(\bot, \mathcal{A}, \bot)$. However, we would like to let N_c allow at least one action to avoid deadlock. Thus we take $N_c := (\top, \mathcal{A}, \neg F coll)$. It is indeed successful as we have $\models \neg coll \to [N_c]G\neg coll$.

For (b), we interpret it as "there must be some available action that is not only possible but also allowed", and then we construct a N_d such than $[N_d]G\Diamond\top$ holds. we should take $N_d := (\top, \mathcal{A}, \neg F\Box\bot)$. This gives us $\models \neg F\Box\bot \to [N_d]G\Diamond\top$.

In other words, as long as there is an infinite path the norm N_d forces agents to follow such a path.

For combining N_c and N_d, $N_c \times N_d$ allows agents to perform actions that result in a situation where movement, while possible, is disallowed because it will lead to a collision. The sequential combination solves this problem: the norm $N_c \circ N_d$ allows exactly those actions that lead to neither collisions nor situations where agents cannot or are not allowed to act. In other words, we have $\models \neg F(\mathit{coll} \vee \Box\bot) \to [N_c \circ N_d]G(\neg \mathit{coll} \wedge \Diamond\top)$.

The self-driving cars example is also useful for illustrating the difference between the static operators $\Box_N$, G_N, and F_N on the one hand, and the dynamic operator $[N]$ on the other. We have $\mathcal{M}, s \models G_N\varphi$ if φ holds after every sequence of action that starts in s and is allowed by N. Importantly, during the evaluation of φ it is not assumed that everyone follows N. We have $\mathcal{M}, s \models [N]G\varphi$ if, under the assumption that all agents follow N permanently from now on, every sequence of actions leads to a φ state. In this case, during the evaluation of φ, we do assume that all agents follow N.

Sometimes we may require that N_c not only avoids collisions, but also situations where a single mistake could cause a collision. We cannot phrase this stronger success condition as $[N_c]\varphi$ for any φ. After all, the φ in $[N_c]\varphi$ is evaluated under the assumption that all agents follow the norm N_c—so no mistakes are made. This is where the static operator G_{N_c} is useful. Consider the formula $G_{N_c}(\neg \mathit{coll} \wedge \Box\neg \mathit{coll})$. The $\Box$ in that formula is not evaluated under the assumption that the agents follow N_c, so $G_{N_c}(\neg \mathit{coll} \wedge \Box\neg \mathit{coll})$ holds exactly if every sequence of actions allowed by N_c leads to a state where there is no collision and no single action can cause a collision.

4 Satisfiability Problem

Definition 6. *The satisfiability problem for NAUL is defined as follows:*

- ***Input****: an NNF formula φ.*
- ***Output****: YES if and only if there is a model $(\mathcal{M}, s)$ such that $\mathcal{M}, s \models \varphi$.*

In this section, we present a tableau method to show the satisfiability problem of NAUL is decidable. We will use *negation normal form* (NNF) of formulas or norms. An NNF formula only has negation on literals. An NNF norm only has negations on atomic norms instead of clauses.

Definition 7 (Negation normal form (NNF)). *Given a set of variables* $\mathbf{P}$ *and a finite set of agents $\mathcal{A}$.*

$$\varphi ::= p \mid \neg p \mid \varphi\wedge\varphi \mid \varphi\vee\varphi \mid \Box_N\varphi \mid \Diamond_N\varphi \mid G_N\varphi \mid \hat{G}_N\varphi \mid F_N\varphi \mid \hat{F}_N\varphi \mid [N]\varphi \mid \langle N\rangle\,\varphi$$

$$\mathcal{N} ::= (\varphi, a, \varphi) \mid \mathcal{N}, (\varphi, a, \varphi)$$

$$N ::= \mathcal{N} \mid \overline{\mathcal{N}} \mid N + N \mid N \times N \mid N \circ N$$

where $p \in \mathbf{P}$, $a \in \mathcal{A}$.

Theorem 1. *Every NAUL-formula or norm can be transformed to an equivalent formula or norm in NNF.*

Proof. For NAUL-formulas, it can be shown easily by an induction. As for atomic norms, since the order of clauses in an atomic norm does not matter, given an atomic NAUL-norm $\mathcal{N}$, and $\mathcal{N}^+$ as all positive clauses, $\mathcal{N}^-$ as all negative clauses of $\mathcal{N}$, clearly $\mathcal{N}$ is equivalent to $\mathcal{N}^+ + \mathcal{N}^-$ which is an NNF norm. As for the negations of combined norms, we have the following transformations:

- $\overline{\overline{N}} = N$
- $\overline{N_1 + N_2} = \overline{N_1} \times \overline{N_2}$
- $\overline{N_1 \times N_2} = \overline{N_1} + \overline{N_2}$
- $\overline{N_1 \circ N_2} = \overline{N_1} + N_1 \circ \overline{N_2}$

Given an NAUL-formula φ or NAUL-norm N, the time of transforming it into an NNF formula φ' or NNF norm N' and the size of ψ or N' is polynomial in the size of φ or N.

4.1 Tableau method

We introduce some concepts related to tableau method.

Definition 8 (Term). *There are two types of terms:*

F-term $\langle s; \lambda; \varphi\rangle$ *where* $s \in S$, λ *is a sequence of norms,* φ *is a formula. It means the model has been updated by* λ *and* φ *is true on* s.

N-term $\langle s_1 \overset{a}{\mapsto} s_2; \lambda; \eta\rangle$ *where* $s_1, s_2 \in S$, λ, η *are sequences of norms. It means the transition* $s_1 \overset{a}{\mapsto} s_2$ *satisfies* η *successively after the model is updated by* λ.

Definition 9 (Tableau). *A tableau* T *is a structure* $T = (W, V, E, \pi)$ *where* W *is an infinite set, and* V *is a finite set,* E *is a binary relation on* V. *Given a set of terms* L, $\pi : V \to \mathbf{P}(L)$ *is a labelling map.*

Let $A, C_1, \cdots, C_n$ *be sets of terms. A tableau rule is represented as*

$$\frac{A}{C_1 \mid \cdots \mid C_n}$$

Above the line, A *is the antecedent; below the line, there are consequents. A tableau rule is applicable on a node if the node has terms as an instance of the antecedent. If there are multiple consequents after applying a rule, one need to choose one of them.*

Definition 10 (Interpretability). *Given a model* $\mathcal{M} = (S, R, v)$, *it interpret (noted as* $\vDash_T$*) a set of terms* T *if any term in* T *satisfies:*

- $\mathcal{M} \vDash_T \langle s; \lambda; \varphi\rangle$ *if and only if* $\mathcal{M} * \lambda, s \vDash \varphi$.
- $\mathcal{M} \vDash_T \langle s_1 \overset{a}{\mapsto} s_2; \lambda; \eta\rangle$ *if and only if* $s_1 \overset{a}{\mapsto} s_2$ *satisfies* η *on* $\mathcal{M} * \lambda$.

A set of terms T *is interpretable if there exists a model* $\mathcal{M}$ *such that* $\mathcal{M}$ *interprets all terms in* T.

Definition 11. *Given a tableau T, we define an order $\prec$ on all terms of T as*

- $\langle s; \lambda; \varphi\rangle \prec \langle s; \lambda'; \psi\rangle$ *if φ is a subformula of ψ.*
- $\langle s \overset{a}{\mapsto} s'; \lambda; \eta\rangle \prec r \langle s; \lambda'; \varphi\rangle$ *if η is a parameter of some operator in φ.*
- $\langle s; \lambda; \varphi\rangle \prec \langle s \overset{a}{\mapsto} s'; \lambda'; \eta\rangle$ *if φ is in some clause of η.*
- $\langle s \overset{a}{\mapsto} s'; \lambda; \eta\rangle \prec \langle s \overset{a}{\mapsto} s'; \lambda; \eta'\rangle$ *if η is a sub-norm of η';*
- $\langle s \overset{a}{\mapsto} s'; \lambda; N'\rangle \prec \langle s \overset{a}{\mapsto} s'; \lambda'; N\rangle$ *if λ is an initial segment of λ'.*

Now we define the tableau rules for NAUL. We omit terms which remain the same after applying a certain rule. Let ϵ be the norm $(\top, \mathcal{A}, \top)$ after which nothing is updated.

Definition 12 (Tableau rules).

$$\textit{(lit)}\ \frac{\langle s;\lambda;p\rangle}{\langle s;\epsilon;p\rangle} \qquad \frac{\langle s;\lambda;\neg p\rangle}{\langle s;\epsilon;\neg p\rangle}\ (\wedge)\ \frac{\langle s;\lambda;\varphi\wedge\psi\rangle}{\langle s;\lambda;\varphi\rangle,\langle s;\lambda;\psi\rangle}\ (\vee)\ \frac{\langle s;\lambda;\varphi\vee\psi\rangle}{\langle s;\lambda;\varphi\rangle \mid \langle s;\lambda;\psi\rangle}$$

$$\textit{(G)}\ \frac{\langle s;\lambda;G_N\varphi\rangle}{\langle s;\lambda;\varphi\rangle,\langle s;\lambda;\Box_N G_N\varphi\rangle} \qquad \textit{(}\hat{G}\textit{)]}\frac{\langle s;\lambda;\hat{G}_N\varphi\rangle}{\langle s;\lambda;\varphi\rangle \mid \langle s;\lambda;\Diamond_N\hat{G}_N\varphi\rangle}$$

$$\textit{(F)}\ \frac{\langle s;\lambda;F_N\varphi\rangle}{\langle s;\lambda;\varphi\rangle \mid \langle s;\lambda;\Diamond_N\top\rangle,\langle s;\lambda;\Box_N F_N\varphi\rangle}$$

$$\textit{(}\hat{F}\textit{)}\ \frac{\langle s;\lambda;\hat{F}_N\varphi\rangle}{\langle s;\lambda;\varphi\rangle,\langle s;\lambda;\Box_N\bot\rangle \mid \langle s;\lambda;\varphi\rangle,\langle s;\lambda;\Diamond_N\hat{F}_N\varphi\rangle}$$

$$(\Diamond)\ \frac{\langle s;\lambda;\Diamond_N\varphi\rangle}{\langle s';\lambda;\varphi\rangle,\langle s\overset{a_1}{\mapsto}s';\lambda;N\rangle \mid \cdots \mid \langle s';\lambda;\varphi\rangle,\langle s\overset{a_n}{\mapsto}s';\lambda;N\rangle}$$

$$(\Box)\ \frac{\langle s;\lambda;\Box_N\varphi\rangle,\langle s\overset{a}{\mapsto}s';\epsilon;\lambda\rangle}{\langle s';\lambda;\varphi\rangle,\langle s\overset{a}{\mapsto}s';\lambda;N\rangle \mid \langle s\overset{a}{\mapsto}s';\lambda;-N'\rangle} \qquad \textit{($-N'$ is the NNF of $-N$)}$$

$$\textit{(Dyn)}\ \frac{\langle s;\lambda;[N]\varphi\rangle}{\langle s;\lambda,N;\varphi\rangle} \qquad \frac{\langle s;\lambda;\langle N\rangle\varphi\rangle}{\langle s;\lambda,N;\varphi\rangle}$$

$$\textit{(At)}\ \frac{\langle s\overset{a_i}{\mapsto}s';\lambda;\mathcal{N}\rangle}{\langle s;\lambda;\varphi_i\rangle,\langle s';\lambda;\psi_i\rangle} \qquad \textit{where } (\varphi_i,a_i,\psi_i)\in\mathcal{N}$$

$$\textit{(Neg)}\ \frac{\langle s\overset{a}{\mapsto}s';\lambda;-\mathcal{N}\rangle}{\langle s;\lambda;\bigwedge_{j\in K_1}\varphi'_j\rangle,\langle s';\lambda;\bigwedge_{j\in K_2}\psi'_j\rangle \mid \cdots} \qquad \textit{(@)}$$

$$\textit{(Add)}\ \frac{\langle s\overset{a}{\mapsto}s';\lambda;N_1+N_2\rangle}{\langle s\overset{a}{\mapsto}s';\lambda;N_1)\rangle \mid \langle s\overset{a}{\mapsto}s';\lambda;N_2)\rangle} \qquad \textit{(Multi)}\frac{\langle s\overset{a}{\mapsto}s';\lambda;N_1\times N_2\rangle}{\langle s\overset{a}{\mapsto}s';\lambda;N_1)\rangle,\langle s\overset{a}{\mapsto}s';\lambda;N_2)\rangle}$$

$$\textit{(Seq)}\ \frac{\langle s\overset{a}{\mapsto}s';\lambda;N_1\circ N_2\rangle}{\langle s\overset{a}{\mapsto}s';\lambda,N_1;N_2\rangle} \qquad \textit{(DN)}\ \frac{\langle s\overset{a}{\mapsto}s';\lambda,N_1;N_2\rangle}{\langle s\overset{a}{\mapsto}s';\lambda;N_1\rangle}$$

(@): rule Neg is branching over all $K_1, K_2 \subseteq [1, n]$ such that $K_1 \cup K_2 = \{i \mid (\varphi_i, a, \psi_i) \in \mathcal{N}\}$, and $K_1 \cap K_2 = \varnothing$, and φ'_j and ψ'_j are the NNF of resp. $\neg\varphi_j$ and $\neg\psi_j$ with $(\varphi_j, a, \psi_j) \in \mathcal{N}$.

(lit), $(\wedge)$ and $(\vee)$ are Boolean rules. $(G),(F),(\hat{G}),(\hat{F})$ handle temporal modalities. (G) says if we have $G_N\varphi$ at a word s, then we have φ as well as $\Box_N G_N\varphi$ at s. (F) says if we have $F_N\varphi$ at s, then whether we have φ, or s has some N-successor ($\Diamond_N\top$ is true) and $\Box_N F_N\varphi$. $(\hat{G})$ says if we have $\hat{G}_N\varphi$ at s, then whether we have φ or $\Diamond_N\hat{G}_N\varphi$ at s. $(\hat{F})$ says if we have $\hat{F}_N\varphi$ at s, then we have φ at s and whether s has no N-successor or it has $\Diamond_N\hat{F}_N\varphi$. $(\Diamond)$ says if we have $\Diamond_N\varphi$ at s, then we can choose an agent $a\in\mathcal{A}$ to "assume" that there is a transition $s\overset{a}{\mapsto}s'$ satisfying N and we have φ at s'. Note that $(\Diamond)$ is the only rule that generates new states and whether a state can be actually generated will be examined later. $(\Box)$ says if we have $\Box_N\varphi$ at s and transition $s\overset{a}{\mapsto}s'$ exists, then whether we have φ at s' or $s\overset{a}{\mapsto}s'$ does not satisfy N. (Dynamic) handle dynamic operators. It says if we have $[N]\varphi$ (or $\langle N\rangle\varphi$) at s updated by λ, then we have φ at s updated by λ then by N.

The other rules handle norms. (Atomic) says if we have atomic norm $\mathcal{N}$ for $s\overset{a_i}{\mapsto}s'$ where a_i occurs in some clause $(\varphi_i,a_i,\psi_i)\in\mathcal{N}$, then we have φ_i at s and ψ_i at s'. (Neg) says if we have $\overline{\mathcal{N}}$ for $s\overset{a}{\mapsto}s'$, then given $\{i\mid(\varphi_i,a,\psi_i)\in\mathcal{N}\}$ we choose some $K_1,K_2\subseteq[1,n]$ such that $K_1\cup K_2=\{i\mid(\varphi_i,a,\psi_i)\in\mathcal{N}\}$ and $\bigwedge_{j\in K_1}\neg\varphi$ is at s and $\bigwedge_{j\in K_2}\neg\psi$ is at s'. As a result, none of clause in $\mathcal{N}$ will be satisfied by $s\overset{a}{\mapsto}s'$. (Add), (Multi) and (Seq) are standard with respect to Def 4. (DN) says if $s\overset{a}{\mapsto}s'$ satisfies some norm N_2 after updating by λ,N_1, then it satisfies N_1 after updating by λ. A special case of (DN) is

$$(\text{DN*})\ \frac{\langle s\overset{a}{\mapsto}s';\lambda;N\rangle}{\langle s\overset{a}{\mapsto}s';\epsilon;\lambda\rangle}$$

(DN*) says if transition $s\overset{a}{\mapsto}s'$ is updated by λ, then it satisfies λ.

Besides above tableau rules, we also need principles to delete inconsistent states, set an order of applying rules, and avoid infinite consequents

Definition 13 (Tableau principles). *Given an NNF formula φ, we start from the root with label $\langle s_0;\epsilon;\varphi\rangle$. We have the following the principles of generating a tableau of φ:*

(Inc) If a node has inconsistent literals, then mark it as "deleted". If all consequents are marked deleted, then mark the antecedent as deleted. In particular, if one node have no consequent then mark it as deleted directly.
(Exh) We should apply rules to terms with respect to one state until no rule is applicable on that state. When no rule is applicable on a state s, we mark s as "exhausted". After that, we can apply rules to terms on its successors.
(Cyc) When a state s are marked as "exhausted', one needs to check if there some exhausted ancestor s^* of s which has the same F-terms with s on some node t^*. If so, we should add $(t,t^*)\in E$ and mark s as "exhausted" as well. If a state s is merged with some ancestor, then all successors of s are also marked as "exhausted", and we stop to explore any term with respect to these successors further. In addition, let $\sim\subseteq S\times S$ be an equivalent relation, and use $s^*\sim s$ to "merge" these two state to a reflexive state.

(Eve$\hat{G}$) If all consequences of an antecedent t are marked as deleted, then mark t as "deleted". If $\hat{G}_N\varphi$ is in some term of a node t with respect to a state s, and there is no reachable state from s such that φ occurs in some term, then mark t as "deleted".
(EveF) If $F_N\varphi$ is in some term of a node t with respect to a state s, and there exists a full branch from s on which φ does not occur in any term of state on that branch, then mark t as "deleted".

If there is no rule applicable any more, the procedure of generating the tableau terminate, and the tableau is complete. If the root of a complete tableau T is not marked as "deleted", then we call a path from the root to a leaf node on T an open branch. If a complete tableau has at least one open branch, then we call it an open tableau.

Proposition 1. *For any NNF-formula φ, the procedure of generate a tableau for φ will terminate.*

4.2 Soundness and Completeness

Proposition 2 (Soundness). *Given an NNF-formula φ, if φ is satisfiable then there is an open tableau rooted at $(s_0; \epsilon; \varphi)$.*

Proof. We show all tableau rules preserve interpretability. If a tableau rule has multiple consequents, then as least one of them is interpretable.

- (lit) and $(\wedge)$ preserve interpretability obviously. For $(\vee)$, if one of its consequences is interpretable, then so is the antecedent.
- For the rules (G), (F), $(\hat{G})$, $(\hat{F})$ and $(Dynamic)$, it can be shown by semantics. We present the case of (F) as an example. Suppose $\mathcal{M} * \lambda, s \vDash F_N\varphi$. By semantics, for every full N-path P starting from s, there is some $s' \in P$ such that $\mathcal{M} * \lambda, s' \vDash \varphi$. Since s in every N-path starting from s, it is sufficient if $\mathcal{M} * \lambda, s \vDash \varphi$. Otherwise, we have there is some N-successor s' of s such that $\mathcal{M} * \lambda, s' \vDash F_N\varphi$. In this case, $\mathcal{M} * \lambda, s \vDash \Diamond_N\top$ and $\mathcal{M} * \lambda, s \vDash \Diamond_N F_N\varphi$.
- $(\Diamond_N)$: Suppose $\mathcal{M} * \lambda, s \vDash \Diamond_N\varphi$. By semantics, there is a transition $s \overset{a}{\mapsto} s'$ satisfying N and $\mathcal{M} * \lambda, s' \vDash \varphi$ for some $a \in \mathcal{A}$.
- $(\Box_N)$: Suppose $\mathcal{M} * \lambda, s \vDash \Box_N\varphi$ and $s \overset{a}{\mapsto} s'$ satisfies λ on $\mathcal{M}$. If $s \overset{a}{\mapsto} s'$ satisfies N on $\mathcal{M} * \lambda$, then by semantics we have $\mathcal{M} * \lambda, s' \vDash \varphi$. If $s \overset{a}{\mapsto} s'$ does not satisfy N on $\mathcal{M} * \lambda$, then it satisfies $-N$ on $\mathcal{M} * \lambda$.
- (Atomic): Suppose $s \overset{a_i}{\mapsto} s'$ satisfies $\mathcal{N}$ on $\mathcal{M} * \lambda$. It follows that $\mathcal{M} * \lambda, s \vDash \varphi_i$ and $\mathcal{M} * \lambda, s' \vDash \psi_i$. Thus $\mathcal{M} \vDash_T \langle s; \lambda; \varphi_i \rangle, \langle s'; \lambda; \psi_i \rangle$.
- (Neg): Suppose $s \overset{a_i}{\mapsto} v$ satisfies $\overline{\mathcal{N}}$ on $\mathcal{M} * \lambda$. It follows that $s \overset{a}{\mapsto} v$ satisfies no clause with respect to a in $\mathcal{N}$. Thus let $K_1 = \{i \mid \mathcal{M} * \lambda, v \vDash \neg\varphi_i$ for any $(\varphi_i, a, \psi_i) \in \mathcal{N}\}$ and $K_2 = \{i \mid \mathcal{M}*\lambda, v \vDash \neg\psi_i$ for any $(\varphi_i, a, \psi_i) \in \mathcal{N}\}$. Therefore, we have $K_1 \cup K_2 = \{i \mid (\varphi_i, a_i, \psi_i) \in \mathcal{N}\}$, and $\mathcal{M} \vDash_T \langle v; \lambda; \bigwedge_{i \in K_1} \neg\varphi_i \rangle$ and $\mathcal{M} \vDash_T \langle v; \lambda; \bigwedge_{i \in K_2} \neg\psi_i \rangle$.
- (Add), (Multi), (Seq) is straightforward by the definition.

– (DN): Suppose $s \overset{a}{\mapsto} s'$ satisfies N_2 on $\mathcal{M}*\lambda*N_1$. By definition, if $s \overset{a}{\mapsto} s'$ is on $\mathcal{M}*\lambda*N_1$, then it satisfies N_1 on $\mathcal{M}*\lambda$ as well. Thus $\mathcal{M} \vDash_T \langle s \mapsto s'; \lambda; N_1\rangle$.

Note that the trace-back links by (Cyc) only connect nodes with the same terms on the same state. Thus interpretability is preserved as well.

Suppose φ is satisfiable, then there is a pointed model $\mathcal{M}, s \vDash \varphi$. Let s be s_0, then there is an open tableau rooted at $\langle s_0; \epsilon; \varphi\rangle$.

Proposition 3 (Completeness). *Given an NNF-formula φ, if there is an open tableau rooted at $(s_0; \epsilon; \varphi)$, then φ is satisfiable.*

Proof. Suppose there is an open tableau T rooted at $\langle s_0; \lambda; \varphi\rangle$. Let T^* be a full branch on T. We construct a model $\mathcal{M} = (S, R, v)$ where

– $S = \{[s] \mid \langle s; \lambda; \psi\rangle \text{ is in } T^*\}$
– $R = \{s \overset{a}{\mapsto} s' \mid \left\langle s \overset{a}{\mapsto} s'; \epsilon; \epsilon\right\rangle \in T^*\} \cup \{s \overset{a}{\mapsto} s \mid a \in \mathcal{A}, |[s]| > 1\}$
– $v(s) = \{p \mid \langle s; \epsilon; p\rangle \in T^*\}$

where $[s] = \{s' \in S \mid s \sim s'\}$.

We show the following claims:

1. if $\langle s; \lambda; \psi\rangle$ is in T^*, then $\mathcal{M} * \lambda, s \vDash \psi$.
2. if $\left\langle s_1 \overset{a}{\mapsto} s_2; \lambda; N\right\rangle$ is in T^*, then $s_1 \overset{a}{\mapsto} s_2$ is in $\mathcal{M}*\lambda$ and satisfies η on $\mathcal{M}*\lambda$.

Make an induction on all terms by the order $\prec$ in Def. 11 to show the above claims. For Claim 2,

– If $\left\langle s_1 \overset{a_i}{\mapsto} s_2; \lambda; \mathcal{N}\right\rangle \in T^*$ where $\mathcal{N} = (\varphi_1, a_1, \psi_1), \cdots, (\varphi_n, a_n, \psi_n)$, $i \in [1, n]$, then by (Atomic) rule $\langle s_1; \lambda; \varphi_i\rangle, \langle s; \lambda; \psi_i\rangle \in T^*$. Then by IH, we have $\mathcal{M} * \lambda, s_1 \vDash \varphi$, $\mathcal{M} * \lambda, s_2 \vDash \psi$. Thus $s_1 \overset{a_i}{\mapsto} s_2$ satisfies $\mathcal{N}$ on $\mathcal{M} * \lambda$.
– If $\left\langle s_1 \overset{a}{\mapsto} s_2; \lambda; \overline{\mathcal{N}}\right\rangle \in T^*$ where $\mathcal{N} = (\varphi_1, a_1, \psi_1), \cdots, (\varphi_n, a_n, \psi_n)$, $i \in [1, n]$, then by (Neg) rule, $\left\langle s_1; \lambda; \bigwedge_{i \in K_1} \neg\varphi_i\right\rangle \in T^*$ and $\left\langle s_2; \lambda; \bigwedge_{i \in K_2} \neg\psi_i\right\rangle \in T^*$ for some disjoint $K_1 \cup K_2 = \{i \mid (\varphi_i, a, \psi_i) \in \mathcal{N}\}$. Thus, no φ_i and ψ_i are satisfied simultaneously so that no clause in $\mathcal{N}$ with respect to a is satisfied. Therefore, $s_1 \overset{a}{\mapsto} s_2$ satisfies $\overline{\mathcal{N}}$.
– The cases of $N_1 + N_2$, $N_1 \times N_2$, and $N_1 \circ N_2$ are straightforward by IH.

For Claim 1,

– If $\langle s; \lambda; p\rangle \in T^*$, then by by the rule (lit), $\langle s; \epsilon; p\rangle \in T^*$. Thus $\mathcal{M}, s \vDash p$, and then $\mathcal{M} * \lambda, s \vDash p$. Similarly, if $\langle s; \lambda; \neg p\rangle \in T^*$, then $\mathcal{M} * \lambda, s \vDash \neg p$. The boolean cases, dynamic case and $\langle s; \lambda; \Diamond_N \psi\rangle \in T^*$ are straightforward by IH.
– If $\langle s; \lambda; \Box_N \psi\rangle \in T^*$, then for any $\left\langle s \overset{a}{\mapsto} s'; \epsilon; \lambda\right\rangle \in T^*$, by $(\Box_N)$ rule, $\langle s'; \lambda; \psi\rangle \in T^*$ or $\left\langle s \overset{a}{\mapsto} s'; \lambda; -N\right\rangle \in T^*$. If $\langle s'; \lambda; \psi\rangle \in T^*$, then by IH $\mathcal{M} * \lambda, s' \vDash \psi$; If $\left\langle s \overset{a}{\mapsto} s'; \lambda; -N\right\rangle \in T^*$, then by Claim 1, $s \overset{a}{\mapsto} s'$ satisfies $-N$, that is to say, $s \overset{a}{\mapsto} s'$ does not satisfy N. Thus by semantics, $\mathcal{M}, s \vDash \Box_N \varphi$.

– Suppose $\langle s;\lambda;G_N\psi\rangle \in T^*$. Let $P = s \overset{a_1}{\mapsto} s_1 \overset{a_2}{\mapsto} s_2 \cdots \overset{a_n}{\mapsto} s_{n+1}$ be any N-path starting from s. We show that $\langle s';\lambda;\Box_N G_N\psi\rangle \in T^*$ and $\mathcal{M} * \lambda, s' \vDash \psi$ for any $s' \in P$ by induction on $n+1$. By (G) rule, $\langle s;\lambda;\psi\rangle \in T^*$ and $\langle s;\lambda;\Box_N G_N\psi\rangle \in T^*$. Since $\langle s;\lambda;\psi\rangle \in T^*$, by IH we have $\mathcal{M} * \lambda, s \vDash \psi$. Assume $\langle s_n;\lambda;\Box_N G_N\psi\rangle \in T^*$ and $\mathcal{M} * \lambda, s_n \vDash \psi$. By (G) rule again, we have $\langle s_{n+1};\lambda;\Box_N G_N\psi\rangle \in T^*$. Since $s_n \overset{a_n}{\mapsto} s_{n+1} \in R$, we have $\left\langle s_n \overset{a_n}{\mapsto} a_{n+1};\lambda;N\right\rangle \in T^*$. Then by $(\Box_N)$ rule, we have $\langle w_{n+1};\lambda;\psi\rangle$. By IH, we have $\mathcal{M}*\lambda, s_{n+1} \vDash \psi$. Thus for every $s' \in P$, we have $\mathcal{M}*\lambda, s' \vDash \psi$. As P is arbitrary, by semantics $\mathcal{M}*\lambda, s \vDash G_N\psi$. For terms with $F_N\psi, \hat{G}_N\psi$ and $\hat{F}_N\psi$, it is routine by IH.

Theorem 2. *For any NNF formula φ, φ is satisfiable if and only if there is an open tableau rooted at (s_0, ϵ, φ).*

Therefore, the satisfiability problem of NAUL is decidable. We wil show its upper bound is in EXPSPACE.

Theorem 3. *The satisfiability problem of NAUL is in EXPSPACE.*

Proof. Let φ be an NNF formula, and T be an open tableau for φ. We show the following claims:

1. The depth of T is at most exponential.
2. The width of T is at most double exponential.
3. The procedure can be done in double exponential amount of time.

Note that tableau rules does not decompose formulas strictly, thus the sizes of formulas in the consequents may be larger than the sizes of formulas in the antecedents. However we can give an upper bound of how many terms a single open branch in T has.

The *agenda* $Ag(\varphi)$ of a formula φ is the smallest set containing ϵ, $SubF(\varphi)$ as well as $SubN(\varphi)$ and satisfying the following conditions:

– If $\psi \in Ag(\varphi)$, then $\neg\psi^* \in Ag(\varphi)$;
– If $G_N\psi \in Ag(\varphi)$, then $\Box_N G_N\psi \in Ag(\varphi)$;
– If $\hat{G}_N\psi \in Ag(\varphi)$, then $\Diamond_N \hat{G}_N\psi \in Ag(\varphi)$;
– If $F_N\psi \in Ag(\varphi)$, then $\Diamond_N\top, \Box_N F_N\psi \in Ag(\varphi)$;
– If $\hat{F}_N\psi \in Ag(\varphi)$, then $\Box_N\bot, \Diamond_N \hat{F}_N\psi \in Ag(\varphi)$;
– If $N \in Ag(\varphi)$, then $\overline{N}^* \in Ag(\varphi)$.

Clearly, the cardinality of $Ag(\varphi)$ is polynomial in $|\varphi|$.

Proof of 1: For any F-term $\langle s;\lambda;\psi\rangle$ or N-term $\left\langle s \overset{a}{\mapsto} s';\lambda;N\right\rangle$ occurring in T when s is marked as exhausted, it can be shown that ψ, $N \in Ag(\varphi)$ and all elements of λ are in $Ag(\varphi)$ by examining every rule. Firstly, we could give an upper bound of states in one open branch. We have shown the formulas of all F-terms are in $Ag(\varphi)$. Since two exhausted states get merged if they have the same F-terms, we can get at most exponential many states in the size of φ. Secondly, we could give an upper bound of how many transitions are generated from one state. Note that the $(\Diamond_N)$ rule is the only rule that generates new transitions. The frequency that $(\Diamond_N)$ rule is applied is bounded by the size of φ, one state has at most polynomial many arrows in the size of φ.

Therefore, the upper bounds of the amount of F-terms and N-terms are both at most exponential in the size of φ. One open branch has at most exponential depth as well, as there are at most exponentially many exhausted states with the same F-terms. This is because if a state is merged with some ancestor, then we will stop exploring terms of it. Therefore, the frequency that each state can be merged is no more than the number of paths starting from it. Since each exhausted state has polynomial many arrows to other states, it can be merged at most exponentially many times. In short, the depth of one open branch is in at most exponential.

Proof of 2: The rule leads to exponentially many branches is (Neg). Given an atomic norm $\mathcal{N}$, $|\overline{\mathcal{N}}|$ is bounded by $|\varphi|$. The cardinality of branches is in $O(2^{|\varphi|})$. As there are at most exponentially many terms in one branch, the width of T is in $O(2^{|\varphi|^2})$, so at most double exponential in the size of φ.

Proof of 3: The algorithm contains: applying tableau rules, checking, marking and pruning the tableau by principles, transforming formulas with negation into NNF. For each branch, as there are at most exponentially many terms in the size of φ, all of three procedures above can be done in an exponential amount of time. To be specific, applying rules contains searching suitable premises and executing. The input of searching is the power set of labels on some node, which is exponential in the size of φ and the executions of applying rules are no more than the amount of terms; the input of checking inconsistency and states with the same terms is exponential in the size of φ and can be done in exponential time; the frequency of transforming NNF formulas is at most exponential and each transformation can be done in polynomial time.

To sum up, as we can reuse the space for each open branch, the procedure is in EXPSPACE.

5 Conclusion

We have presented a logic named *normative arrow update logic* (NAUL). In NAUL, we can combine norms in three ways: additive, multiplicative and sequential. We can also distinguish static and dynamic ways to consider norms. We have shown that the satisfiability problem of NAUL is decidable via a tableau method and the complexity of this problem is in EXPSPACE. For the further research, firstly, we conjecture the satisfiability problem of NAUL is EXPSPACE-hard but have no proof yet. Secondly, we are interested in finding tractable fragments of NAUL. Lastly, it may be interesting to develop a variant of arbitrary arrow update logic (AAUL) [21] for normative systems. It would have quantifier over norms and express “there is some norm that guarantees φ”.

References

1. T. Ågotnes, W. van der Hoek, J. A. Rodríguez-Aguilar, C. Sierra, and M. Wooldridge. On the logic of normative systems. In *Proc. of 20th IJCAI*, pages 1175–1180, 2007.
2. T. Ågotnes, W. van der Hoek, J.A. Rodríguez-Aguilar, C. Sierra, and M. Wooldridge. A Temporal Logic of Normative Systems. In David Makinson, Jacek Malinowski, and Heinrich Wansing, editors, *Towards Mathematical Philosophy: Papers from the Studia Logica conference Trends in Logic IV*, pages 69–106. Springer Netherlands, Dordrecht, 2009.
3. T. Ågotnes, W. van der Hoek, and M. Wooldridge. Robust normative systems and a logic of norm compliance. *Logic Journal of the IGPL*, 18:4–30, 2010.
4. R. Alur, T.A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49:672–713, 2002.
5. A.R. Anderson and O.K. Moore. The formal analysis of normative concepts. *American Sociological Review*, 22:9–17, 1957.
6. P. Bartha. Conditional obligation, deontic paradoxes, and the logic of agency. *Annals of Mathematics and Articial Intelligence*, 9(1-2):1–23, 1993.
7. R.M. Chisholm. Contrary-to-duty imperatives and deontic logic. *Analysis*, 24:33–36, 1963.
8. E. M. Clarke and E. A. Emerson. Design and synthesis of synchronization skeletons using branching-time temporal logic. In Dexter Kozen, editor, *Logics of Programs*, pages 52–71, 1981. LNCS 131.
9. D. Fitoussi and M. Tennenholtz. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119:61–101, 2000.
10. D. Føllesdal and R. Hilpinen. Deontic Logic: An Introduction. In Risto Hilpinen, editor, *Deontic Logic: Introductory and Systematic Readings*, Synthese Library, pages 1–35. Springer Netherlands, Dordrecht, 1971.
11. A. Herzig, E. Lorini, F. Moisan, and N. Troquard. A dynamic logic of normative systems. In T. Walsh, editor, *Proceedings of the Twenty-second International Joint Conference on Artificial Intelligence (IJCAI11)*, pages 228–233, 2011.
12. J.F. Horty. *Agency and deontic logic.* Oxford University Press, 2001.
13. A.J.I. Jones and M. Sergot. On the characterisation of law and computer systems: The normative systems perspective. In *Deontic Logic in Computer Science: Normative System Specification*, pages 275–307, 1993.
14. B. Kooi and B. Renne. Arrow update logic. *Review of Symbolic Logic*, 4(4):536–559, 2011.
15. L.B. Kuijer. An arrow-based dynamic logic of norms. In *3rd International Workshop on Strategic Reasoning (SR 2015)*, 2015.
16. J. Ch. Meyer. A different approach to deontic logic: Deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, 29:109–136, 1988.
17. A. Ross. Imperatives and logic. *Theoria*, 7:53–71, 1941.
18. Y. Shoham and M. Tennenholtz. On the synthesis of useful social laws for artificial agent societies. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI 1992)*, pages 276–281, 1992.
19. Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: off-line design. *Artificial Intelligence*, 73:231–252, 1995.
20. R. van der Meyden. The dynamic logic of permission. *Journal of Logic and Computation*, 6(3):465–479, 1996.
21. H. van Ditmarsch, W. van der Hoek, B. Kooi, and L.B. Kuijer. Arbitrary arrow update logic. *Artificial Intelligence*, 242:80–106, January 2017.