Reasoning about Normative Update

Natasha Alechina
University of Nottingham
Nottingham, UK
nza@cs.nott.ac.uk

Mehdi Dastani
Utrecht University
Utrecht, The Netherlands
M.M.Dastani@uu.nl

Brian Logan
University of Nottingham
Nottingham, UK
bsl@cs.nott.ac.uk

Abstract

We consider the problem of updating a multi-agent system with a set of *conditional norms*. A norm comes into effect when its condition becomes true, and imposes either an obligation or a prohibition on an agent which remains in force until a state satisfying a deadline condition is reached. If the norm is violated, a sanction is imposed on the agent. We define a notion of a *normative update* of a multi-agent system by a set of conditional norms, and study the problem of checking whether the agent(s) can bring about a state satisfying a property without incurring a specified number of sanctions.

1 Introduction

Norms have been widely proposed as a means of coordinating and regulating the behaviours of agents within a multi-agent system (MAS). Norms can be viewed as standards of behaviour which specify that certain states or sequences of actions in a multi-agent environment are prohibited or incur some sanction. For example, the designer of an electronic marketplace may want to encourage registration before exchanging goods and timely payment for goods sold. Norms can be implemented in a MAS environment through norm enforcement or norm regimentation. Enforcement imposes sanctions on bad states or behaviours, whereas regimentation eliminates bad states and behaviours.

Norms can be implemented directly as constraints on actions (or action sequences) within a MAS environment. However, there are a number of reasons to separate the implementation of the environment as a set of (physical) behaviours from the implementation of norms that control and coordinate agents' behaviours. One reason is that normative aspects of an environment typically change more frequently than its physical aspects (e.g., possible actions). Another reason for this separation of concerns is to support the development and maintenance of environments in a modular and reusable manner, by making the development and maintenance of norms independent of the development and maintenance of the physical aspects of the environment. Thirdly, enforced (non-regimented) norms are different from the physical constraints on actions typically expressed in action preconditions, as norms can be violated. The violation of enforced norms incurs sanctions, and the type of sanction applied depends on the circumstances under which norms are violated. Lastly, a separate implementation of norms makes it easier to analyse the effect of updating the same physical environment with different sets of norms. We call the addition of a set of norms to a MAS environment a *normative update* of the environment.

A key problem in the design of normative MAS is whether a proposed normative update will have the effect intended by the designer of the system. Previous work on verifying properties of normative updates has considered only a relatively simple view of norms where some actions or states are designated as violations.

In this paper, we focus on *conditional norms* with deadlines and sanctions [Tinnemeier *et al.*, 2009], and present a formal framework that allows us to reason about the effects of a normative update of a MAS environment by a set of conditional norms. Conditional norms are triggered (detached) in certain states of the environment and have a temporal dimension specified by a deadline. The satisfaction or violation of a detached norm depends on whether the behaviour of the agent(s) brings about a specified state of the environment before a state in which the deadline condition is true. Norms can be enforced by means of sanctions or they can be regimented by disabling actions in specific states. We introduce two new logics to reason about conditional norms, $CTLS$ (CTL with Sanctions) and $ATLS$ (ATL with Sanctions), which allow us to express properties such as 'the agent(s) are unable to bring about a state satisfying ϕ' and 'the agent(s) are unable to bring about a state satisfying ϕ without incurring at least sanctions Z'. Sanction and regimentation properties are verified by means of model checking. We prove that the complexity of model-checking problems for $CTLS$ and $ATLS$ is PSPACE.

2 Normative Update

Let Π be a finite set of propositional variables. Π is partitioned into a set of physical atoms Π_b and a set of sanction atoms Π_s. The distinguished atom $san_\perp \in \Pi_s$ denotes the sanction in regimented norms (that is, norms which cannot be violated by agents).

Definition 1 (Physical Transition System) *A* physical transition system *is a tuple* $M = (S, R, V)$*, where* S *is a finite,*

non-empty set of states, $R \subseteq S \times S$ is a transition relation, and V is a propositional valuation $S \longrightarrow 2^{\Pi}$.

Intuitively, M describes the computational behaviour of a (multi) agent system. R corresponds to agents executing actions, and actions have preconditions and effects specified in terms of physical atoms that are satisfied by the transitions.

Before we can define norms and normative update, we need some temporal logic preliminaries. Given a physical transition system $M = (S, R, V)$, a *path* through M is a sequence $s_0, s_1, s_3, \ldots$ of states such that $s_i R s_{i+1}$ for $i = 0, 1, \ldots$. A *fullpath* is a maximal path and a *run* of M is a fullpath which starts from a state $s_I \in S$ designated as the initial state of M. We denote fullpaths by $\rho, \rho', \ldots$, and the state at position i on ρ by $\rho[i]$. For a state $s \in S$, the *tree* rooted at s is the infinite tree $T(s)$, obtained by unfolding M from s (the nodes of T are finite paths starting from s ordered by the prefix relation). $T(M) = T(s_I)$ is the *computation tree* of M. Note that branches of $T(M)$ are runs of M.

To make the notions of norm violation, enforcement and update precise, we use Computational Tree Logic with Past ($CTL^* + Past$), see, e.g., [Schnoebelen, 2003]. The syntax of $CTL^* + Past$ formulas is defined as follows:

$$p \in \Pi \mid \neg\phi \mid \phi \wedge \psi \mid \mathcal{X}\phi \mid \phi \mathcal{U} \psi \mid \mathcal{X}^{-1}\phi \mid \phi \mathcal{S} \psi \mid E\phi$$

where $\mathcal{X}$ means next state, $\mathcal{X}^{-1}$ means previous state, $\mathcal{U}$ stands for until, $\mathcal{S}$ for since, and E for 'exists a run'. The truth definition for formulas is given relative to $T(M)$, a run ρ and the state at position i on ρ:

- $T(M), \rho, i \models p$ iff $p \in V(\rho[i])$
- $T(M), \rho, i \models \neg\phi$ iff $T(M), \rho, i \not\models \phi$
- $T(M), \rho, i \models \phi \wedge \psi$ iff $T(M), \rho, i \models \phi$ and $T(M), \rho, i \models \psi$
- $T(M), \rho, i \models \mathcal{X}\phi$ iff $T(M), \rho, i+1 \models \phi$
- $T(M), \rho, i \models \phi \mathcal{U} \psi$ iff $\exists j \geq i$ such that $T(M), \rho, j \models \psi$ and $\forall k : i \leq k < j, T(M), \rho, k \models \phi$
- $T(M), \rho, i \models \mathcal{X}^{-1}\phi$ iff $T(M), \rho, i-1 \models \phi$
- $T(M), \rho, i \models \phi \mathcal{S} \psi$ iff $\exists j \leq i$ such that $T(M), \rho, j \models \psi$ and $\forall k : i \geq k > j, T(M), \rho, s_k \models \phi$
- $T(M), \rho, i \models E\phi$ iff for some run ρ' which is identical to ρ on the first i indices, $T(M), \rho', i \models \phi$.

$F\phi$ (some time in the future) is defined as $\top \mathcal{U} \phi$, $G\phi$ (always in the future) is defined as $\neg F \neg \phi$.

Definition 2 (Norms) *Let $cond$, ϕ, d be boolean combinations of propositional variables from Π_b and $san \in \Pi_s$. A* conditional obligation *is represented by a tuple $(cond, O(\phi), d, san)$ and a* conditional prohibition *is represented by a tuple $(cond, P(\phi), d, san)$. A norm set N is a set of conditional obligations and conditional prohibitions.*

Conditional norms are evaluated on runs of the physical transition system. A conditional norm $n = (cond, Y(\phi), s, san)$, where Y is O or P, is *detached* in a state satisfying its condition $cond$. Detached norms persist as long as they are not obeyed or violated, even if the triggering condition of the corresponding conditional norm does not hold any longer. A detached obligation $(cond, O(\phi), d, san)$ is *obeyed* if no state satisfying d is encountered before execution reaches a state satisfying ϕ, and *violated* if a state satisfying d is encountered before execution reaches a state satisfying ϕ. Conversely, a detached prohibition $(cond, P(\phi), d, san)$ is obeyed if no state satisfying ϕ is encountered before execution reaches a state satisfying d, and violated if a state satisfying ϕ is encountered before execution reaches a state satisfying d. If a detached norm is violated in a state s, the sanction corresponding to the norm is applied (becomes true) in s. We say a detached norm is annulled in a state s' immediately after a state s in which the norm is obeyed or violated, unless the same norm is detached again in s'. Note that given an original physical transition system, we cannot say whether a norm is violated in a given state s; to determine that, we need to know the path taken to reach s (e.g., whether any norms were detached in the past), and there may be more than one path to s. This is the reason why conditional norms are evaluated on *runs* of the system.

Definition 3 (Norm Violation) *A state $\rho[i]$ violates a conditional obligation $(cond, O(\phi), d, san)$ on run ρ in $T(M)$ iff*

$$T(M), \rho, i \models d \wedge \neg\phi \wedge (((\neg\phi \wedge \neg d)\, \mathcal{S}\, (cond \wedge \neg\phi \wedge \neg d)) \vee cond)$$

$\rho[i]$ violates a conditional prohibition $(cond, P(\phi), d, san)$ iff

$$T(M), \rho, i \models \phi \wedge \neg d \wedge (((\neg\phi \wedge \neg d)\, \mathcal{S}\, (cond \wedge \neg\phi \wedge \neg d)) \vee cond)$$

We distinguish two kinds of sanctions: regimentation sanctions and resource sanctions. *Regimentation sanctions* arise in the context of wholly or partially regimented normative systems, where the normative system enforces norms to ensure that certain behaviours never occur. If a norm labels a state with the distinguished sanction atom $san_{\perp}$, then the run containing this state is removed from the set of runs of the system by the normative update. *Resource sanctions*, on the other hand, arise in the context of (wholly or partially) unregimented normative systems, which treat sanctions essentially like fines or taxes. Such sanctions penalize rather than eliminate certain execution paths by reducing the resources of the agent. An undesirable state (from the point of view of the system designer) may or may not be achievable by an agent (or agents) depending on the resources the agent is able or willing to commit to achieving it.

The *normative update* of a physical transition system is defined by applying sanctions to violating states in the execution paths of the system, and removing regimented paths.

Definition 4 (Norm Enforcement) *Let N be a norm set, and $n = (cond, Y(\phi), d, san) \in N$ be a norm (for $Y \in \{O, P\}$). The norm n is enforced on a run ρ when, for every i, $\rho[i]$ violates n iff $\rho, i \models san$. The norm set N is enforced on ρ iff all norms in $n \in N$ are enforced on ρ.*

Given a physical transition system M and a set of norms N, a normative update of M with N, denoted by M^N, is realized by updating the tree $T(M)$ of M with the norm set N.

Definition 5 (Normative Update) *Let $M = (S, R, V)$ be a finite physical transition system with initial state s_I and N a finite set of conditional obligations and prohibitions. A normative update of M with N, M^N, is $T(M)$ where all norms*

from N are enforced on all runs. In other words, in each tree node s', $V^N(s')$ contains sanction atoms for all norms violated in s'. Paths which contain a state satisfying the distinguished sanction atom $san_\perp$ are removed from M^N.

We assume that M^N is non-empty, i.e., that regimentation does not remove all possible paths from the system (cf the 'reasonableness assumption' in [Ågotnes *et al.*, 2010]).

3 CTLS

We now show how to reason about the effects of updating a physical transition system with norms. To simplify the presentation, in this section we first consider the case of a single agent, and consider the multi-agent case in Section 4. We use an extension of Computational Tree Logic CTL [Clarke *et al.*, 1986] with sanction bounds which we call $CTLS$.

To express properties of a normative update, we need to be able to talk about the number of sanctions applied on a run. For simplicity, we assume that each norm i has its own distinct sanction proposition san_i, apart from regimented norms which all have $san_\perp$ as the sanction atom. To say that a path contains no occurrences of san_0, at most 2 occurrences of san_1, and at most infinitely many occurrences of san_2 we will use sanction bound expressions which are multisets of the form $\{san_1, san_1, \infty * san_2\}$. We will use abbreviations 0 for the multiset which contains 0 occurrences of any sanctions, and ∞ for the multiset where the multiplicity of each sanction is ∞. Below, we use set notation for multisets with the obvious semantics. In particular, the union of two multisets Z_1 and Z_2 is a multiset, where the multiplicity of each element is the sum of its multiplicities in Z_1 and Z_2, and $n + \infty = \infty$ for any multiplicity n. We also use $\leq$ to compare two multisets, again as an obvious generalisation of set inclusion ($Z_1 \leq Z_2$ iff the multiplicity of each element in Z_1 is less then or equal to its multiplicity in Z_2, with $n \leq \infty$ for any multiplicity n).

The language of $CTLS$ is defined by:

$$p \in \Pi \mid \neg\phi \mid \phi \wedge \phi \mid E^{\leq Z}\mathcal{X}\phi \mid E^{\leq Z}\phi\mathcal{U}\phi \mid E^{\leq Z}G\phi$$

where Z is a multiset of sanction atoms (sanction bound). $E^{\leq Z}$ means 'there exists a path of sanction cost at most Z'. We define $E^{\leq Z}F\phi =_{df} E^{\leq Z}\top\mathcal{U}\phi$. Note that CTL operators are a special case of $CTLS$ operators with an infinite bound for all sanctions.

We use the abbreviation $sanctions(s)$ for a state $s \in S$ to mean the set of sanction atoms in $V(s)$. (Recall that, in a tree, each s is on a unique run.) For a run ρ, $sanctions(\rho) = \cup_i sanctions(\rho[i])$ is a multiset of sanction atoms.

The truth of $CTLS$ formulas is defined relative to a tree model T (intuitively, a normative update) and a state in T:

- $T, s \models E^{\leq Z}\mathcal{X}\phi$ iff there a fullpath ρ' with $\rho'[0] = s$, such that $T, \rho'[1] \models \phi$ and $sanctions(\rho') \leq Z$
- $T, s \models E^{\leq Z}\phi\mathcal{U}\psi$ iff there exists a fullpath ρ' with $\rho'[0] = s$, such that for some $n \geq 0$, $T, \rho'[n] \models \psi$ and for every i, $i < n$, $T, \rho'[i] \models \phi$ and $sanctions(\rho') \leq Z$
- $T, s \models E^{\leq Z}G\phi$ iff there exists a fullpath ρ' with $\rho'[0] = s$, such that for every i, $T, \rho'[i] \models \phi$ and $sanctions(\rho') \leq Z$.

In general, two kinds of properties of a normative update are of interest: liveness properties and safety properties. *Liveness properties* check that, following a normative update, desirable states (from the point of view of the system designer) are still possible and can be achieved without an agent incurring sanctions, or with some 'admissible' number of sanctions. For example, if ϕ is a desirable state, then $E^{\leq 0}F\,\phi$ is a liveness property which says that there is an execution path which achieves ϕ and no state of this path has a sanction applied. Conversely, *safety properties* check that it is impossible for an agent to reach states where some undesirable property holds without incurring some minimum level of sanctions. For example, if ψ is an undesirable property, then the formula $\neg E^{\leq san}F\psi$ says that there is no execution path where ψ is reachable and the set of sanctions is less than san. So the agent may be able to achieve ψ, but it will have to pay more than san.

Model-Checking

The model-checking problem for a normative update of a single agent physical transition system takes as inputs a physical transition system $M = (S, R, V)$ with initial state $s_I \in S$, a finite set of conditional norms N, and a formula ϕ of $CTLS$. It returns true if $M^N, s_I \models \phi$, and false otherwise.

Theorem 1 *The model-checking problem for a normative update of a single agent physical transition system is in PSPACE.*

Proof. We give a non-deterministic algorithm for checking whether for an arbitrary s, $M^N, s \models \phi$, which requires space polynomial in $|S|$, $|N|$ and $|\phi|$. This shows that the model-checking problem is in NPSPACE. Since NPSPACE=PSPACE, it is also in PSPACE.

The algorithm is as follows. Assume we have a set of subformulas of ϕ closed under single negations, ordered in increasing order of complexity. We label states with subformulas in ϕ in order. For subformulas which do not start with a path quantifier, the algorithm is as for propositional logic (constant time), and for subformulas which start with a path quantifier with an infinite bound, the algorithm is as for CTL. If we have a subformula of the form $E^{\leq Z}\psi$, then we guess a run ρ which goes through s, and continues infinitely (note that such a run can still be represented using space polynomial in $|S|$) such that the part of ρ starting in s satisfies ψ, accumulates no more than Z sanctions, and does not violate any norms which result in the sanction $san_\perp$. Note that to check the number of sanctions incurred in the future of s, we need to know what happens on the run before s. □

Theorem 2 *The model-checking problem for a normative update of a single agent physical transition system is PSPACE-hard.*

Proof. The proof is by reduction of the QSAT (Satisfiability of Quantified Boolean Formulas) problem (which is a PSPACE-complete problem) to the model-checking problem for a normative update. The idea of the proof comes from the PSPACE-completeness proof for ATL^+ with perfect recall in [Bulling and Jamroga, 2010].

The QSAT problem takes as an input a formula of the form $\exists p_1 \forall p_2 \ldots Q p_n \phi$ where p_i are propositional variables, the

quantifier for p_i is $\exists$ if i is odd and $\forall$ if i is even, and ϕ is a propositional formula in negation normal form (negations only apply to propositional variables).

For each quantified boolean formula $\psi = \exists p_1 \forall p_2 \ldots Q p_n \phi$, we are going to construct a structure M_ψ which encodes conditions for its satisfiability: it starts with transitions corresponding to selection of truth values for the variables, continues with a parse tree of ϕ, then evaluates literals at the leaves of the parse tree and finally transits to a 'success' or 'failure' state. We are also going to introduce a set of norms N which prohibit evaluating p_i to true and false simultaneously. Then we reduce the problem of whether ψ is true to the problem of whether an update of M_ψ with N satisfies a $CTLS$ formula ψ' which says that there is a sanction-free way to traverse the structure which ends in the success state. The formula ψ' is polynomial in the size of ψ, as are M_ψ and N.

The structure M_ψ and formula ψ' use the following propositional variables: for each p_i in ψ, p_i itself and also a special variable $notp_i$ (the first one becomes true if p_i is assigned the value true, and the second if it is assigned false). We also need a propositional variable yes which holds in the success state.

M_ψ consists of three 'sections'. In the first section, for every variable p_i in ϕ, we have a state s_i from which there are two transitions to states $s_i\top$ and $s_i\bot$; in $s_i\top$ p_i is true, and in $s_i\bot$ $notp_i$ is true. From each of these states, there is a transition to s_{i+1}. From the last two states in the value selection section, $s_n\top$ and $s_n\bot$, there is a transition to the first state s_ϕ in the second (parse tree) section. From each state s_χ in this section, there are n transitions to the states corresponding to the arguments of the main connective of χ. Unlike the parse tree assumed in [Bulling and Jamroga, 2010], we are building an and-or tree: at one level there are n conjuncts, at the next level m disjuncts, etc. When we start hitting the literals, we add a dummy single transition to the next state to make sure that the and-or tree has the same depth on all branches. The last level of states in the parse tree section are states corresponding to literals p_i or $\neg p_i$: s_{p_i} or $s_{\neg p_i}$. The last section of the model is as follows. From a state s_{p_i} corresponding to a positive literal, there are two transitions to states $s'_i\top$ and $s'_i\bot$ which satisfy p_i and $notp_i$ respectively, and there is a transition from $s'_i\top$ to the s_{yes} state which satisfies yes and a transition from $s'_i\bot$ to the s_{no} state. From a state $s_{\neg p_i}$ corresponding to a negative literal, there are also two transitions to states $s'_i\top$ and $s'_i\bot$ which satisfy p_i and $notp_i$ respectively, but this time there is a transition from $s'_i\top$ to the s_{no} state and a transition from $s'_i\bot$ to the s_{yes} state. Both s_{yes} and s_{no} have transitions to themselves.

The norms prohibit going through a $s'_i\bot$ state after going through a $s_i\top$ state. This intuitively corresponds to prohibiting assigning first true and then false to p_i. Similarly they prohibit going through a $s'_i\top$ state after going through a $s_i\bot$ state. For each p_i we have two prohibitions, $(p_i, P(notp_i), yes, san_i^+)$ and $(notp_i, P(p_i), yes, san_i^-)$.

Finally, the formula ψ' is built as follows. It starts with normal CTL quantifiers (quantifiers of sanction bound infinity) which say that there exists an assignment to p_1 such that for all assignments to p_2 etc.: $E\mathcal{X}\ E\mathcal{X}\ A\mathcal{X}\ E\mathcal{X}\ldots$ (the second $E\mathcal{X}$ after each quantifier describes the transition from $s_i\top$ or $s_i\bot$ to s_{i+1} state). Then it does the same for the and-or tree section of M_ψ: for 'and' levels it uses $A\mathcal{X}$, and for 'or' levels it uses $E\mathcal{X}$. Finally it says $E^{\leq 0}\mathcal{X} yes$, namely there is a way of making ϕ true without violating the norms. □

Translation to $CTL^* + Past$

Non-deterministic algorithms are primarily of theoretical interest. Below we give a more concrete algorithm for model-checking $CTLS$ by reducing the problem to model-checking $CTL^* + Past$. It involves a straightforward translation of sanction bounds into $CTL^* + Past$ formulas which encode information about sanctions. We can then apply a model-checking algorithm for $CTL^* + Past$.

Before we state the translation, we introduce an *expanded representation* for sanction bounds. Namely, for a sanction bound Z, the expanded representation $e(Z)$ is of the form $n_1 * san_1, \ldots, n_m * san_m$ where n_i is the multiplicity of san_i and a sanction for each non-regimented norm in N occurs unless its multiplicity is ∞. In other words, if $san_i \notin Z$, then it occurs in $e(Z)$ with $n_i = 0$, and if san occurs with multiplicity ∞, we do not include it in $e(Z)$.

The translation from $CTLS$ to $CTL^* + Past$ only affects path operators with sanction bounds:

$$tr(E^{\leq Z}\psi) = E(tr(\psi) \wedge \neg\phi_{san_\bot} \wedge \neg\phi_{(n_1+1)*san_1} \wedge \ldots \wedge \neg\phi_{(n_m+1)*san_m})$$

where $e(Z) = n_1 * san_1, \ldots, n_m * san_m$. Each formula $\phi_{(n_i+i)*san_i}$ is a $CTL^* + Past$ formula which says that sanction san_i is applied $n_i + 1$ times or more; its negation therefore states that sanction san_i is applied at most n times. Its precise form depends on whether the corresponding norm is an obligation or a prohibition. $\phi_{san_\bot}$ states that some regimented norm has been violated on the run.

On any run, for there to be (at least) $n_i + 1$ violations of a conditional obligation $(cond, O(p), d, san_i)$ in the future of a state s_j, there should be either at least $n_i + 1$ violations from obligations detached in s_j or in future states $s_k, k > j$, or one of the obligations was detached in the past (in a state $s_i, i < j$) and the remaining n_i violations result from obligations detached in the future. Note that in both cases, all the violations (and hence sanctions) occur in the current state or in future states. The case in which all $n_i + 1$ norms are detached in the current or future states can be expressed as a formula $\phi^{future}_{(n_i+1)*san_i}$ where

$$\phi^{future}_{(n_i+1)*san_i} =_{df} F(cond \wedge (\neg p \mathcal{U}(d \wedge \mathcal{X}\phi^{future}_{n_i*san_i})))$$

and

$$\phi^{future}_{san_i} =_{df} F(cond \wedge (\neg p \mathcal{U} d))$$

The case in which one instance of the norm was detached in the past and violated in the future of s_j (note that there can be at most one such instance) can be expressed as a formula $\phi^{past}_{san_i}$ where

$$\phi^{past}_{san_i} =_{df} ((\neg p \wedge \neg d)\, \mathcal{S}\, cond) \wedge (\neg p \mathcal{U} d)$$

and the formula expressing that n_i norms are detached in the current and/or future states is just $\phi^{future}_{(n_i)*san_i}$. The complete

formula which says that the obligation is violated at most n_i times is then:

$$\neg(\phi^{future}_{(n_i+1)*san_i} \vee (\phi^{past}_{san_i} \wedge \phi^{future}_{n_i*san_i}))$$

Similarly for conditional prohibitions, we need to write a formula saying that $(cond, P(q), d, san_i)$ is detached and violated at least $n_i + 1$ times in the current and/or future states

$$\phi^{future}_{(n_i+1)*san_i} =_{df} F(cond \wedge (\neg d\mathcal{U}(q \wedge \neg d \wedge \mathcal{X}\phi^{future}_{n_i*san_i})))$$

where

$$\phi^{future}_{san_i} = F(cond \wedge (\neg d\mathcal{U}(q \wedge \neg d)))$$

and a formula which says that it is detached once in the past and violated in the current or a future state:

$$\phi^{past}_{san_i} =_{df} ((\neg q \wedge \neg d)\, \mathcal{S}\, cond) \wedge \neg d\mathcal{U}q$$

The complete formula which says that the prohibition violated at most n_i times has the same form as for obligations.

Finally, $\phi_{san_\perp}$ is defined as a disjunction which says that one of the regimented norms is violated at least once.

If the sanction bound is written as a multiset (i.e., $\{san_1, san_1, san_2\}$) the translation above is polynomial in the original formula and in $|N|$, and $CTL^* + Past$ model-checking is PSPACE-complete [Schnoebelen, 2003]. In practice this means that a concrete model-checking algorithm will be exponential in the formula size. For the general case, when several non-regimented norms may have the same sanction atom, the approach above will result in a translation which is in the worst case exponential in the sanction bound. This would result in a double exponential in the formula if $CTL^* + Past$ model-checking is used. However, translation into $CTL^* + Past$ is not the only way of defining a concrete model-checking procedure for normative updates. It is possible to give an algorithm for the general problem (i.e., where multiple norms can result in the same sanction atom) that is exponential in the sanction bound, but polynomial in the number of states and edges in M. The algorithm extends a standard CTL algorithm with an additional pass over M to compute a labelling that records all non-dominated sanction bounds on paths from each state. We do not include the algorithm in the paper due to space limitations.

4 ATLS

In this section we consider normative updates of multi-agent systems. We assume that a physical multi-agent system is represented as a concurrent game structure (CGS).

To express properties such as 'a group of agents C cannot maintain property ϕ unless they incur sanction san at least once' we extend the syntax of Alternating Time Temporal Logic ATL [Alur *et al.*, 2002] with sanction bounds. We call the resulting logic $ATLS$. The syntax of $ATLS$ is defined relative to a set of propositional variables Π, a finite set of agents $\mathcal{A}$, and a set of sanction atoms $\Pi_s \subseteq \Pi$ as follows:

$$p \in \Pi \mid \neg\phi \mid \phi \wedge \psi \mid \langle\langle C\rangle\rangle^{\leq Z}\mathcal{X}\phi \mid \langle\langle C\rangle\rangle^{\leq Z}G\phi \mid \langle\langle C\rangle\rangle^{\leq Z}\phi\mathcal{U}\psi$$

where $C \subseteq \mathcal{A}$ and Z is a multiset of atoms from Π_s. Intuitively, $\langle\langle C\rangle\rangle^{\leq Z}\gamma$ means 'the group of agents C has a strategy, all executions of which incur at most Z sanctions and satisfy the formula γ, whatever the other agents in $\mathcal{A} \setminus C$ do'.

Definition 6 (Concurrent Game Structure) *A Concurrent Game Structure (CGS) is a tuple $M = (S, V, a, \delta)$ which is defined relative to a set of agents $\mathcal{A} = \{1, \ldots, n\}$ and a set of propositional variables Π, where:*

- *S is a non-empty set of states*
- *$V : S \to \wp(\Pi)$ is a function which assigns each state in S a subset of propositional variables*
- *$a : S \times \mathcal{A} \to \mathbb{N}$ is a function which indicates the number of available moves (actions) for each player $i \in \mathcal{A}$ at a state $s \in S$ such that $a(s, i) \geq 1$. At each state $s \in S$, we denote the set of joint moves available for all players in $\mathcal{A}$ by $A(s)$. That is*

$$A(s) = \{1, \ldots, a(s, 1)\} \times \ldots \times \{1, \ldots, a(s, n)\}$$

- *$\delta : S \times \mathbb{N}^{|\mathcal{A}|} \to S$ is a partial function where $\delta(s, m)$ is the next state from s if the players execute the move $m \in A(s)$.*

In what follows, we assume norms apply to individual agents. The definition of a normative system therefore does not change, except that each norm now has an extra parameter representing the agent to whom the norm applies: a *conditional obligation* for agent i is represented by the tuple $(cond, O(i, \phi), d, san)$ and a *conditional prohibition* is represented by the tuple $(cond, P(i, \phi), d, san)$ where $i \in \mathcal{A}$. As before, we assume that each norm has a unique sanction atom. We do not explicitly model group norms: rather we assume that each group obligation and prohibition is decomposed into individual obligations and prohibitions for the agents in the group, and that the sanction associated with the group norm is automatically decomposed into sanctions associated with the individual norms.

Definition 7 (Move) *Given a CGS M and a state $s \in S$, a move (or joint action) for a coalition $C \subseteq \mathcal{A}$ is a tuple $\sigma_C = (\sigma_i)_{i \in C}$ such that $1 \leq \sigma_i \leq a(s, i)$.*

By $A_C(s)$ we denote the set of all moves for C at state s. Given a move $m \in A(s)$, we denote by m_C the actions executed by C, $m_C = (m_i)_{i \in C}$. The set of all possible outcomes of a move $\sigma_C \in A_C(s)$ at state s is defined as follows:

$$out(s, \sigma_C) = \{s' \in S \mid \exists m \in A(s) : m_C = \sigma_C \wedge s' = \delta(s, m)\}$$

Definition 8 (Strategy) *Given a CGS M, a strategy for a subset of players $C \subseteq \mathcal{A}$ is a mapping F_C which associates each finite path $s_I, \ldots, s$ to a move in $A_C(s)$.*

A fullpath ρ is consistent with F_C iff for all $i \geq 0$, $\rho[i+1] \in out(\rho[i], F_C(\rho[0], \ldots, \rho[i]))$. We denote by $out(s, F_C)$ the set of all such fullpaths ρ starting from s, i.e. where $\rho[0] = s$.

For each run ρ, as before we can define what it means for a norm to be enforced, and which states on a run are assigned sanction atoms. For an agent i and a path ρ, we denote the multiset of sanctions applied to i in the states on this path by $sanctions(i, \rho)$. For a coalition C, $sanctions(C, \rho) = \cup_{i \in C} sanctions(i, \rho)$.

Definition 9 (Normative Update) *Given a CGS M and a finite set of norms N, a normative update of M with N, M^N, is a tree unravelling of M where all the norms in N are enforced (sanction atoms added to the appropriate states and paths containing states with the $san_\perp$ atom are removed).*

Definition 10 (Z-Strategy) *Given a sanction bound Z, an execution path $\rho \in out(s, F_C)$ is Z-consistent with F_C in s iff $sanctions(C, \rho) \leq Z$. We denote by $out(s, F_C, Z)$ the set of all Z-consistent paths in $out(s, F_C)$ starting in s. A strategy F_C is a Z-strategy in s iff $out(s, F_C) = out(s, F_C, Z)$.*

Given a normative update of a CGS, $M^N = (S^N, V^N, a^N, \delta^N)$, the truth definition for $ATLS$ is:

- $M^N, s \models \langle\langle C\rangle\rangle^{\leq Z}\mathcal{X}\phi$ iff there exists a startegy F_C which is a Z-strategy in s such that for all $\rho \in out(s, F_C)$, $M^N, \rho[1] \models \phi$
- $M^N, s \models \langle\langle C\rangle\rangle^{\leq Z}G\phi$ iff there exists a strategy F_C which is a Z-strategy in s such that for all $\rho \in out(s, F_C)$, $M^N, \rho[i] \models \phi$ for all $i \geq 0$
- $M^N, s \models \langle\langle C\rangle\rangle^{\leq Z}\phi\mathcal{U}\psi$ iff there exists a strategy F_C Z-strategy in s such that for all $\rho \in out(s, F_C)$, there exists $i \geq 0$ such that $M^N, \rho[i] \models \psi$ and $M^N, \rho[j] \models \psi$ for all $j \in \{0, \ldots, i-1\}$

In $ATLS$, we can express liveness and safety properties of normative updates of multi-agent systems, for example (liveness) that a group of agents can enforce ϕ while incurring at most sanction san, or (safety) that a group of agents cannot enforce ϕ while incurring less than 10 times san.

Model-Checking

The model-checking problem for a normative update to a physical multi-agent system takes as inputs the physical system $M = (S, V, a, \delta)$ with initial state $s_I \in S$, a finite set of conditional norms N and a formula ϕ of $ATLS$, and returns true if $M^N, s_I \models \phi$, false otherwise.

Theorem 3 *The model-checking problem for a normative update of a multi-agent physical system is in PSPACE.*

Proof. The proof is very similar to the proof for $CTLS$. To check whether a formula of the form $\langle\langle C\rangle\rangle^{\leq Z}\phi$ is true in a state s, we guess C's Z-strategy from that state, prune the structure to contain only the moves which are consistent with this strategy, and then check whether a CTL formula which says that ϕ is true on all paths from s is true in the resulting structure. □

We provide a concrete model-checking algorithm by translating into $ATL^* + Past$. $ATL^* + Past$ is defined as for $CTL^* + Past$, but with coalition modalities instead of path quantifiers. The model-checking problem for $ATL^* + Past$ is decidable [Mogavero *et al.*, 2010]. We can express in $ATL^* + Past$ that a path contains no more that Z sanction states, where the set of norms is restricted to norms which apply to a particular set of agents in C. The translation as before only affects modalities with sanction bounds:

$$tr(\langle\langle C\rangle\rangle^{\leq Z}\psi) = \langle\langle C\rangle\rangle(tr(\psi) \wedge \neg\phi_{san_\perp} \wedge \neg\phi_{(n_1+1)*san_1} \wedge \ldots \wedge \neg\phi_{(n_m+1)*san_m})$$

where $e(Z) = n_1 * san_1, \ldots, n_m * san_m$, $\phi_{(n_i+i)*san_i}$ is a formula which says that sanction san_i is applied $n_i + 1$ or more times, and $\phi_{san_\perp}$ is a formula which says that one of the regimented norms is violated at least once.

5 Related Work

The conditional norms we consider in this paper have some similarities with those proposed in [Dignum *et al.*, 2004; Boella *et al.*, 2008]. In [Dignum *et al.*, 2004] obligations with deadlines are characterised in CTL using a special violation constant. However, in contrast to our norms, the obligations in [Dignum *et al.*, 2004] are not conditional and do not have sanctions. In [Boella *et al.*, 2008], regulative norms (without sanctions) are used to specify obligations with deadlines that are detached under specific conditions. In contrast to our work they do not evaluate norms in a temporal logic setting or consider verification.

The approach presented in this paper extends work on normative environment programming [Tinnemeier *et al.*, 2009; Dastani *et al.*, 2013], and can be used to analyze and reason about normative environment programs as specified in [Tinnemeier *et al.*, 2009] and to verify that a normative update guarantees desirable system properties. In [Dastani *et al.*, 2013], norms are represented by counts-as rules where the consequent consists of a violation atom. The norms include obligations and prohibitions, but lack conditions or deadlines. As such, they can only be used to characterize violation states (and not violation runs).

There is also a body of work on verification of normative environment programs, e.g., [Astefanoaei *et al.*, 2009; Dennis *et al.*, 2010]. In these approaches, norms are represented by counts-as rules characterizing violated states. The norms do not have conditions or deadlines, and sanctions are not modelled by multisets, so different violations of a norm cannot be distinguished.

Another strand of work related to our approach is [Ågotnes *et al.*, 2010], where the notion of a robust normative system is introduced. We believe that the form of normative update presented in [Ågotnes *et al.*, 2010] can be modelled in our approach by considering only regimented norms.

The framework presented in [Knobbout and Dastani, 2012] introduces different types of norm compliance and allows reasoning about agent behaviours under the assumption that the agents behave according to a specific norm compliance type. However they do not consider norm enforcement.

6 Conclusions

We define the notion of a normative update of a physical system with respect to a set of conditional norms with sanctions and deadlines. Our notion of a normative update is more complex than previous work on regimentation of a physical system or marking states or edges as violations. We extend CTL (for single agents) and ATL (for multiple agents) with sanction bounds on paths to reason about the properties of such normative updates. In particular, we can say how many sanctions the agents must incur to achieve some property. We show how to reduce the problem of verifying properties of a normative update to existing model-checking problems, and characterize its computational complexity. In future work, we plan to extend our approach to allow norms to be assigned to coalitions of agents and, moreover, redefine the enforcement of norms to allow distribution of sanctions among members of a coalition when a norm is violated by the coalition.

References

[Ågotnes *et al.*, 2010] Thomas Ågotnes, Wiebe van der Hoek, and Michael Wooldridge. Robust normative systems and a logic of norm compliance. *Logic Journal of the IGPL*, 18(1):4–30, 2010.

[Alur *et al.*, 2002] Rajeev Alur, Thomas Henzinger, and Orna Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[Astefanoaei *et al.*, 2009] L. Astefanoaei, M. Dastani, J.J. Meyer, and F. de Boer. On the semantics and verification of normative multi-agent systems. *International Journal of Universal Computer Science*, 15(13):2629–2652, 2009.

[Boella *et al.*, 2008] G. Boella, J. Broersen, and L. van der Torre. Reasoning about constitutive norms, counts-as conditionals, institutions, deadlines and violations. In *Proceedings of the International Conference on Principles and Practice of Multi-Agent Systems (PRIMA)*, pages 86–97, 2008.

[Bulling and Jamroga, 2010] Nils Bulling and Wojciech Jamroga. Verifying agents with memory is harder than it seemed. In Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen, editors, *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, pages 699–706. IFAAMAS, 2010.

[Clarke *et al.*, 1986] E. M. Clarke, E. A. Emerson, and A. P. Sistla. Automatic verification of finite-state concurrent systems using temporal logic specifications. *ACM Transactions on Programming Languages and Systems*, 8(2):244–263, 1986.

[Dastani *et al.*, 2013] Mehdi Dastani, Davide Grossi, and John-Jules Meyer. A logic for normative multi-agent programs. *Journal of Logic and Computation, special issue on Normative Multiagent Systems*, 23(2):335–354, 2013.

[Dennis *et al.*, 2010] Louise A. Dennis, Nick A. M. Tinnemeier, and John-Jules Ch. Meyer. Model checking normative agent organisations. In Jürgen Dix, Michael Fisher, and Peter Novák, editors, *Computational Logic in Multi-Agent Systems - 10th International Workshop, CLIMA X, Hamburg, Germany, September 9-10, 2009, Revised Selected and Invited Papers*, volume 6214 of *Lecture Notes in Computer Science*, pages 64–82. Springer, 2010.

[Dignum *et al.*, 2004] F. Dignum, J. Broersen, V. Dignum, and J.-J. C. Meyer. Meeting the deadline: Why, when and how. In *Proceedings of the International Workshop on Formal Approaches to Agent-Based Systems (FAABS)*, pages 30–40, 2004.

[Knobbout and Dastani, 2012] Max Knobbout and Mehdi Dastani. Reasoning under compliance assumptions in normative multiagent systems. In Wiebe van der Hoek, Lin Padgham, Vincent Conitzer, and Michael Winikoff, editors, *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012)*, pages 331–340. IFAAMAS, 2012.

[Mogavero *et al.*, 2010] Fabio Mogavero, Aniello Murano, and Moshe Y. Vardi. Relentful strategic reasoning in alternating-time temporal logic. In Edmund M. Clarke and Andrei Voronkov, editors, *Logic for Programming, Artificial Intelligence, and Reasoning - 16th International Conference, LPAR-16, Dakar, Senegal, April 25-May 1, 2010, Revised Selected Papers*, volume 6355 of *Lecture Notes in Computer Science*, pages 371–386. Springer, 2010.

[Schnoebelen, 2003] Ph. Schnoebelen. The complexity of temporal logic model checking. In Philippe Balbiani, Nobu-Yuki Suzuki, Frank Wolter, and Michael Zakharyaschev, editors, *Advances in Modal Logic 4*, pages 393–436. King's College Publications, 2003.

[Tinnemeier *et al.*, 2009] Nick Tinnemeier, Mehdi Dastani, John-Jules Meyer, and Leon van der Torre. Programming normative artifacts with declarative obligations and prohibitions. In *Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology (IAT'09)*, pages 69–78, 2009.