Reasoning under Compliance Assumptions in Normative Multiagent Systems

Max Knobbout
Utrecht University
Dept. of Computer Science
The Netherlands
M.Knobbout@students.uu.nl

Mehdi Dastani
Utrecht University
Dept. of Computer Science
The Netherlands
M.M.Dastani@uu.nl

ABSTRACT

The use of norms in multiagent systems has proven to be a successful approach in order to coordinate and regulate the behaviour of participating agents. In such normative systems it is generally assumed that agents can obey or disobey norms. In this paper, we develop a logical framework for normative systems that allows reasoning about agents' abilities under a multitude of norm compliance assumptions. In particular, we investigate different types of norm compliance and propose an extension of Alternating Temporal Logic (ATL) to reason about the abilities of (coalitions of) agents under different types of norm compliance assumptions. For this extension we show that the problem of model-checking remains close to the domain of standard ATL. Finally, we show that some norms can limit an agent's autonomy in the sense that an agent cannot control the violation of these norms. We present and discuss various classes of the so-called self-supporting norms, i.e., norms for which individual agents have control over their violations.

Categories and Subject Descriptors

I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Multiagent system*

General Terms

Theory, Design, Verification

Keywords

Normative Systems, Organizations, Verification, Logic

1. INTRODUCTION

The use of norms and social laws in multiagent systems has proven to be a successful approach in order to ensure the overall objectives of such systems. Early examples of such an approach can be found in Shoham and Tennenholtz [7] and Moses and Tennenholtz [6]. In these works, social laws are used to constrain the behaviour of the agents by forbidding certain actions in specific situations. This line of research was later extended by several authors in a multitude of ways. For example, in [8] the basic idea is extended to the more expressive modelling domain of alternating temporal logic (ATL) with the idea that the application of a social law in a multiagent system is successful if, by implementing it, the overall objective of the system is satisfied. Later these ideas were extended with the notion of preference to reason about norm compliance in normative systems. Example of such extensions are [9], where each agent was attributed a single goal, and [2], where agents were given a priority list of goals. The main topic of these papers was to investigate whether certain sets of norms can be considered 'stable' under consideration of the agents' preferences. Another extension of these ideas was introduced in [4], where norms are considered from a mechanism design perspective. In this framework, norms are related to game theoretic solution concepts such that one can specify and verify whether a set of norms in a multiagent system implements some social/overall objectives in specific equilibria.

In this paper, we introduce an extension of ATL to reason about properties of normative multiagent systems under various norm compliance assumptions. In our setting, norms are assumed to be directed to coalitions of agents, e.g., a norm states that a set of agents should not take certain actions in specific states. Moreover, we assume that agents can disobey norms in the sense that they can take actions that are disallowed by the norms. We then define various types of norm compliance behaviors such as "a coalition of agents obey/disobey norms that are directed to precisely this coalition" or "a coalition of agents obey/disobey norms that are directed to this coalition and all its subcoalitions". Our proposed extension of ATL can be used to specify and verify whether some overall objective of multi-agent systems can be satisfied under the assumption that a coalition of agents behave according to a specific norm compliance type while the agents outside the coalition behave according to another norm compliance type. The proposed extension can be used by the designers of normative systems to analyze norms that are directed to coalitions of agents and to investigate their impacts on the overall system behaviour under the assumption of specific norm compliance types. This extension can also be used by individual agents who need to reason and decide if they can achieve their own objectives by behaving according to a specific norm compliance type in a normative system when the system is populated by other agents that behave according to another specific norm compliance type.

In our framework, we introduce abstract normative constraints as a basic extension of the social law paradigm. We consider norms as similar to social laws in the sense that

Appears in: *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2012),* Conitzer, Winikoff, Padgham, and van der Hoek (eds.), 4-8 June 2012, Valencia, Spain.

they denote the disallowed actions of the agents. However, norms are considered as different from social laws in the sense that the agents are allowed to violate them. Moreover, and in contrast to social laws, norms are directed to coalitions of agents and not only to individual agents. Thus, we do not assume that "implementing" a system of norms enforces every agent to be perfectly norm obedient. To some extent, our research is related to [1] in which the need for robust normative systems is discussed. They introduce an extension of CTL (Computation Tree Logic) in which statements such as "if coalition C is norm compliant, then this is sufficient to guarantee φ" can be expressed. In their work a robust normative system is defined as a multiagent system which remains 'effective' (specified by some criterium) even if certain agents behave in a non-compliant manner. In our framework, a more expressive language is proposed to also reason about system properties under various norm compliance assumptions. An important difference with this paper is that we depart from the domain of 'agent-labelled Kripke Structures' to the more expressive and natural domain of Concurrent Game Structures. Using our proposed logic, we can then construct such statements as "does there exist a norm compliant strategy for a given agent to guarantee φ under the assumption that the other agents are non-compliant?". We believe that this extra layer of expressivity is indeed important for agents in order to decide whether to comply with the given norms or participate in the multiagent system. Our framework can also be useful from the perspective of a designer when trying to design systems which formally abide certain properties under certain norm compliance assumptions.

In section 2 we will give a quick recap on the syntax and semantics of standard ATL. In section 3 we introduce the notion of *abstract normative constraints* and in section 4 we provide the syntax and semantics of our proposed ATL extension with an elaborate example. In section 5 we discuss the model checking complexity of the proposed ATL extension. Finally, in section 6 we present the notion of self-supporting norm sets.

2. CONCURRENT GAME STRUCTURES

In this section, we first briefly recall on the definition of Concurrent Game Structures, the underlying model we use for multiagent systems. A Concurrent Game Structure, or CGS, can be seen as a multiagent extension of a simple transition system. It consists of states of the world, and a complete labelling of joint-actions over the transitions connecting these states. More formally, a CGS is a tuple $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$ such that:

- A natural numbers $k \geq 1$ of players. In our model, each player corresponds to a number. We sometimes use Σ to talk about the set $\{1, ..., k\}$.
- A finite set Q of states.
- A finite set Π of atomic propositions.
- A mapping π which maps each state $q \in Q$ to a subset of propositions which are true at q. Thus for each $q \in Q$ we have $\pi(q) \subseteq \Pi$.
- A mapping Ac which maps each player $a \in \Sigma$ and each state $q \in Q$ to a non-empty subset of the natural numbers denoting the moves for player a in state q. Thus for each $a \in \Sigma$ and each $q \in Q$ it holds that $Ac(a, q) \in \mathcal{P}(\mathbb{N})$. In our model, each action corresponds to a natural number. For each state $q \in Q$, a move vector is a tuple $\langle \alpha_1, ..., \alpha_k \rangle$ such that $\alpha_i \in Ac(i, q)$. The set of all move vectors for a state $q \in Q$, denoted by $D(q)$, is given by $Ac(1, q) \times ... \times Ac(k, q)$.
- A mapping δ which maps each state $q \in Q$ and each move vector $\langle \alpha_1, ..., \alpha_k \rangle \in D(q)$ to another state that results from state q if each player adopted the move denoted in the move vector. Thus for each $q \in Q$ and each $\langle \alpha_1, ..., \alpha_k \rangle \in D(q)$ we have $\delta(q, \langle \alpha_1, ..., \alpha_k \rangle) \in Q$.

Note that this model is synchronous, meaning that at any moment in time each agent needs to decide on an action synchronously. Moreover, it is also deterministic; the same action in the same state will always yield the same resulting state.

Alternating-time temporal logic, as discussed in [3], is interpreted over a concurrent game structure S that has the same propositions and players. Evaluating a propositional formula at a given state amounts to verifying whether the formula is satisfied given the labelling of that state. To evaluate a formula of the form $\langle\langle A \rangle\rangle \psi$ at a state q of S, we can consider a game between a protagonist and an antagonist which results in a computation. At every round the protagonist chooses for each player in A a move, and then the antagonist proceeds by choosing for every player $\Sigma \backslash A$ a move, after which the position is updated from q to q'. This process is repeated indefinitely, which results in a computation λ. The protagonist wins the game if the resulting computation satisfies the subformula ψ, which is a temporal formula of the form $\bigcirc\varphi$, $\Box\varphi$ or $\varphi_1 \mathcal{U} \varphi_2$ (where φ,φ_1,φ_2 are again ATL formula's), otherwise the antagonist wins. Then the formula $\langle\langle A \rangle\rangle \psi$ is satisfied at q if the protagonist has a winning strategy for this game.

More formally, ATL is characterized by the following grammar, where $p \in \Pi$ and $A \subseteq \Sigma$: $\varphi ::= p | \neg\varphi | \varphi \wedge \varphi | \langle\langle A \rangle\rangle \bigcirc \varphi | \langle\langle A \rangle\rangle \Box \varphi \,|\, \langle\langle A \rangle\rangle \varphi \mathcal{U} \varphi$. In order to define the semantics, we first have to define the notion of strategy. A strategy for a player $a \in \Sigma$ is a mapping s_a which maps a finite (non-empty) sequence of states to an action belonging to the last state of this sequence. Thus for each sequence $q_0, ..., q_k \in Q^+$ we have $s_a(q_0, ..., q_k) \in Ac(a, q_k)$. Given a set of players $A \subseteq \Sigma$ and a state $q \in Q$, let $S_A = \{s_a \,|\, a \in A\}$ be the set of strategies A adopt and let $out(q, S_A)$ be the set of computations starting from state q which the players in A can enforce by following their respective strategies (that is, independent of what the players $\Sigma \backslash A$ play). A computation $\lambda = q_0, q_1, q_2, ...$ is in $out(q_0, S_A)$ if it holds that for all positions $i \geq 0$ there is a move vector $\langle \alpha_1, ..., \alpha_k \rangle \in D(\lambda[i])$ (where $\lambda[i]$ denotes the state at position i) such that $\delta(\lambda[i], \langle \alpha_1, ..., \alpha_k \rangle) = \lambda[i+1]$ and for all $a \in A$ it is the case that $s_a(\lambda[0, i]) = \alpha_a$.

Given a game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$ and a state $q \in Q$, we define the semantics inductively as follows:

- $S, q \vDash p$ for any proposition $p \in \Pi$ iff $p \in \pi(q)$.
- $S, q \vDash \neg\varphi$ iff $S, q \nvDash \varphi$.
- $S, q \vDash \varphi_1 \wedge \varphi_2$ iff $S, q \vDash \varphi_1$ and $S, q \vDash \varphi_2$.
- $S, q \vDash \langle\langle A \rangle\rangle \bigcirc \varphi$ iff there exists a strategy set S_A for A such that for every computation $\lambda \in out(q, S_A)$ it holds that $S, \lambda[1] \vDash \varphi$.

- $S, q \vDash \langle\langle A \rangle\rangle \Box \varphi$ iff there exists a strategy set S_A for A such that for every computation $\lambda \in out(q, S_A)$ and all positions $i \geq 0$ it holds that $S, \lambda[i] \vDash \varphi$.

- $S, q \vDash \langle\langle A \rangle\rangle \varphi_1 \mathcal{U} \varphi_2$ iff there exists a strategy set S_A for A such that for every computation $\lambda \in out(q, S_A)$ there exists a position $i \geq 0$ such that for all positions $0 \leq j < i$ it holds that $S, \lambda[j] \vDash \varphi_1$ and $S, \lambda[i] \vDash \varphi_2$.

A formula of the form $\langle\langle A \rangle\rangle \psi$ should intuitively be read as "Coalition A has a strategy in order to enforce ψ", where ψ can be a temporal formula of the form $\bigcirc \varphi$, to be read as "in the next state φ", $\Box \varphi$, to be read as "always in the future φ" and $\varphi_1 \mathcal{U} \varphi_2$, to be read as "$\varphi_1$ untill φ_2 starts to hold".

3. ABSTRACT NORMATIVE CONSTRAINTS

Before we discuss the notions of compliance and non-compliance in normative systems, it is important to clarify what we understand under a normative multiagent system. For our purposes, much in line with previous research seen in e.g. [2], a normative system is simply a set of constraints on the behaviour of the agents. However, since we have entered the domain of concurrent game structures, we extend this notion to not only account for behaviour of agents, but also behaviour of coalitions. Moreover, we also allow the agents to violate these constraints; they are not hard-constraints on the behaviour of the agents. More precisely, given a game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$, an abstract normative constraint $\langle A, \gamma \rangle$ is a tuple consisting of a subset of player $A \subseteq \Sigma$ and a mapping γ which maps a player $a \in A$ and state $q \in Q$ to a set of actions for each player that can be taken in that state. Thus given a state $q \in Q$ and a set of player $A \subseteq \Sigma$, for all $a \in A$ we have that $\gamma(a, q) \subseteq Ac(a, q)$. The set $\gamma(a, q)$ denotes all the actions that are normatively demotivated in state q for agent a. Given a set of abstract normative constraints Γ, we define $\Gamma_A = \{\langle X, \gamma \rangle \in \Gamma \mid X = A\}$ and $\Gamma_A^- = \{\langle X, \gamma \rangle \in \Gamma \mid X \subseteq A\}$. In words, Γ_A is the set of abstract normative constraints only applicable to exactly A and Γ_A^- the set of abstract normative constraints only applicable to A or any sub-coalition of A. Given a computation $\lambda = q_0, q_1, q_2, ...$ and a set of abstract normative constraints Γ, we say that $\langle A, \gamma \rangle \in \Gamma$ is enabled for A at position $i+1 \geq 0$ if $\forall a \in A$ it holds that $\gamma(a, \lambda[i]) \neq \varnothing$ and taken at position $i + 1 \geq 0$ if there is a move vector $\langle \alpha_1, ..., \alpha_k \rangle \in D(\lambda[i])$ such that $\delta(\lambda[i], \langle \alpha_1, ..., \alpha_k \rangle) = \lambda[i + 1]$ and $\forall a \in A$ it holds that $\alpha_a \in \gamma(a, \lambda[i])$. Notice that by this interpretation, an abstract normative constraint can still be taken even though the actual action an agent performed along a computation differs from the one prescribed by γ.

Given a set of agents A, we can give different interpretations towards obedience with respect to Γ. Below we define 7 different norm compliance types: 3 types of obediences, 3 types of disobediences, and 1 type of neglectfulness. However, we stress that this primarily is a choice; certainly more types of obediences can be considered and constructed based on the logical framework we provide.

1. **Coalitional obedience** A computation λ is *coalitional obedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, c\text{-}ob \rangle$-*obedient*, if at any point in the computation for every $\langle A', \gamma \rangle \in \Gamma_A^-$ it is the case that if $\langle A', \gamma \rangle$ is enabled in λ, then $\langle A', \gamma \rangle$ is not taken.
2. **Total/Selective individual obedience** A computation λ is *total individual obedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, t\text{-}ob \rangle$-*obedient*, if at any point in the computation for all $a \in A$ it holds that for every $\langle \{a\}, \gamma \rangle \in \Gamma_{\{a\}}$ it is the case that if $\langle \{a\}, \gamma \rangle$ is enabled, then $\langle \{a\}, \gamma \rangle$ is not taken. A computation λ is *selective individually obedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, s\text{-}ob \rangle$-*obedient*, if at any point in the computation there exists an $a \in A$ such that it holds that for every $\langle \{a\}, \gamma \rangle \in \Gamma_{\{a\}}$ it is the case that if $\langle \{a\}, \gamma \rangle$ is enabled, then $\langle \{a\}, \gamma \rangle$ is not taken.
3. **Neglectful obedience** A computation λ with respect to a set of agents A and an abstract normative constraint set Γ is always neglectful obedient, or $\langle \Gamma, A, \top \rangle$-*obedient*. In other words, every computation is by definition neglectful obedient.
4. **Selective/Total individual disobedience** A computation λ is *selective individual disobedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, s\text{-}dob \rangle$-*obedient*, if at any point in the computation there exists an $a \in A$ such that it holds that there exists $\langle \{a\}, \gamma \rangle \in \Gamma_{\{a\}}$ such that if $\langle \{a\}, \gamma \rangle$ is enabled, then $\langle \{a\}, \gamma \rangle$ is taken. A computation λ is *total individual disobedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, t\text{-}dob \rangle$-*obedient*, if at any point in the computation for all $a \in A$ it holds that there exists $\langle \{a\}, \gamma \rangle \in \Gamma_{\{a\}}$ such that if $\langle \{a\}, \gamma \rangle$ is enabled, then $\langle \{a\}, \gamma \rangle$ is taken.
5. **Coalitional disobedience** A computation λ is *coalitional disobedient* with respect to a set of agents A and an abstract normative constraint set Γ, or $\langle \Gamma, A, c\text{-}dob \rangle$-*obedient*, if at any point in the computation for every $\langle A', \gamma \rangle \in \Gamma_A^-$ it is the case that if $\langle A', \gamma \rangle$ is enabled in λ, then $\langle A', \gamma \rangle$ is taken.

From these definitions, we see that the following holds for obediences: Given that a computation is $\langle \Gamma, A, c\text{-}ob \rangle$-obedient, it is also $\langle \Gamma, A, t\text{-}ob \rangle$-obedient. Given that it is $\langle \Gamma, A, t\text{-}ob \rangle$-obedient, it is also $\langle \Gamma, A, s\text{-}ob \rangle$-obedient and given that it is $\langle \Gamma, A, s\text{-}ob \rangle$-obedient, it is also $\langle \Gamma, A, \top \rangle$-obedient. For the disobediences a similar result holds: $\langle \Gamma, A, c\text{-}dob \rangle$-obedience implies $\langle \Gamma, A, t\text{-}dob \rangle$-obedience, $\langle \Gamma, A, t\text{-}dob \rangle$-obedience implies $\langle \Gamma, A, s\text{-}dob \rangle$-obedience and $\langle \Gamma, A, s\text{-}dob \rangle$-obedience implies $\langle \Gamma, A, \top \rangle$-obedience. An easy way of verifying this is by considering that each step in these implications allows for more possible state-transitions to take place in a computation.

4. an-ATL SEMANTICS

In this section we will give the semantics of our new logic an-ATL; ATL extended in order to reason under the presence of these abstract norms. We will first define the notion of obedience to the level of strategies. We define the set Ω as the set of obedience types denoted by a single literal, thus we have that $\Omega = \{c\text{-}ob, t\text{-}ob, s\text{-}ob, \top, s\text{-}dob, t\text{-}dob\}$. Given a game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$, an abstract normative constraint set Γ, a state q and an obedience type $\omega \in \Omega$, we say that the strategy set S_A for players A is $\langle \Gamma, A, \omega \rangle$-obedient if it holds that for every computation $\lambda \in out(q, S_A)$, λ is $\langle \Gamma, A, \omega \rangle$-obedient. In our semantics, an obedience assump-

tion is of the form $\langle \omega, \omega' \rangle$, where $\omega, \omega' \in \Omega$. The satisfaction relation $S, \Gamma, q \vDash \phi$ in our new semantics replaces the following cases in ATL semantics:

- $S, \Gamma, q \vDash \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \bigcirc \varphi$ iff there exists a $\langle \Gamma, A, \omega \rangle$-obedient strategy set S_A for A such that for every $\langle \Gamma, \Sigma \backslash A, \omega' \rangle$-obedient computation $\lambda \in out(q, S_A)$ it holds that $S, \Gamma, \lambda[1] \vDash \varphi$.
- $S, \Gamma, q \vDash \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \Box \varphi$ iff there exists a $\langle \Gamma, A, \omega \rangle$-obedient strategy set S_A for A such that for every $\langle \Gamma, \Sigma \backslash A, \omega' \rangle$-obedient computation $\lambda \in out(q, S_A)$ and all positions $i \geq 0$ it holds that $S, \Gamma, \lambda[i] \vDash \varphi$.
- $S, \Gamma, q \vDash \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \varphi_1 \mathcal{U} \varphi_2$ iff there exists a $\langle \Gamma, A, \omega \rangle$-obedient strategy set S_A for A such that for every $\langle \Gamma, \Sigma \backslash A, \omega' \rangle$-obedient computation $\lambda \in out(q, S_A)$ there exists a position $i \geq 0$ such that for all positions $0 \leq j < i$ it holds that $S, \Gamma, \lambda[j] \vDash \varphi_1$ and $S, \Gamma, \lambda[i] \vDash \varphi_2$.

The formula $\langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \varphi$ should intuitively be read as "coalition A has an ω-obedient strategy in order to enforce φ if the remaining agents $\Sigma \backslash A$ played in accordance with an ω'-obedient computation". Moreover, we write $[\![A_{\langle \omega,\omega' \rangle}]\!] \bigcirc \varphi$ for $\neg \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \bigcirc \neg \varphi$ and $[\![A_{\langle \omega,\omega' \rangle}]\!] \Box \varphi$ for $\neg \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \Diamond \neg \varphi$ (where $\Diamond \varphi \equiv \top \mathcal{U} \varphi$; similar abbreviations can be defined for the dual of the $\mathcal{U}$ operator). The formula $[\![A_{\langle \omega,\omega' \rangle}]\!] \varphi$ should be read as "coalition A does not have a ω-obedient strategy in order to avoid φ if the remaining agents played in accordance with an ω'-obedient computation". Slightly similar to the result found in ATL, we have the following validity (notice the reversal of the obedience assumption tuple):

$$\vDash \langle\!\langle A_{\langle \omega,\omega' \rangle} \rangle\!\rangle \varphi \rightarrow [\![(\Sigma \backslash A)_{\langle \omega',\omega \rangle}]\!] \varphi$$

Moreover, the following validities hold, where β can either be replaced with "ob" or "dob":

1. $\vDash \langle\!\langle A_{\langle c\text{-}\beta,\omega \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle t\text{-}\beta,\omega \rangle} \rangle\!\rangle \varphi$
2. $\vDash \langle\!\langle A_{\langle t\text{-}\beta,\omega \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle s\text{-}\beta,\omega \rangle} \rangle\!\rangle \varphi$
3. $\vDash \langle\!\langle A_{\langle s\text{-}\beta,\omega \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle \top,\omega \rangle} \rangle\!\rangle \varphi$

Intuitively, what these formula's state is that a coalition, when having an obedient strategy to guarantee φ (under a certain assumption that the remaining agents play in accordance with an ω'-obedient computation), they also have a "less restrictive" obedient strategy (here, with "less restrictive" we mean that they are allowed to violate more normative constraints) to guarantee the same (and vice versa for disobedient strategies). This is as expected, since we have a richer pool of strategies to choose from when we have less restriction to cope with. Conversely, when reasoning about the other players, the implication works the other way around. In this case, we have the following validities (where again β can either be replaced with "ob" or "dob"):

1. $\vDash \langle\!\langle A_{\langle \omega,\top \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle \omega,s\text{-}\beta \rangle} \rangle\!\rangle \varphi$
2. $\vDash \langle\!\langle A_{\langle \omega,s\text{-}\beta \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle \omega,t\text{-}\beta \rangle} \rangle\!\rangle \varphi$
3. $\vDash \langle\!\langle A_{\langle \omega,t\text{-}\beta \rangle} \rangle\!\rangle \varphi \rightarrow \langle\!\langle A_{\langle \omega,c\text{-}\beta \rangle} \rangle\!\rangle \varphi$

This intuitively means that a coalition, when having a certain strategy to guarantee φ under a certain assumption that the other agents play in accordance with a certain obedient computation, they also have a strategy to guarantee the same under a "more restrictive" obedience assumption of the computation they move along to (where "more restrictive" means that there are less normative constraints which can be violated). This is again as expected, since we have to take into account less computations; namely we do not have to consider the computations where a normative constraint was enabled and taken which was previously not considered.

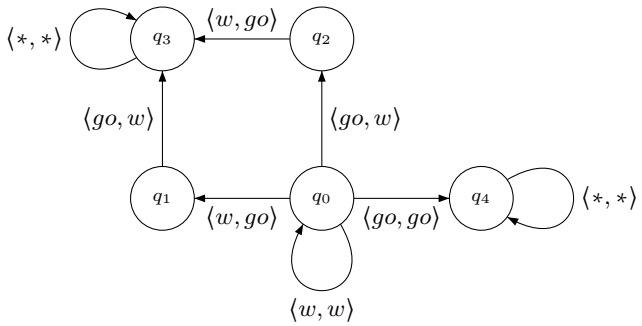

Figure 1: CGS train example.

4.1 Example

In this example we consider a scenario where there are two trains, each controlled by one agent, at the opposite ends of a tunnel. The tunnel only has room for one train, and the agents initially have 2 actions available: 'wait' and 'go'. If the agents decide to go simultaneously, the trains will crash, if they wait simultaneously nothing will happen and if one goes and the other waits, they can both successfully make it to the end of the tunnel without crashing. The CGS belonging to this scenario is shown in Figure 1. Let us denote the first agent as a_1 and the second agent as a_2. Moreover, we assume we have only two atomic propositions, 'crash' and 'no_crash', the former which is only true in q_4 and the latter only in q_3. Right of the bat, we can conclude that the following holds for every possible Γ:

$$M, \Gamma, q_0 \nvDash \langle\!\langle a_{1\langle \top,\top \rangle} \rangle\!\rangle \Diamond \text{no\_crash}$$

and

$$M, \Gamma, q_0 \nvDash \langle\!\langle a_{2\langle \top,\top \rangle} \rangle\!\rangle \Diamond \text{no\_crash}$$

In words, no agent individually has the ability to eventually bring about the fact that the trains make it through the tunnel without crashing. Moreover, the agents are collectively able to make the trains crash, displayed by the following formula (again for every possible Γ):

$$M, \Gamma, q_0 \vDash \langle\!\langle \{a_1, a_2\}_{\langle \top,\top \rangle} \rangle\!\rangle \Diamond \text{crash}$$

We will now construct a normative constraint set Γ in such a way that it brings about the following:

1. It is normatively demotivated for both agents that they enter the tunnel simultaneously.
2. It is normatively demotivated for the second agent, a_2, that he waits when the other train has not gone through the tunnel yet.

As it will turn out, the second normative constraint is sufficient for agent 1 to conclude that he has the ability to bring about the fact they will eventually reach the end of the tunnel safely under certain obedience assumptions of the other agent. However, from the perspective of agent 2, this constraint is still not sufficient: even if he adopts an obedient strategy, it is still not guaranteed that the trains will not

crash.

We can formalize the above mentioned constraints as abstract normative constraints as follows. We construct $\Gamma = \{\langle\{a_1, a_2\}, \gamma\rangle, \langle\{a_2\}, \gamma'\rangle\}$, such that:

- $\gamma(a_1, q_0) = \{\text{go}\}$, $\gamma(a_2, q_0) = \{\text{go}\}$; and
- $\gamma'(a_2, q_0) = \{\text{w}\}$

In this example, we are first going to reason about the abilities of agent one, a_1. The following an-ATL formula is valid:

$$M, \Gamma, q_0 \vDash \langle\!\langle a_{1\langle\top, t\text{-}ob\rangle}\rangle\!\rangle \Diamond \text{no\_crash}$$

In words, our first agent now has a strategy to eventually bring about the fact that the train makes it through the tunnel without crashing under the assumption that a_2 plays according to an obedient computation (in this case, he only has to take into account the normative constraint associated with γ'). To see this, observe that we do not consider any computations with transitions from q_0 to q_0 any more (and from q_0 to q_2, but this is beyond the point), thus disallowing the scenario where both agents wait for each other. The strategy for agent 1 is then to first wait (since he knows that agent 2 will not wait), and then go. Interestingly enough, agent 1 also has a strategy to eventually safely reach the end of the tunnel under the assumption that the other agent plays in accordance with disobedient computation:

$$M, \Gamma, q_0 \vDash \langle\!\langle a_{1\langle\top, t\text{-}dob\rangle}\rangle\!\rangle \Diamond \text{no\_crash}$$

To easily see this, observe that we do not consider any computations with transitions from q_0 to q_4 (and from q_0 to q_1, but again is beyond the point), thus the strategy where agent 1 immediately picks 'go' will ensure that both agents will eventually make it to the end of the tunnel safely.

Let us now reason about the abilities of agent a_2. We can see that a_2, as opposed to a_1, can not obediently bring about that both agents will reach the end of the tunnel safely:

$$M, \Gamma, q_0 \nvDash \langle\!\langle a_{2\langle t\text{-}ob, \top\rangle}\rangle\!\rangle \Diamond \text{no\_crash}$$

To see this, selecting the action 'go' in q_0 will not guarantee that we will not end up in q_4. However, interestingly enough, agent 2 does have the ability to disobediently bring about the fact that the ability of agent 1 to reach the end of the tunnel safely (under assumption that agent 2 plays in such a way that his normative constraints are not violated) is not lost:

$$M, \Gamma, q_0 \vDash \langle\!\langle a_{2\langle t\text{-}dob, \top\rangle}\rangle\!\rangle \bigcirc \langle\!\langle a_{1\langle\top, t\text{-}ob\rangle}\rangle\!\rangle \Diamond \text{no\_crash}$$

To see this, observe that the action 'wait' for agent 2 in q_0 will result in either q_0 or q_2, both in which the ability of agent 1 to safely reach the end of the tunnel, under assumption that agent 2 obeys his normative constraints, is retained. However, in case agent 2 plays an obedient strategy, this is not guaranteed, as seen by the following validity:

$$M, \Gamma, q_0 \nvDash \langle\!\langle a_{2\langle t\text{-}ob, \top\rangle}\rangle\!\rangle \bigcirc \langle\!\langle a_{1\langle\top, t\text{-}ob\rangle}\rangle\!\rangle \Diamond \text{no\_crash}$$

The reason for this is that agent 2 can not guarantee with an obedient strategy (action 'go' in q_0) that he will not end up in q_4. We could say that agent 2 is faced with a dilemma: he can play obedient, possibly wasting the ability of the other agent to reach the end of the tunnel safely if the other player assumed obedience over the normative constraints of agent 2, or play disobedient, preserving the former mentioned ability but possibly ending up in the same state again.

Let us finally reason about the abilities of the coalition of agents a_1 and a_2. We already saw that the coalition of agents have the ability to bring about the crashing of the trains. However, even if the individual agents play a totally individual obedient strategy (or a selective individual obedient strategy for that matter), the agents can still select a strategy that can make the trains crash, illustrated by the following validities:

$$M, \Gamma, q_0 \vDash \langle\!\langle \{a_1, a_2\}_{\langle t\text{-}ob, \top\rangle}\rangle\!\rangle \Diamond \text{crash}$$

and

$$M, \Gamma, q_0 \vDash \langle\!\langle \{a_1, a_2\}_{\langle s\text{-}ob, \top\rangle}\rangle\!\rangle \Diamond \text{crash}$$

The reason for this is because these obedience assumptions apply only to the agents on an individual level, and thus we only have to take into account the abstract normative constraint associated with a_2. On the coalitional level, the agents indeed do not have an coalitional obedient strategy to bring about the fact that eventually a crash will occur:

$$M, \Gamma, q_0 \nvDash \langle\!\langle \{a_1, a_2\}_{\langle c\text{-}ob, \top\rangle}\rangle\!\rangle \Diamond \text{crash}$$

Moreover, to illustrate how strong some obedience assumptions can be, the following formula holds for every possible φ:

$$M, \Gamma, q_0 \nvDash \langle\!\langle \{a_1, a_2\}_{\langle c\text{-}dob, \top\rangle}\rangle\!\rangle \varphi$$

The reason for this is because there does not exist a joint strategy that violates both abstract normative constraints simultaneously. This illustrates an important distinction with ATL, since ATL assumes that there is always a strategy available for any (sub)coalition of agents at any moment in time. This also inherently means that checking the validity of an an-ATL formula of the form $\langle\!\langle A_{\langle\omega, \omega'\rangle}\rangle\!\rangle \varphi$ in a model M cannot be reduced to checking a formula of the form $\langle\!\langle A\rangle\!\rangle \varphi$ in a transformed model M' with removed edges, the reason simply being that M' might not be an actual valid concurrent game structure any more.

5. MODEL CHECKING

The problem of determining whether a formula in an-ATL is satisfied at a certain state reduces to the application of model checking to the concurrent game structure. The model checking problem for transition systems is discussed in [5], and in [3] model checking for ATL is discussed.

We define the extended game structure

$$S^F = \langle k, Q^F, \Pi^F, \pi^F, Ac^F, \delta^F \rangle$$

as follows:

- $Q^F = \{\langle \bot, q\rangle \mid q \in Q\} \cup$
  $\{\langle q', q\rangle \mid q', q \in Q \text{ and } q \text{ is a successor of } q' \text{ in } Q\}$.
- $\Pi^F = \Pi \cup (\Gamma \times \{enabled, taken\})$.
- For all $\langle \bot, q\rangle \in Q^F$ we have $\pi^F(\langle \bot, q\rangle) = \pi(q)$.
  For all $\langle q', q\rangle \in Q^F$ we have $\pi^F(\langle q', q\rangle) = \pi(q) \cup$
  $\{\langle\langle A, \gamma\rangle, enabled\rangle \mid \forall a \in A : \gamma(a, q') \neq \varnothing\} \cup$
  $\{\langle\langle A, \gamma\rangle, taken\rangle \mid$ exists $\langle \alpha_1, ..., \alpha_k\rangle \in D(q')$ such that $\forall a \in A : \alpha_a \in \gamma(a, q')$ and $\delta(q', \langle \alpha_1, ..., \alpha_k\rangle) = q\}$

- For all $a \in \Sigma$ and all $\langle \bot, q\rangle, \langle q', q\rangle \in Q^F$ we have $Ac^F(a, \langle \bot, q\rangle) = Ac^F(a, \langle q', q\rangle) = Ac(a, q)$.
- For all $\langle \bot, q\rangle, \langle q', q\rangle \in Q^F$ and all $\langle \alpha_1, ..., \alpha_k\rangle \in D(q)$ we have $\delta^F(\langle \bot, q\rangle, \langle \alpha_1, ..., \alpha_k\rangle) = \delta^F(\langle q', q\rangle, \langle \alpha_1, ..., \alpha_k\rangle) = \langle q, \delta(q, \langle \alpha_1, ..., \alpha_k\rangle)\rangle$.

Let f_A^Γ be a function that maps, given a coalition A and abstract normative constraint set Γ, each obedience assumption in Ω to a propositional formula with variables $\Gamma \times \{enabled, taken\}$. Thus we have $f_A^\Gamma : \Omega \mapsto \mathcal{L}_{prop}(\Gamma \times \{enabled, taken\})$. We define the function as follows (notice that we have written $\langle \varphi, e\rangle$ and $\langle \varphi, t\rangle$ as shorthand for $\langle \varphi, enabled\rangle$ and $\langle \varphi, taken\rangle$ respectively):

1. $$f_A^\Gamma(c\text{-}ob) = \bigwedge_{\langle A,\gamma\rangle \in \Gamma_A^-} (\langle\langle A, \gamma\rangle, e\rangle \rightarrow \neg\langle\langle A, \gamma\rangle, t\rangle)$$
2. $$f_A^\Gamma(t\text{-}ob) = \bigwedge_{a \in A} , \bigwedge_{\langle\{a\},\gamma\rangle \in \Gamma_{\{a\}}} (\langle\langle\{a\}, \gamma\rangle, e\rangle \rightarrow \neg\langle\langle\{a\}, \gamma\rangle, t\rangle)$$
3. $$f_A^\Gamma(s\text{-}ob) = \bigvee_{a \in A} , \bigwedge_{\langle\{a\},\gamma\rangle \in \Gamma_{\{a\}}} (\langle\langle\{a\}, \gamma\rangle, e\rangle \rightarrow \neg\langle\langle\{a\}, \gamma\rangle, t\rangle)$$
4. $$f_A^\Gamma(\top) = \top$$
5. $$f_A^\Gamma(s\text{-}dob) = \bigvee_{a \in A} , \bigvee_{\langle\{a\},\gamma\rangle \in \Gamma_{\{a\}}} (\langle\langle\{a\}, \gamma\rangle, e\rangle \rightarrow \langle\langle\{a\}, \gamma\rangle, t\rangle)$$
6. $$f_A^\Gamma(t\text{-}dob) = \bigwedge_{a \in A} , \bigvee_{\langle\{a\},\gamma\rangle \in \Gamma_{\{a\}}} (\langle\langle\{a\}, \gamma\rangle, e\rangle \rightarrow \langle\langle\{a\}, \gamma\rangle, t\rangle)$$
7. $$f_A^\Gamma(c\text{-}dob) = \bigwedge_{\langle A,\gamma\rangle \in \Gamma_A^-} (\langle\langle A, \gamma\rangle, e\rangle \rightarrow \langle\langle A, \gamma\rangle, t\rangle)$$

We then claim that evaluating a formula of the form $S, \Gamma, q \vDash \langle\!\langle A_{\langle\omega,\omega'\rangle}\rangle\!\rangle\varphi$ amounts to model checking an ATL* formula in the extended game structure.

PROPOSITION 1. *$S, \Gamma, q \vDash \langle\!\langle A_{\langle\omega,\omega'\rangle}\rangle\!\rangle\varphi$ holds if and only if:*

$$S^F, \langle \bot, q\rangle \vDash \langle\!\langle A\rangle\!\rangle(\Box f_A^\Gamma(\omega) \wedge (\Box f_{(\Sigma\setminus\Gamma)}^\Gamma(\omega') \rightarrow \varphi))$$

Although this is an ATL* formula, the model checking complexity can still be done in efficient time (polynomial in the number of transitions, the size of the abstract normative constraint set and the length of the an-ATL formula). The exact details of this result are not relevant for this paper, but we can give a basic intuition behind this finding. Consider a game structure S with m transitions and an abstract normative constraint set Γ of size g. We start out by constructing $S^F = \langle k, Q^F, \Pi^F, \pi^F, Ac^F, \delta^F\rangle$ from S. Notice that if S has m transitions, S^F has $O(m)$ states and $O(m)$ transitions. Now, the interesting cases arise when we want to check a sub-formula of the form $\langle\!\langle A_{\langle\omega,\omega'\rangle}\rangle\!\rangle \Box \varphi$ or $\langle\!\langle A_{\langle\omega,\omega'\rangle}\rangle\!\rangle\varphi_1 \mathcal{U} \varphi_2$. The idea now is that, when looking for states satisfying these conditions, we can just encode the winning and losing conditions in the game structure itself. We do this by adding two states to the game structure S^F, an "always winning" and an " always losing" state. The "always winning" state q_W makes the goal formula true forever, the "always losing" state q_L makes the goal formula false forever regardless of the goal formula $\Box\varphi$ or $\varphi_1 \mathcal{U} \varphi_2$. Notice that the states q_W and q_L do not follow the usual definitions of a Concurrent Game Structures; for example the formula $\Box\bot$ is still true at q_W even though there would not exist a valuation for q_W that makes this possible. However for the sake of model-checking this does not matter. Now consider we are model checking a formula $\langle\!\langle A_{\langle\omega,\omega'\rangle}\rangle\!\rangle\psi$ (where $\psi = \Box\theta$ or $\psi = \theta_1 \mathcal{U} \theta_2$). We proceed as follows: for each state $q \in Q^F$, if it holds that $q \vDash \neg f_A^\Gamma(\omega)$, then redirect all incoming transitions to state q_L, if it holds that $q \vDash \neg f_{(\Sigma\setminus A)}^\Gamma(\omega') \wedge f_A^\Gamma(\omega)$, then redirect all incoming transitions to q_W. This routine can be done in efficient time, and ensures that if A can only select $\langle\Gamma, A, \omega\rangle$-obedient transitions to make the goal formula true (else it would end up in q_L) and ensures that if $\Sigma\setminus A$ selects a non $\langle\Gamma, \Sigma\setminus A, \omega'\rangle$-obedient transition the goal formula is by default satisfied. Now we can just proceed with "normal" model checking, which as shown in [3], can be done in time proportional to the number of transitions in the concurrent game structure, which is $O(m)$.

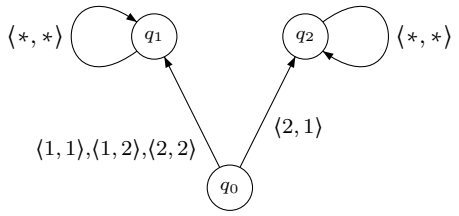

Figure 2: CGS with 2 agents.

6. SELF-SUPPORTING SETS

When we introduced the notion of abstract normative constraints, we saw that that it is still possible that an abstract normative constraint of the form $\langle A, \gamma\rangle$ is taken at a certain state in the model, even though the agents in A might not have selected the exact actions prescribed by γ. Thus, even though it seems they were selecting an action in order to avoid a violation, they still ended up in a situation where this is not the case. In these special circumstances it is the case that the agents do not have the power to autonomously avoid a violation. Autonomy is a key facet within the (multi)agent paradigm and can play a major role for the agents to decide whether they want to participate in the multiagent system or comply with the given norms, so we devote this last section to identify and classify the circumstances in which a normative constraint set limits the autonomy of (coalitions of) agents.

Let us first consider the concurrent game structure shown in Figure 2. Moreover, let us consider the normative constraint set $\Gamma = \{\langle\{a_1\}, \gamma\rangle\}$, where $\gamma(a_1, q_0) = \{1\}$, we see that agent 2, while being in state q_0, has the ability to enforce agent 1 into a violation. Namely, we see that agent 2 can select action 2, which causes agent 1 to end up in q_1 regardless of the action he chooses. Since there exists an action for agent 1 that disallows going from state q_0 to q_1 (namely action 1), even by selecting action 2 the agent can not avoid violating the constraint. This brings us to the notion of

self-supporting constraint sets. Intuitively, self-supporting means that if a normative constraint is targeted towards coalition A, the agents in this coalition have (in some way) "control" over whether or not they will violate this particular constraint. However, as we will see in this section, multiple gradations of "control" can be given. We start out with a weak form of self-supporting, called *weakly self-supporting*.

Definition 1 (Weakly Self-supporting). *Given a concurrent game structure S and abstract normative constraint set Γ, we say that Γ is weakly self-supporting with respect to S if and only if it holds that for every coalition $A \subseteq \Sigma$ and at any state $q \in Q$ there is no strategy available to A in order to force the remaining players $\Sigma \backslash A$ into a non-$\langle \Gamma, \Sigma \backslash A, c\text{-}ob \rangle$-obedient computation.*

We see that in our previous example this was not the case since agent 2 could force agent 1 into a violation by selecting action 2. The following proposition shows how we can identify weakly self-supporting constraint sets using our new an-ATL logic.

Proposition 2 (Weakly Self-supporting). *Given a concurrent game structure S and abstract normative constraint set Γ, Γ is weakly self-supporting with respect to S if and only if for all $A \subseteq \Sigma$ it holds that:*

$$S, \Gamma \vDash [\![A_{\langle \top, c\text{-}ob \rangle}]\!] \bigcirc \top$$

Let us give an intuition about why this proposition holds. Observe that, for a given A, the formula $[\![A_{\langle \top, c\text{-}ob \rangle}]\!] \bigcirc \top$ is equal to $\neg \langle\!\langle A_{\langle \top, c\text{-}ob \rangle} \rangle\!\rangle \bigcirc \bot$. Now let us suppose that $\langle\!\langle A_{\langle \top, c\text{-}ob \rangle} \rangle\!\rangle \bigcirc \bot$ holds at state q. This means that there exists a strategy for A, let us call this S_A, such that if coalition $\Sigma \backslash A$ behaved according to a $\langle \Gamma, \Sigma \backslash A, c\text{-}ob \rangle$-obedient computation, $\bigcirc \bot$ would be true. However, the latter can never be true for any computation, thus we must conclude that every computation in $out(q, S_A)$ is not $\langle \Gamma, \Sigma \backslash A, c\text{-}ob \rangle$-obedient, which implies that Γ is not weakly self-supporting with respect to S. A similar reasoning can be applied for the other way around.

Suppose for now we have a weakly self-supporting constraint set Γ with respect to S. Now, even though there exists no coalition that can enforce the other players into a non collective obedient computation, it does not mean that every coalition has a collective obedient strategy available to them. Consider the CGS shown in Figure 3, with again the normative constraint set $\Gamma = \{\langle \{a_1\}, \gamma \rangle\}$, where $\gamma(a_1, q_0) = \{1\}$. We see that Γ is weakly self-supporting with respect to S, since it is not possible for agent 2 to force agent 1 into a non-$\langle \Gamma, \{a_1\}, c\text{-}ob \rangle$-obedient computation. However, it is not the case that agent 1 has a collective obedient strategy available to him as both action 2 and 3 might bring him into state q_1. This brings up a more stronger notion of self-supporting constraint sets, namely those in which each coalition always has a collective obedient strategy available to them. If this is the case, we say that a normative constraint set is *strongly self-supporting* with respect to a concurrent game structure.

Definition 2 (Strongly Self-supporting). *Given a concurrent game structure S and abstract normative constraint set Γ, we say that Γ is strongly self-supporting with respect to S if and only if it holds that for every coalition $A \subseteq \Sigma$ there is at any state $q \in Q$ a collective obedient strategy available to them.*

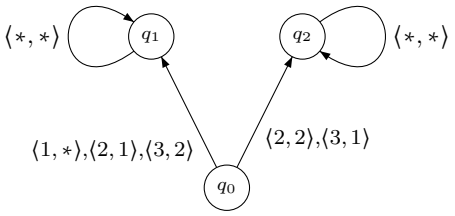

Figure 3: CGS with 2 agents.

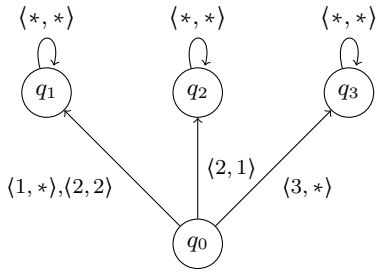

Figure 4: Another CGS with 2 agents.

Just like in the case of weakly self-supporting constraint sets, we can identify when a normative constraint set is strongly self-supporting with respect to a concurrent game structure with the use of our an-ATL logic.

Proposition 3 (Strongly Self-supporting). *Given a concurrent game structure S and abstract normative constraint set Γ, we say that Γ is strongly self-supporting with respect to S if and only if for all $A \subseteq \Sigma$ it holds that:*

$$S, \Gamma \vDash \langle\!\langle A_{\langle c\text{-}ob, \top \rangle} \rangle\!\rangle \bigcirc \top$$

We now claim that if a normative constraint set is strongly self-supporting, it is also weakly self-supporting. This is not hard to verify because, as we have already seen in Section 4, we have the result that $S, \Gamma, q \vDash \langle\!\langle A_{\langle \omega, \omega' \rangle} \rangle\!\rangle \varphi$ implies $S, \Gamma, q \vDash [\![(\Sigma \backslash A)_{\langle \omega', \omega \rangle}]\!] \varphi$, and thus that for all $A \subseteq \Sigma$, $S, \Gamma \vDash \langle\!\langle A_{\langle c\text{-}ob, \top \rangle} \rangle\!\rangle \bigcirc \top$ implies for all $A \subseteq \Sigma$, $S, \Gamma \vDash [\![A_{\langle \top, c\text{-}ob \rangle}]\!] \bigcirc \top$ (but not the other way around).

Again suppose we have a strongly self-supporting constraint set Γ with respect to S. An example can be seen in Figure 4 with again the normative constraint set $\Gamma = \{\langle \{a_1\}, \gamma \rangle\}$, where $\gamma(a_1, q_0) = \{1\}$. Although there is a collective obedient strategy available for agent 1 in q_0, namely selecting at this state action 3, there is still an action available to him which is not in the constraint set but can causes him to violate a normative constraint, i.e., if agent 1 selects action 2 in q_0, there is still a possibility that agent 2 selects action 2. This gives rise to yet another notion of self-supporting constraint sets, namely that of *unconditional self-supporting normative constraint sets*. Intuitively, if this is the case it means that only the (joint) actions normatively demotivated by the constraint set Γ will result in a violation. Thus, "unconditional" does not mean that the status of whether or not a constraint set is self-supporting does not rely on the game structure itself, it merely means that if it is established that a constraint set is unconditionally self-supporting, it is sufficient to only look at the constraint set in order to select a collective obedient strategy. To make

this more formal, let us introduce the notion of an A-move. An A-move is a function c_q^A that maps each player $a \in A$ to an action for that player, thus we have $c_q^A(a) \in Ac(a,q)$. We write $C(A,q)$ for the set of all A-moves c_q^A for coalition A at state q. Now we define $C_\Gamma(A,q)$ as the set of all A-moves such that it holds that:

$$C_\Gamma(A,q) = \left\{c_q^A \in C(A,q) \mid \forall \langle \gamma, A' \rangle \in \Gamma_A^- : allows(\langle \gamma, A' \rangle, c_q^A)\right\}$$

where: $allows(\langle \gamma, A' \rangle, c_q^A) =$

$$(\forall a \in A' : \gamma(a,q) \neq \varnothing) \Rightarrow \bigvee_{a \in A'} c_q^A(a) \notin \gamma(a,q)$$

In words, the set $C_\Gamma(A,q)$ contain all A-moves for coalition A which are not normatively demotivated by a constraint in Γ_A^-. We can now give the formal definition of unconditional self-supporting constraint sets.

Definition 3 (Unconditional Self-supporting). . *Given a concurrent game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$, we say that an abstract normative constraint set Γ is unconditional self-supporting with respect to S if and only if for all $A \subseteq \Sigma$ it holds that for all states $q \in Q$ it is the case that $C_\Gamma(A,q)$ is non-empty and for every A-move $c_q^A \in C_\Gamma(A,q)$, it holds that by playing it, all of the normative constraints $\langle A', \gamma \rangle \in \Gamma_A^-$ will be either not enabled or not taken.*

Note that an unconditional self-supporting normative constraint set is (by definition) also strongly self-supporting since we demanded $C_\Gamma(A,q)$ to be non-empty. Now given a concurrent game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$ and an abstract normative constraint set Γ, we can define out_q as a function from a set of A-moves to the set of all possible states the agents can arrive in by playing such an A-move.

$$out_q(C(A,q)) = \{\delta(q, d_q) \in Q \mid d_q \in g(c_q^A) \text{ and } c_q^A \in C(A,q)\}$$

where

$$g(c_q^A) = \{\langle \alpha_1, ..., \alpha_k \rangle \in D(q) \mid \forall a \in A : c_q^A(a) = \alpha_a\}$$

Using this definition, the following proposition now states how we can verify when an abstract normative constraint set is unconditional self-supporting with respect to a game structure.

Proposition 4 (Unconditional Self-supporting). *A normative constraint set Γ with respect to a concurrent game structure $S = \langle k, Q, \Pi, \pi, Ac, \delta \rangle$ is unconditional self-supporting iff $\forall q \in Q$ and $\forall A \subseteq \Sigma : out_q(C_\Gamma(A,q)) \neq \varnothing$ and $out_q(C_\Gamma(A,q)) \cap out_q(C(A,q) \backslash C_\Gamma(A,q)) = \varnothing$.*

To see why this proposition holds, note that it states that the states reachable from an A-move in $C_\Gamma(A,q)$ (all the A-moves which are not normatively demotivated by the constraints in Γ_A^-) and the states reachable from $C(A,q) \backslash C_\Gamma(A,q)$ (all the A-moves which are normatively demotivated by the constraints in Γ_A^-) must be disjoint. If this is not the case, then there exists an $A - move$ in $C_\Gamma(A,q)$ which by playing it would result in one of the constraints in Γ_A^- to be enabled and taken, thus not unconditionally self-supporting. A similar reasoning can be applied for the other way around.

What we have seen in this section is that it is possible to characterize and identify multiple levels of "control" the agents have over the ability to adhere to the normative constraints. As we already argued, identifying when a normative constraint set is not (weakly/strongly/unconditional) self-supporting with respect to a concurrent game structure may play a crucial role for the agents in order to decide whether to participate in the system, since they restrict the autonomy of the agents in some way.

7. DISCUSSION AND FUTURE RESEARCH

In this paper we have developed a model of normative systems that allows reasoning about agents' (normative) abilities under a multitude of compliance assumptions. This can be both crucial in the design and validation of these systems. To do this, we introduced the notion of abstract normative constraints and we developed an extension of Alternating Temporal Logic, an-ATL, to allow reasoning about the abilities of (coalitions of) agents under different compliance assumptions. Moreover, we showed that model-checking an-ATL formula's remains close to the complexity of model-checking standard ATL. In the last part of the paper, we discussed the notion of self-supporting constraint sets and explained its relation to autonomy of agents. In particular, we explained that if a constraint set is self-supporting, the agents have (in some way) control over whether they can avoid a violation.

There are multiple ways to extend this line of research. Firstly, it would be very interesting to consider more compliance assumptions. As can be seen in Section 5, given a normative constraint set Γ, our logic allows us to create arbitrary complex obedience assumptions in the language $\mathcal{L}_{prop}(\Gamma \times \{enabled, taken\})$. Moreover, it would be interesting to incorporate goals and preferences of agents and see how they relate to the obedience behaviour of the other agents. Finally, the question what "good coalitions" for agents are in order to not violate any of the norms must be answered. Perhaps this question can be related to the topic of coalitional game theory.

8. REFERENCES

[1] T. Ågotnes, W. van der Hoek, and M. Woolridge. Robust normative systems and a logic of norm compliance. *Logic journal of the IGPL*, 18:4–30, 2010.

[2] T. Ågotnes, M. Wooldridge, and W. van der Hoek. Normative system games. In *AAMAS'07*, pages 876–883, 2007.

[3] R. Alur, T. A. Henzinger, and O. Kupferman. Alternating-time temporal logic. *Journal of the ACM*, 49(5):672–713, 2002.

[4] N. Bulling and M. Dastani. Verifying normative behaviour via normative mechanism design. In *IJCAI'11*, pages 103–108, 2011.

[5] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model Checking*. MIT Press, Cambridge, 1999. ISBN 0-262-03270-8.

[6] Y. Moses and M. Tennenholtz. Artificial social systems. *Computers and AI*, 14(6):533 – 562, 1995.

[7] Y. Shoham and M. Tennenholtz. On the synthesis of useful social laws for artificial agent societies. In *AAAI'92*, pages 276–281, 1992.

[8] W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: effectiveness, feasibility, and synthesis. *Synthese*, 156(1):1–19, 2007.

[9] M. Woolridge and W. van der Hoek. On obligations and normative ability. *Journal of Applied Logic*, 3:396–420, 2005.