Norm-regulated Transition System Situations

Magnus Hjelmblom[1,2]

[1]*Faculty of Engineering and Sustainable Development, University of Gävle, SE-80176 Gävle, Sweden*

[2]*Department of Computer and Systems Sciences, Stockholm University, Forum 100, SE-16440, Kista, Sweden*

Keywords: transition system, multi-agent system, norm-regulated, norm-governed, normative system.

Abstract: Many multi-agent systems (MAS) and other kinds of dynamic systems may be modeled as transition systems, in which actions are associated with transitions between different system states. This paper presents an approach to normative systems in this context, in which the permission or prohibition of actions is related to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1,...,x_\nu$ in a state. It introduces the notion of a norm-regulated transition system situation, which is intended to represent a single step in the run of a (norm-regulated) transition system. The normative framework uses an algebraic representation of conditional norms and is based on a systematic exploration of the possible types of state transitions with respect to $d(x_1,...,x_\nu)$. A general-level Java/Prolog framework for norm-regulated transition system situations is currently being developed.

1 INTRODUCTION

Many dynamic systems, including multi-agent systems (MAS), may be modeled as transition systems, in which the actions of an agent are associated with transitions between different states of the system. There is a number of different approaches to normative systems in this context. The permission or prohibition of a specific action in a transition system is naturally connected to permissible or prohibited transitions between states of the system, and norms (sometimes referred to as 'social laws') may then be formulated as restrictions on states and state transitions.

This paper will introduce the notion of a *norm-regulated transition system situation*, which is intended to represent a single step in the run of a (norm-regulated) transition system. The permission or prohibition of actions in this framework is related to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1,...,x_\nu$ in a state. The framework uses an algebraic representation of conditional norms, based on the representation used in the norm-regulated DALMAS architecture (see Previous Work, Sect. 1.2). The novel feature presented here is primarily an extension to the DALMAS's normative framework, based on a systematic exploration of the possible types of state transitions with respect to $d(x_1,...,x_\nu)$. A norm-regulated transition system situation is easily instrumentalized into a general-level Prolog module that can be used to implement a wide range of specific norm-regulated dynamic systems.

Important norm-related issues such as enforcement of norms, norm change and consistency of normative systems are beyond the scope of this paper; however, the approach presented here is general in nature, and may be combined with many different approaches to, e.g., norm enforcement. The term 'agent' will be frequently used for some sort of 'acting entity' within a dynamic system, but no special assumptions are made about for example autonomy, reasoning capability, architecture, and so on.

1.1 The Setting

A labelled transition system (LTS) is usually defined (see for example (Craven and Sergot, 2008, p. 174)) as an ordered 3-tuple $\langle S,E,R\rangle$ where S is a non-empty set of *states*; E is a set of transition *labels*, often called *events*; and $R \subseteq S\times E\times S$ is a non-empty set of labelled *transitions*. If (s,ε,s') is a transition, s is the initial state and s' is the resulting state of ε. An event ε is *executable* in a state s if there is a transition $(s,\varepsilon,s')\in R$, and *non-deterministic* if there are transitions $(s,\varepsilon,s')\in R$ and $(s,\varepsilon,s^*)\in R$ with $s'\neq s^*$. A *path* (or *run*) of length m ($m\geq 0$) of a labelled transition system is a sequence $s_0\varepsilon_0 s_1\cdots s_{m-1}\varepsilon_{m-1}s_m$ such that, for $i\in 1..m$, $(s_{i-1},\varepsilon_{i-1},s_i)\in R$.

In the following, we restrict our attention to transition systems in which all events are deterministic. This means that, for each state s, the labels associated with the outgoing transitions from s are distinct. Furthermore, we assume that a ν-ary condition d is true or false on ν agents $x_1,...,x_\nu\in\Omega$ in s, where Ω is a set of agents associated with s; this will be written $d(x_1,...,x_\nu;s)$. In the special case that the sequence of agents is empty, i.e. $\nu=0$, d represents a proposition which is true or false in s.

1.2 Related Work

This section will give a brief overview of different approaches to the design of normative systems and the formulation of norms. A common feature of many approaches is the idea to partition states and (possibly) transitions into two categories, for example 'permitted' and 'non-permitted'. This may be accomplished with the use of if-then-else rules or constraints on the states and/or the transitions between states. The Ballroom system in (Gaertner et al., 2007) and the anticipatory system for plot development guidance in (Laaksolahti and Boman, 2002) both serve as examples of this approach. Some approaches are purely algebraic or based on modal logics, for example temporal or deontic logic. The DALMAS architecture (see Previous Work below) for norm-regulated MAS is based on an algebraic approach to the representation of normative systems. Dynamic deontic logic (Meyer, 1987) and Dynamic logic of permission (van der Meyden, 1996) are two well-known examples of the modal logic approach. Other examples are the combination of temporalised agency and temporalised normative positions (Governatori et al., 2005), in the setting of Defeasible Logic, and Input/Output Logic by Makinson and van der Torre (see for example (Makinson and van der Torre, 2007)). Vázquez-Salceda et al. use a language consisting of deontic concepts which can be conditional and can include temporal operators. They characterize norms by whether they refer to states (i.e., norms concerning that an agent sees to it that some condition holds) or actions (i.e., norms concerning an agent performing a specific action), whether they are conditional, whether they include a deadline, or whether they are norms concerning other norms. (Vazquez-Salceda et al., 2004) $n\mathcal{C}+$, an extension of the action language $\mathcal{C}+$, is employed within the context of 'coloured agent-stranded transition systems' (Craven and Sergot, 2008) to formulate two kinds of norms: *state permission laws* and *action permission laws*. A state permission law states that certain (types of) states are permissible or prohibited, while an action permission law states that specific (types of) transitions are permissible or prohibited in certain states. By picking out the component ('strand') corresponding to an individual agent's contribution to an event, different categories of non-compliant behaviour ('sub-standard' resp. 'unavoidably non-compliant' behaviour) can be distinguished. Cliffe et al. use Answer Set Programming (ASP) for representing institutional norms, as part of the representation and analysis of specifications of agent-based institutions. (Cliffe et al., 2006; Cliffe et al., 2007) In Deontic Petri nets, and variants thereof such as Organizational Petri nets, varying degrees of 'ideal' or 'sub-ideal' (more or less 'allowed' or 'preferred') behaviour is modeled by preference orderings on executions of Petri nets; see for example (Raskin et al., 1996; Combettes et al., 2006).

1.2.1 Previous Work: The DALMAS Architecture

DALMAS (Odelstad and Boman, 2004) is an abstract architecture for a class of (norm-regulated) multi-agent systems. A deterministic DALMAS is a simple multi-agent system in which the actions of an agent are connected to transitions between system states. In a deterministic DALMAS the agents take turns to act; only one agent at a time may perform an action. Therefore, each individual step in a run of the system may be represented by a transition system situation.

A DALMAS is formally described by an ordered 9-tuple, where the arguments are various sets, operators and functions which give the specific DALMAS its unique features. Of particular interest is the deontic structure-operator, which for each situation of the system determines an agent's *deontic structure* (i.e., the set of permissible acts) on the feasible acts in the current situation, and the preference structure-operator, which for each situation determines the *preference structure* on the permissible acts. In a *norm-regulated simple deterministic* DALMAS, the deontic structure consists of all acts that are not explicitly prohibited by a normative system; thereby employing what is often referred to as 'negative permission'. The preference structure consists of the most preferable (according to the agent's *utility function*) of the acts in the deontic structure. In other words, a DALMAS agent's behaviour is regulated by the combination of a normative system and a utility function. The normative system consists of conditional norms based on the Kanger-Lindahl theory of normative positions, expressed in an algebraic notation for norms. See for example (Lindahl, 1977; Lindahl and Odelstad, 2004; Odelstad, 2008) for an introduction. A general-level Java/Prolog implementation of the DALMAS architecture has been developed, to facilitate the implementation of specific systems. The **Colour & Form** system, the **Waste-collector** system and the **Forest Cleaner** system are three specific systems that have been implemented using this framework. The reader is referred to (Odelstad and Boman, 2004; Hjelmblom, 2008; Hjelmblom and Odelstad, 2009; Hjelmblom, 2011) for a description of these systems and their implementations.

2 NORMATIVE SYSTEMS AND TYPES OF STATE TRANSITIONS

Let us consider the transition from a state s to a following state s^+, and focus on the condition $d(x_1,...,x_\mathrm{v})$. To facilitate reading, X_v will be used as an abbreviation for the argument sequence $x_1,...,x_\mathrm{v}$. With regard to $d(X_\mathrm{v})$, there are four possible alternatives for the transition from s to s^+, since in s as well as in s^+, $d(X_\mathrm{v})$ or $\neg d(X_\mathrm{v})$ could hold:

$$\begin{aligned}
&\text{(I)} && d(X_\mathrm{v};s) \wedge d(X_\mathrm{v};s^+)\\
&\text{(II)} && \neg d(X_\mathrm{v};s) \wedge d(X_\mathrm{v};s^+)\\
&\text{(III)} && d(X_\mathrm{v};s) \wedge \neg d(X_\mathrm{v};s^+)\\
&\text{(IV)} && \neg d(X_\mathrm{v};s) \wedge \neg d(X_\mathrm{v};s^+)
\end{aligned}$$

Each alternative represents a basic type of transition with regard to the state of affairs $d(X_\mathrm{v})$. Following the notation in (Sergot, 2008), (I) could be written $0 : d(X_\mathrm{v}) \wedge 1 : d(X_\mathrm{v})$, (II) could be written $0 : \neg d(X_\mathrm{v}) \wedge 1 : d(X_\mathrm{v})$, and similarly for (III) and (IV).

2.1 Prohibition of State Transition Types

The set $\{$(I),(II),(III),(IV)$\}$ has 16 subsets. Let us explore the idea that each subset might represent a prohibited combination of basic transition types with regard to $d(X_\mathrm{v})$. We may then formulate (conditional) norms whose normative consequents are represented by such prohibited combinations. A specific event ε is taken to be prohibited if, in a certain state s, the normative system contains a norm which prohibits the type of transition represented by ε.

The subsets of $\{$(I),(II),(III),(IV)$\}$ are summarized in Table 1, where '-' denotes a permissible transition type while 'X' denotes a prohibited transition type. For each row, we form the disjunction of the prohibited transition types to obtain conditions on

Table 1: Possible Combinations of Prohibited State Transition Types.

	(I)	(II)	(III)	(IV)	
1	-	-	-	-	-
2	-	-	-	X	$\neg d(X_v;s) \wedge \neg d(X_v;s^+)$
3	-	-	X	-	$d(X_v;s) \wedge \neg d(X_v;s^+)$
4	-	-	X	X	$\neg d(X_v;s^+)$
5	-	X	-	-	$\neg d(X_v;s) \wedge d(X_v;s^+)$
6	-	X	-	X	$\neg d(X_v;s)$
7	-	X	X	-	$\neg(d(X_v;s) \leftrightarrow d(X_v;s^+))$
8	-	X	X	X	$\neg d(X_v;s) \vee \neg d(X_v;a(x,s))$
9	X	-	-	-	$d(X_v;s) \wedge d(X_v;s^+)$
10	X	-	-	X	$d(X_v;s) \leftrightarrow d(X_v;s^+)$
11	X	-	X	-	$d(X_v;s)$
12	X	-	X	X	$d(X_v;s) \vee \neg d(X_v;s^+)$
13	X	X	-	-	$d(X_v;s^+)$
14	X	X	-	X	$\neg d(X_v;s) \vee d(X_v;s^+)$
15	X	X	X	-	$d(X_v;s) \vee d(X_v;s^+)$
16	X	X	X	X	$\top$

Table 2: Meaningful Combinations of Prohibited State Transition Types.

(I)	(II)	(III)	(IV)	$\tau_j^\varepsilon d(X_v;s)$
-	-	-	-	-
-	-	X	-	$d(X_v;s) \wedge \neg d(X_v;s^+)$
-	-	-	X	$\neg d(X_v;s) \wedge \neg d(X_v;s^+)$
-	X	-	-	$\neg d(X_v;s) \wedge d(X_v;s^+)$
X	-	-	-	$d(X_v;s) \wedge d(X_v;s^+)$
-	-	X	X	$\neg d(X_v;s^+)$
-	X	X	-	$\neg(d(X_v;s) \leftrightarrow d(X_v;s^+))$
X	-	-	X	$d(X_v;s) \leftrightarrow d(X_v;s^+)$
X	X	-	-	$d(X_v;s^+)$

state transitions. E.g., row 5 represents the condition $\neg d(X_\nu;s) \wedge d(X_\nu;s^+)$; now, we make the corresponding stipulation that if $\neg d(X_\nu;s)$ and $d(X_\nu;s^+)$, and $(s,\varepsilon,s^+) \in R$, then the event ε is not permissible in state s. Another example is row 8, in which the disjunction of (II),(III) and (IV) yields the condition

$$\begin{gathered}\big(\neg d(X_\nu;s) \wedge d(X_\nu;s^+)\big) \vee \\ \big(d(X_\nu;s) \wedge \neg d(X_\nu;s^+)\big) \vee \\ \big(\neg d(X_\nu;s) \wedge \neg d(X_\nu;s^+)\big)\end{gathered}$$

which may be simplified to $\neg d(X_\nu;s) \vee \neg d(X_\nu;s^+)$. A closer look at Table 1 reveals, however, that not all rows seem to represent meaningful norms. For example, the disjunction (I) and (III) in row 11 yields the condition $(d(X_\nu;s) \wedge d(X_\nu;s^+)) \vee (d(X_\nu;s) \wedge \neg d(X_\nu;s^+))$, which may be simplified to $d(X_\nu;s)$. Let us try to make the corresponding stipulation that if $d(X_\nu;s)$, and $(s,\varepsilon,s^+) \in R$, then ε is not permissible in s. This would mean that, if $d(X_\nu;s)$, then no events are permissible in s. In other words, if $d(X_\nu;s)$ holds *before* the execution of an event, then *all* events are prohibited, no matter their result. This can hardly represent a meaningful norm; after all, an event in a state s can only affect the truth of $d(X_\nu)$ in a *following* state s^+. Another example is row 6, which expresses that if $\neg d(X_\nu;s)$, then no event is permissible, since neither of transition type (II) or (IV) is permissible. We conclude that all rows that contain {(I),(III)} or {(II),(IV)} represent norms that are not meaningful. Table 2 contains the rows (slightly reordered) that represent meaningful normative conditions. For each of the rows in Table 2 (except Π_1, which expresses no restrictions at all) we define a 'transition type operator' τ_j, $j \in \{2,2',4,4',5,6,6',7\}$, based on the corresponding row in the table:[1]

For all ν-ary conditions d and for all agents $x_1,\ldots,x_\nu$ and all states s and s' such that $(s,\varepsilon,s') \in R$,

1. $\tau^\varepsilon_2 d(X_\nu;s)$ iff $[d(X_\nu;s) \wedge \neg d(X_\nu;s')]$
2. $\tau^\varepsilon_{2'} d(X_\nu;s)$ iff $[\neg d(X_\nu;s) \wedge \neg d(X_\nu;s')]$
3. $\tau^\varepsilon_4 d(X_\nu;s)$ iff $[\neg d(X_\nu;s) \wedge d(X_\nu;s')]$
4. $\tau^\varepsilon_{4'} d(X_\nu;s)$ iff $[d(X_\nu;s) \wedge d(X_\nu;s')]$
5. $\tau^\varepsilon_5 d(X_\nu;s)$ iff $\neg d(X_\nu;s')$
6. $\tau^\varepsilon_6 d(X_\nu;s)$ iff $\neg[d(X_\nu;s) \leftrightarrow d(X_\nu;s')]$
7. $\tau^\varepsilon_{6'} d(X_\nu;s)$ iff $[d(X_\nu;s) \leftrightarrow d(X_\nu;s')]$
8. $\tau^\varepsilon_7 d(X_\nu;s)$ iff $d(X_\nu;s')$

The 'transition type condition' $\tau^\varepsilon_j d(X_\nu;s)$ indicates whether or not, in state s, the event ε has transition type j with regard to $d(X_\nu)$. We note in passing that the following 'symmetry principles' hold (cf. the observation in (Odelstad and Boman, 2004, p. 148)):

1. $\tau^\varepsilon_2 d(X_\nu;s)$ iff $\tau^\varepsilon_4 \neg d(X_\nu;s)$
2. $\tau^\varepsilon_{2'} d(X_\nu;s)$ iff $\tau^\varepsilon_{4'} \neg d(X_\nu;s)$
3. $\tau^\varepsilon_5 d(X_\nu;s)$ iff $\tau^\varepsilon_7 \neg d(X_\nu;s)$
4. $\tau^\varepsilon_6 d(X_\nu;s)$ iff $\tau^\varepsilon_6 \neg d(X_\nu;s)$
5. $\tau^\varepsilon_{6'} d(X_\nu;s)$ iff $\tau^\varepsilon_{6'} \neg d(X_\nu;s)$

Next, we define a normative 'transition type prohibition operator' Π_j, $j \in \{1,2,2',4,4',5,6,6',7\}$, such that for all ν-ary conditions d and for all agents $x_1,\ldots,x_\nu$ and for all states s, and for all events ε such that $(s,\varepsilon,s') \in R$,

$$\Pi_j d(X_\nu;s) \text{ iff } [\tau^\varepsilon_j d(X_\nu;s) \text{ is forbidden}].$$

Now suppose that $\Pi_j d(X_\nu;s)$ holds (is 'in effect'), and that the corresponding transition type condition $\tau^\varepsilon_j d(X_\nu;s)$ also holds for some event ε. Then we forbid the transition (s,ε,s'): For all states s and all events ε such that $(s,\varepsilon,s') \in R$,

$$\begin{gathered}\mathit{Prohibited}_s(\varepsilon) \text{ if there exists a condition } c, \\ \text{a sequence of agents } x_1,\ldots,x_\nu, \\ \text{and a } j \in \{2,2',4,4',5,6,6',7\} \text{ such that} \\ \Pi_j d(x_1,\ldots,x_\nu;s) \;\&\; \tau^\varepsilon_j c(x_1,\ldots,x_\nu;s).\end{gathered}$$

The representation of a normative system $\mathcal{N}$ based on transition type conditions $\Pi_j d$ is similar to the one used in the DALMAS architecture (see Sect. 1.2.1); a norm in $\mathcal{N}$ is represented by an ordered pair $\langle G,C\rangle$ where G is the (descriptive) *ground* of the norm and C is its (normative) *consequence*.[2] For example, the elementary norm $\langle g,\Pi_j c\rangle$ represents the sentence

$$\begin{gathered}\forall x_1,x_2,...,x_\nu \in \Omega : \\ g(x_1,x_2,...,x_p;s) \rightarrow \Pi_j c(x_1,x_2,...,x_q;s)\end{gathered}$$

where Ω is the set of agents and $\nu = \max(p,q)$. If the condition specified by the ground of a norm is true in some situation, then the (normative) consequence of the norm is in effect in that situation. In the example above, if $\langle g,\Pi_j c\rangle$ is a norm in $\mathcal{N}$, and $g(x_1,x_2,...,x_p;s)$, and $\tau^\varepsilon_j c(x_1,\ldots,x_q;s)$, then $\mathit{Prohibited}_s(\varepsilon)$ according to $\mathcal{N}$:

[1]The numbering is based on the numbering used in (Hjelmblom, 2011).

[2]This view of normative systems is further developed and compared with other approaches in (Lindahl and Odelstad, 2012). It bears some resemblance to the treatment of norms in Input/Output Logic (Makinson and van der Torre, 2007); conditional norms are simply treated as ordered pairs, and are not assumed to bear truth-values. In Input/Output Logic however, it is the conditional as a whole that represents, for example, a permission or prohibition, while in Lindahl's and Odelstad's algebraic approach it is the consequence of the norm that is normative.

$$\begin{gathered}
\mathit{Prohibited}_s(\varepsilon) \text{ according } \mathcal{N} \\
\text{if there exists a condition } g \text{ and a condition } c \text{ and a} \\
j \in \{2,2',4,4',5,6,6',7\} \text{ such that } \langle g, \Pi_j c\rangle \text{ is a norm} \\
\text{in } \mathcal{N}\text{, and there exists } x_1,\ldots,x_\nu \text{ such that} \\
g(x_1,\ldots,x_\nu;s) \ \&\ \tau^\varepsilon_j c(x_1,\ldots,x_\nu;s).
\end{gathered}$$

2.2 Prohibition of Actions

By adding the requirements that (1) each event ε is of the form $x : a$, i.e., represents an action a performed by an individual agent x, and that (2) norms apply to an individual agent x in a state s, it is straightforward to interpret the prohibition of state transitions as prohibition of actions. A specific action a is taken to be prohibited for x in s if the normative system contains a norm which prohibits the type of transition represented by the event $\varepsilon = x : a$: $\mathit{Prohibited}_{x,s}(a)$ if $\mathit{Prohibited}_s(x : a)$. This idea will be further elaborated in the following section.

3 NORM-REGULATED TRANSITION SYSTEM SITUATIONS

In the following, we focus on an arbitrary state in a deterministic LTS, with the added requirement that each event ε represents an action performed by a single agent x. The term 'transition system situation' will be used for an ordered 5-tuple $\mathbf{S} = \langle x, s_0, A, \Omega, S\rangle$ characterized by a set of states S, an initial state $s_0 \in S$, an agent-set $\Omega = \{x_1, ..., x_n\}$, an acting ('moving') agent x, and an action-set $A = \{a_1, ..., a_m\}$. An event ε is of the form $x : a$, i.e., it refers both to an agent x and an action a. In this setting, an action a may be regarded as a function such that $a(x,s) = s^+$ means that s^+ is the resulting state when x performs act a in state s.[3] In the following, the abbreviation s^+ will be used for $a(x,s_0)$ when there is no need for an explicit reference to the action a and the acting agent x.

As indicated by Fig. 1, a transition system situation is intended to represent, for example, a 'snapshot' of a labelled transition system in which each transition is deterministic and represents the action of a single agent. In this case, s_0 represents an arbitrarily chosen state in the LTS, and S is the set of states reachable from s_0 by the transitions $x : a$, for all $a \in A$. At the same time, a transition system situation is designed to be general enough to also represent a step in a run of other kinds of dynamic systems, including systems modeled by finite automata (see for example

[3]Note that s_0 may also be an element of S, i.e. the action a may lead back to s_0.

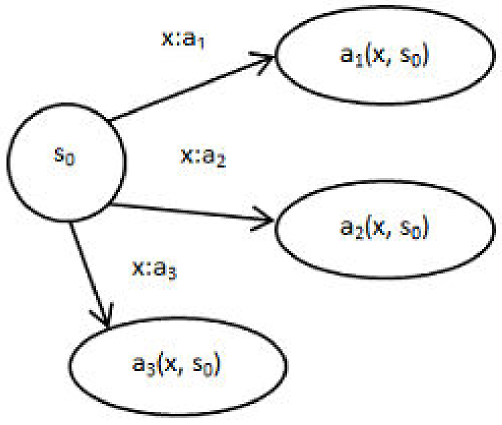

Figure 1: A state diagram for a transition system situation with three events.

(Laaksolahti and Boman, 2002)) or Petri nets, and deterministic DALMASes.

A *norm-regulated transition system situation* is represented by an ordered pair $\langle \mathbf{S}, \mathcal{N}\rangle$ where $\mathbf{S} = \langle x, s, A, \Omega, S\rangle$ is a transition system situation and $\mathcal{N}$ is a normative system. For each τ^ε_j, $j \in \{2,2',4,4',5,6,6',7\}$, we define a corresponding operator C^a_j: For all ν-ary conditions d and for all agents $x_1,\ldots,x_\nu,x_{\nu+1}$ and all actions $a \in A$,

$$\begin{gathered}
C^a_j d(x_1,...,x_\nu,x_{\nu+1};x,s) \text{ iff} \\
\tau^\varepsilon_j d(x_1,...,x_\nu;s) \ \&\ \varepsilon = x_{\nu+1} : a.
\end{gathered}$$

The extra argument $x_{\nu+1}$ denotes the agent to which the normative condition applies.[4] Next, we define a normative transition type prohibition operator P_j, $j \in \{1,2,2',4,4',5,6,6',7\}$, such that for all ν-ary conditions d and for all agents $x_1,\ldots,x_\nu,x_{\nu+1} \in \Omega$ and for all actions $a \in A$,

$$\begin{gathered}
P_j d(x_1,...,x_\nu,x_{\nu+1};x,s) \text{ iff} \\
[C^a_j d(x_1,...,x_\nu,x_{\nu+1};x,s) \text{ is forbidden}].
\end{gathered}$$

Now suppose that $P_j d(X_\nu,x_{\nu+1};x,s)$ holds (is 'in effect'), and that the corresponding transition type condition $C^a_j d(X_\nu,x_{\nu+1};x,s)$ also holds for some action a and some agent $x_{\nu+1}$. Then we forbid a for $x_{\nu+1}$: For all actions $a \in A$ and all agents $x_{\nu+1} \in \Omega$,

$$\begin{gathered}
\mathit{Prohibited}_{x,s}(x_{\nu+1},a) \text{ if there exists a condition } d, \\
\text{a sequence of agents } x_1,\ldots,x_\nu, \\
\text{and a } j \in \{2,2',4,4',5,6,6',7\}\text{, such that} \\
P_j d(x_1,\ldots,x_\nu,x_{\nu+1};x,s) \ \&\ C^a_j d(x_1,...,x_\nu,x_{\nu+1};x,s).
\end{gathered}$$

[4]It may seem unnecessary to introduce $x_{\nu+1}$; one might expect that the agent to which the normative condition applies is always the same as the 'moving' agent x. This distinction is made to allow for normative systems in which, for example, agents other than the 'moving' agent may perform punishments or other 'reaction acts'.

To ensure that the agent $x_{\nu+1}$ to which the norm applies is the same as the moving agent, we apply the 'move operator' M which transforms the condition g on ν agents in a state s to a condition on $\nu+1$ agents in the *situation* $\langle x,s\rangle$, characterized by s and the moving agent x, while at the same time identifying $x_{\nu+1}$ with the 'moving' agent x. (See (Odelstad and Boman, 2004; Hjelmblom, 2008) for a deeper explanation of the operator M.) Action a is prohibited for $x_{\nu+1}$ in the situation $\langle x,s\rangle$ if the normative system contains a norm which prohibits the type of transition represented by the event $x_{\nu+1} : a$:

$Prohibited_{x,s}(x_{\nu+1},a)$ according to $\mathcal{N}$
if there exists a condition g and a condition c
and a $j \in \{2,2',4,4',5,6,6',7\}$
such that $\langle Mg,P_jc\rangle$ is a norm in $\mathcal{N}$,
and there exists $x_1,\dots,x_\nu$ such that
$Mg(x_1,\dots,x_\nu,x_{\nu+1};s)$ & $C^a_jc(x_1,\dots,x_\nu,x_{\nu+1};s)$.

As in Sect. 2.1, a norm in $\mathcal{N}$ is represented by an ordered pair $\langle G,C\rangle$, where G is a descriptive, and C a normative, condition on a situation $\langle x,s\rangle$. For example, $\langle Mg,P_jc\rangle$ represents the sentence

$$\forall x_1,x_2,...,x_\nu,x_{\nu+1} \in \Omega :$$
$$Mg(x_1,x_2,...,x_p,x_{\nu+1};x,s) \rightarrow$$
$$P_jc(x_1,x_2,...,x_q,x_{\nu+1};x,s)$$

where Ω is the set of agents, $x_{\nu+1}$ is the agent to which the norm applies, x is the 'moving' agent in the situation $\langle x,s\rangle$, and $\nu = \max(p,q)$. If the condition specified by the ground of a norm is true in some situation, then the (normative) consequence of the norm is in effect in that situation. Since each situation for a DALMAS can be viewed as a transition system situation, it is straightforward to evolve the existing general-level Java/Prolog implementation of the DALMAS architecture (see Sect. 1.2.1) into a general-level implementation of a norm-regulated transition system situation. A norm is then represented by a Prolog term `n/3` of the form `n(Id/N,OpG*G,OpC*C)`, where `Id` is an identifier of a norm-system and `N` is an identifier of an individual norm. `OpG*G` is a compound term representing an operator `OpG` applied to (the functor of) a state condition predicate `G`, forming the norm's ground. Similarly, `OpC*C` represents the norm's consequence.

3.1 Applications

The existing implementations of the **Colour & Form** system, the **Waste-collector** system and the **Forest Cleaner** system are easily adapted to serve as demonstrations of the use of norm-regulated transition system situations. However, the use of norm-regulated transition system situations is not limited to the DALMAS context. Many kinds of dynamic systems (including different types of transition systems and multi-agent systems) in which state transitions are connected to the actions of a single 'moving' agent, could be modelled and implemented by (iterated) use of a norm-regulated transition system situation. The general-level Java/Prolog implementation is intended to serve as a tool for the implementation of such systems, with a Prolog logic server as a backend and a Java user interface as frontend[5], functioning as a lookup-service that answers questions such as 'is act a permissible for x in state s, according to normative system $\mathcal{N}$' or 'which acts are permissible for x in state s, according to $\mathcal{N}$'. At the system level, it could be used to maintain a normative system for some society, in combination with some norm enforcement strategy. At the agent level, it could be used as a common normative framework that is shared by individual agents that take norms into account in their reasoning cycle, or as part of an agent's internal architecture, either to represent a model of society's normative system or to represent an agent's 'internal' normative system ('ethics'). Naturally, the use of both Java and Prolog as implementation languages has both advantages and disadvantages. The primary advantage is that this approach combines the strengths of two different programming paradigms and languages. On the other hand, it puts high demands regarding skills in both object-oriented and logic programming on the developer wishing to use the framework to develop a specific system.

4 CONCLUSION AND FUTURE WORK

This paper has introduced the notion of a transition system situation, which is intended to represent a single step in the run of many kinds of transition systems. In a norm-regulated transition system situation, the permission or prohibition of actions is related to the permission or prohibition of different types of state transitions with respect to some condition d on a number of agents $x_1,\dots,x_\nu$ in a state. The framework uses a representation of conditional norms based on the algebraic approach[6] to normative systems used in (Odelstad and Boman, 2004) and a systematic exploration of the possible types of state transitions with

[5]The source code will be made available for download and will be publicly and freely disseminated.

[6]This approach was originally developed in a series of papers; see for example (Lindahl and Odelstad, 2003; Lindahl and Odelstad, 2008; Lindahl and Odelstad, 2011; Lindahl and Odelstad, 2012).

respect to $d(x_1,\ldots,x_\nu)$. A general-level Java/Prolog framework for norm-regulated transition system situations is currently being developed, by adaption of the existing implementation of the DALMAS architecture. The set of eight transition type conditions C^a_i is an extension of the set of six E^a_i conditions in (Odelstad and Boman, 2004). These conditions were intended as an interpretation in the DALMAS context of Lindahl's set of one-agent types of normative positions. The (potential) connection between the combination of τ_i and C^a_i and the Kanger-Lindahl theory of normative positions is interesting. It has been partly investigated in (Hjelmblom, 2011), but deserves to be further explored.

Lindahl and Odelstad argue that a normative system should express "... general rules where no individual names occur. If the task is to represent a normative system this feature of generality has to be taken into account." (Lindahl and Odelstad, 2012, p. 5) One of the strengths of their algebraic approach to normative systems is in fact the expressive power it yields. The algebraic normative framework presented in this paper allows the construction of norms based on conditions on an arbitrary number of agents, in contrast to for example Dynamic deontic logic (Meyer, 1987) and Dynamic logic of permission (van der Meyden, 1996) which both have their roots in Propositional Dynamic Logic (PDL). Unlike in the agent-stranded coloured transition systems (Craven and Sergot, 2008; Sergot, 2008), the framework presented in this paper does not explicitly distinguish between state permission laws and action permission laws. It allows, however, a state permission law to be represented implicitly as a special case, by a norm which prohibits all transitions that lead to an undesired state. Our framework treats all norms as action permission laws, in the sense that actions are prohibited in different states as a consequence of certain transition types being prohibited by the normative system. It allows the creation of norms that forbid specific named actions in certain situations, by choosing a normative consequence that forbids the agent to act so that it ends up in a state where the last action performed was the prohibited action. This requires some sort of history of actions to be part of the state of the system.

The idea to base norms on permissible and prohibited types of state transitions has, to the author's knowledge, not been systematically explored before. It appears that the language for action permission laws used by Craven and Sergot also allows the formulation of norms that prohibit certain types of transitions, but an example of this feature is not given in (Craven and Sergot, 2008). In Dynamic deontic logic it is only the state resulting from a transition that determines if the transition is classified as 'permitted/non-permitted', while in Dynamic logic of permission, it is executions of actions that are classified as 'permitted/non-permitted'. van der Meyden's treatment of permission uses the process semantics for actions, in which the denotation of an action expressions is a set of sequences of states. This allows for the description of the states of affairs during the execution of an action; the permission of an action is not dependent only on the state resulting from the execution of the action, but also on the intermediate states.

The systematic treatment of the different types of transitions ensures that the set of transition type operators C^a_j and the corresponding prohibition operators P_i exhaust the space of meaningful transition type prohibitions. Therefore, norm-regulated transition system situations could be used in a given problem domain to systematically search for the 'best' normative system for (a class of) dynamic systems, according to some criteria for evaluation of the system's performance. For example, as suggested in (Hjelmblom, 2011), a genetic algorithm or some other mechanism from machine learning could be employed to seek the optimal normative system for a particular task.

The requirement that each event ε in a norm-regulated transition system situation represents an action performed by a single agent deserves further attention. This bears some resemblance to the restriction in the 'rooms' example in (Craven and Sergot, 2008, p. 178ff) that only one agent at a time can move through a doorway in this environment. This raises a number of questions regarding the relationship between norm-regulated transition system situations and transition systems in which a single transition may correspond to the simultaneous action of several agents, possibly including 'actions' by the environment itself. These issues deserve a deeper discussion, which is left for future papers.

It is possible to combine a norm-regulated transition system situation with some mechanism for norm change, but in the current formulation norms may not be changed as a consequence of an action by an agent in a state s, since the normative system $\mathcal{N}$ is not itself considered a part of s. An interesting line of future work is to explore the possibility to let the normative system be a part of the state, thereby letting agents choose actions that modify the normative system. Another interesting issue is consistency; an inconsistent normative system may lead to a situation in which the deontic structure is empty, i.e. all actions are prohibited. How the system should behave in such a situation is heavily dependent on the nature of the specific application at hand; this is not specified by the general-level framework.

ACKNOWLEDGEMENTS

The author wishes to thank Jan Odelstad and Magnus Boman for valuable ideas and suggestions.

REFERENCES

Cliffe, O., De Vos, M., and Padget, J. (2006). Specifying and analysing agent-based social institutions using answer set programming. In Boissier, O., Padget, J., Dignum, V., Lindemann, G., Matson, E., Ossowski, S., Sichman, J., and Vzquez-Salceda, J., editors, *Coordination, Organizations, Institutions, and Norms in Multi-Agent Systems*, volume 3913 of *Lecture Notes in Computer Science*, pages 99–113. Springer Berlin / Heidelberg. doi:10.1007/11775331_7.

Cliffe, O., De Vos, M., and Padget, J. (2007). Answer set programming for representing and reasoning about virtual institutions. In Inoue, K., Satoh, K., and Toni, F., editors, *Computational Logic in Multi-Agent Systems*, volume 4371 of *Lecture Notes in Computer Science*, pages 60–79. Springer Berlin / Heidelberg. doi:10.1007/978-3-540-69619-3_4.

Combettes, S., Hanachi, C., and Sibertin-Blanc, C. (2006). Organizational petri nets for protocol design and enactment. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems*, AAMAS '06, pages 1384–1386, New York, NY, USA. ACM. doi:10.1145/1160633.1160892.

Craven, R. and Sergot, M. (2008). Agent strands in the action language nC+. *Journal of Applied Logic*, 6(2):172–191. Selected papers from the 8th International Workshop on Deontic Logic in Computer Science, 8th International Workshop on Deontic Logic in Computer Science.

Gaertner, D., Clark, K., and Sergot, M. (2007). *Ballroom etiquette:* a case study for norm-governed multi-agent systems. In Noriega, P., Vzquez-Salceda, J., Boella, G., Boissier, O., Dignum, V., Fornara, N., and Matson, E., editors, *Coordination, Organizations, Institutions, and Norms in Agent Systems II*, volume 4386 of *Lecture Notes in Computer Science*, pages 212–226. Springer Berlin / Heidelberg. doi:10.1007/978-3-540-74459-7_14.

Governatori, G., Rotolo, A., and Sartor, G. (2005). Temporalised normative positions in defeasible logic. In *Proceedings of the 10th international conference on Artificial intelligence and law*, ICAIL '05, pages 25–34, New York, NY, USA. ACM. doi:10.1145/1165485.1165490.

Hjelmblom, M. (2008). Deontic action-logic multi-agent systems in Prolog. Technical Report 30, University of Gävle, Division of Computer Science.

Hjelmblom, M. (2011). State transitions and normative positions within normative systems. Technical Report 37, University of Gävle, Department of Industrial Development, IT and Land Management.

Hjelmblom, M. and Odelstad, J. (2009). jDALMAS: A Java/Prolog framework for deontic action-logic multi-agent systems. In Håkansson, A., Nguyen, N., Hartung, R., Howlett, R., and Jain, L., editors, *Agent and Multi-Agent Systems: Technologies and Applications*, volume 5559 of *Lecture Notes in Computer Science*, pages 110–119. Springer Berlin / Heidelberg. doi:10.1007/978-3-642-01665-3_12.

Laaksolahti, J. and Boman, M. (2002). Anticipatory guidance of plot. *CoRR*, cs.AI/0206041. doi:10.1007/978-3-540-45002-3_14.

Lindahl, L. (1977). *Position and change: a study in law and logic*. Synthese library. D. Reidel Pub. Co.

Lindahl, L. and Odelstad, J. (2003). Normative systems and their revision: An algebraic approach. *Artificial Intelligence and Law*, 11:81–104. doi:10.1023/B:ARTI.0000046005.10529.47.

Lindahl, L. and Odelstad, J. (2004). Normative positions within an algebraic approach to normative systems. *Journal of Applied Logic*, 2(1):63 – 91. The Sixth International Workshop on Deontic Logic in Computer Science. doi:10.1016/j.jal.2004.01.004.

Lindahl, L. and Odelstad, J. (2008). Intermediaries and intervenients in normative systems. *Journal of Applied Logic*, 6(2):229 – 250. Selected papers from the 8th International Workshop on Deontic Logic in Computer Science, 8th International Workshop on Deontic Logic in Computer Science. doi:10.1016/j.jal.2007.06.010.

Lindahl, L. and Odelstad, J. (2011). Stratification of normative systems with intermediaries. *Journal of Applied Logic*, 9(2):113 – 136. Special Issue: Selected and revised papers from the Ninth International Conference on Deontic Logic in Computer Science (DEON 2008), Ninth International Conference on Deontic Logic in Computer Science. doi:10.1016/j.jal.2010.01.002.

Lindahl, L. and Odelstad, J. (2012). *The Theory of Joining-Systems*, volume 1. College Publications, London. To appear in Gabbay; Horthy; van der Meyden; and van der Torre: Handbook of Normative systems.

Makinson, D. and van der Torre, L. (2007). What is input/output logic? input/output logic, constraints, permissions. In Boella, G., van der Torre, L., and Verhagen, H., editors, *Normative Multi-agent Systems*, number 07122 in Dagstuhl Seminar Proceedings, Dagstuhl, Germany. Internationales Begegnungs- und Forschungszentrum für Informatik (IBFI), Schloss Dagstuhl, Germany.

Meyer, J.-J. C. (1987). A different approach to deontic logic: deontic logic viewed as a variant of dynamic logic. *Notre Dame Journal of Formal Logic*, 29(1):109–136. doi:10.1305/ndjfl/1093637776.

Odelstad, J. (2008). *Many-Sorted Implicative Conceptual Systems*. PhD thesis, Royal Institute of Technology, Computer and Systems Sciences, DSV. QC 20100901.

Odelstad, J. and Boman, M. (2004). Algebras for agent norm-regulation. *Annals of Mathematics and Artificial Intelligence*, 42:141–166. doi:10.1023/B:AMAI.0000034525.49481.4a.

Raskin, J.-F., van der Torre, L. W., and Tan, Y.-H. (1996). How to model normative behavior in petri nets. In *Proceedings of the 2nd Modelage Workshop on Formal Models of Agents*, pages 223–241.

Sergot, M. (2008). Action and agency in norm-governed multi-agent systems. In Artikis, A., O'Hare, G., Stathis, K., and Vouros, G., editors, *Engineering Societies in the Agents World VIII*, volume 4995 of *Lecture Notes in Computer Science*, pages 1–54. Springer Berlin / Heidelberg. doi:10.1007/978-3-540-87654-0_1.

van der Meyden, R. (1996). The dynamic logic of permission. *Journal of Logic and Computation*, 6(3):465–479.

Vazquez-Salceda, J., Aldewereld, H., and Dignum, F. (2004). Implementing norms in multiagent systems. In Lindemann, G., Denzinger, J., Timm, I., and Unland, R., editors, *Multiagent System Technologies*, volume 3187 of *Lecture Notes in Computer Science*, pages 313–327. Springer Berlin / Heidelberg. doi:10.1007/978-3-540-30082-3_23.