Normative System Games

Thomas Ågotnes
Dept of Computer Engineering
Bergen University College
PB. 2030, N-5020 Bergen
Norway
tag@hib.no

Wiebe van der Hoek
Dept of Computer Science
University of Liverpool
Liverpool L69 7ZF
UK
wiebe@csc.liv.ac.uk

Michael Wooldridge
Dept of Computer Science
University of Liverpool
Liverpool L69 7ZF
UK
mjw@csc.liv.ac.uk

ABSTRACT

We develop a model of normative systems in which agents are assumed to have multiple goals of increasing priority, and investigate the computational complexity and game theoretic properties of this model. In the underlying model of normative systems, we use Kripke structures to represent the possible transitions of a multi-agent system. A normative system is then simply a subset of the Kripke structure, which contains the arcs that are forbidden by the normative system. We specify an agent's goals as a hierarchy of formulae of Computation Tree Logic (CTL), a widely used logic for representing the properties of Kripke structures: the intuition is that goals further up the hierarchy are preferred by the agent over those that appear further down the hierarchy. Using this scheme, we define a model of ordinal utility, which in turn allows us to interpret our Kripke-based normative systems as games, in which agents must determine whether to comply with the normative system or not. We then characterise the computational complexity of a number of decision problems associated with these Kripke-based normative system games; for example, we show that the complexity of checking whether there exists a normative system which has the property of being a Nash implementation is NP-complete.

Categories and Subject Descriptors

I.2.11 [**Distributed Artificial Intelligence**]: Multiagent Systems; I.2.4 [**Knowledge representation formalisms and methods**]

General Terms

Theory

Keywords

normative systems, goals, logic, games, complexity

1. INTRODUCTION

Normative systems, or social laws, have proved to be an attractive approach to coordination in multi-agent systems [13, 14, 10, 15, 1]. Although the various approaches to normative systems proposed in the literature differ on technical details, they all share the same basic intuition that a normative system is a set of constraints on the behaviour of agents in the system; by imposing these constraints, it is hoped that some desirable objective will emerge. The idea of using social laws to coordinate multi-agent systems was proposed by Shoham and Tennenholtz [13, 14]; their approach was extended by van der Hoek *et al.* to include the idea of specifying a desirable global objective for a social law as a logical formula, with the idea being that the normative system would be regarded as successful if, after implementing it (i.e., after eliminating all forbidden actions), the objective formula was guaranteed to be satisfied in the system [15]. However, this model did not take into account the *preferences* of individual agents, and hence neglected to account for possible strategic behaviour by agents when deciding whether to comply with the normative system or not. This model of normative systems was further extended by attributing to each agent a single goal in [16]. However, this model was still too impoverished to capture the kinds of decision making that take place when an agent decides whether or not to comply with a social law. In reality, strategic considerations come into play: an agent takes into account not just whether the normative system would be beneficial for itself, but also whether other agents will rationally choose to participate.

In this paper, we develop a model of normative systems in which agents are assumed to have *multiple* goals, of increasing priority. We specify an agent's goals as a hierarchy of formulae of Computation Tree Logic (CTL), a widely used logic for representing the properties of Kripke structures [8]: the intuition is that goals further up the hierarchy are preferred by the agent over those that appear further down the hierarchy. Using this scheme, we define a model of ordinal utility, which in turn allows us to interpret our Kripke-based normative systems as games, in which agents must determine whether to comply with the normative system or not. We thus provide a very natural bridge between logical structures and languages and the techniques and concepts of game theory, which have proved to be very powerful for analysing social contract-style scenarios such as normative systems [3, 4]. We then characterise the computational complexity of a number of decision problems associated with these Kripke-based normative system games; for example, we show that the complexity of checking whether there exists a normative system which has the property of being a Nash implementation is NP-complete.

AAMAS'07 May 14–18 2007, Honolulu, Hawai'i, USA.

2. KRIPKE STRUCTURES AND CTL

We use *Kripke structures* as our basic semantic model for multi-agent systems [8]. A Kripke structure is essentially a directed graph, with the vertex set S corresponding to possible *states* of the system being modelled, and the relation $R \subseteq S \times S$ capturing the possible *transitions* of the system; intuitively, these transitions are caused by *agents* in the system performing *actions*, although we do not include such actions in our semantic model (see, e.g., [13, 2, 15] for related models which include actions as first class citizens). We let S^0 denote the set of possible *initial states* of the system. Our model is intended to correspond to the well-known *interleaved concurrency* model from the reactive systems literature: thus an arc corresponds to the execution of an atomic action by one of the processes in the system, which we call *agents*.

It is important to note that, in contrast to such models as [2, 15], we are therefore here *not* modelling *synchronous* action. This assumption is not in fact essential for our analysis, but it *greatly* simplifies the presentation. However, we find it convenient to include within our model the agents that cause transitions. We therefore assume a set A of agents, and we label each transition in R with the agent that causes the transition via a function $\alpha : R \to A$. Finally, we use a vocabulary $\Phi = \{p, q, \ldots\}$ of Boolean variables to express the properties of individual states S: we use a function $V : S \to 2^\Phi$ to label each state with the Boolean variables true (or satisfied) in that state.

Collecting these components together, an *agent-labelled Kripke structure* (over Φ) is a 6-tuple:

$$K = \langle S, S^0, R, A, \alpha, V\rangle, \text{ where:}$$

- S is a finite, non-empty set of *states*,
- $S^0 \subseteq S$ ($S^0 \neq \emptyset$) is the set of *initial states*;
- $R \subseteq S \times S$ is a total binary relation on S, which we refer to as the *transition relation*[1];
- $A = \{1, \ldots, n\}$ is a *set of agents*;
- $\alpha : R \to A$ labels each transition in R with an agent; and
- $V : S \to 2^\Phi$ labels each state with the set of propositional variables true in that state.

In the interests of brevity, we shall hereafter refer to an agent-labelled Kripke structure simply as a *Kripke structure*. A *path* over a transition relation R is an infinite sequence of states $\pi = s_0, s_1, \ldots$ which must satisfy the property that $\forall u \in \mathbb{N}$: $(s_u, s_{u+1}) \in R$. If $u \in \mathbb{N}$, then we denote by $\pi[u]$ the component indexed by u in π (thus $\pi[0]$ denotes the first element, $\pi[1]$ the second, and so on). A path π such that $\pi[0] = s$ is an *s-path*. Let $\Pi_R(s)$ denote the set of s-paths over R; since it will usually be clear from context, we often omit reference to R, and simply write $\Pi(s)$. We will sometimes refer to and think of an s-path as a possible computation, or system evolution, from s.

EXAMPLE 1. *Our running example is of a system with a single non-sharable resource, which is desired by two agents. Consider the Kripke structure depicted in Figure 1. We have two states, s and t, and two corresponding Boolean variables p_1 and p_2, which are mutually exclusive. Think of p_i as meaning "agent i has currently control over the resource". Each agent has two possible actions, when in possession of the resource: either give it away, or keep it. Obviously there are infinitely many different s-paths and t-paths. Let us say that our set of initial states S^0 equals $\{s, t\}$, i.e., we don't make any assumptions about who initially has control over the resource.*

[1] In the branching time temporal logic literature, a relation $R \subseteq S \times S$ is said to be total iff $\forall s \, \exists s' : (s, s') \in R$. Note that the term "total relation" is sometimes used to refer to relations $R \subseteq S \times S$ such that for every pair of elements $s, s' \in S$ we have either $(s, s') \in R$ or $(s', s) \in R$; we are *not* using the term in this way here. It is also worth noting that for some domains, other constraints may be more appropriate than simple totality. For example, one might consider the *agent totality* requirement, that in every state, every agent has at least one possible transition available: $\forall s \forall i \in A \exists s' : (s, s') \in R$ and $\alpha(s, s') = i$.

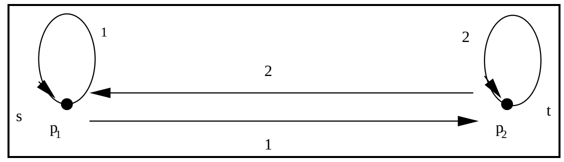

Figure 1: The resource control running example.

2.1 CTL

We now define Computation Tree Logic (CTL), a branching time temporal logic intended for representing the properties of Kripke structures [8]. Note that since CTL is well known and widely documented in the literature, our presentation, though complete, will be somewhat terse. We will use CTL to express agents' goals.

The syntax of CTL is defined by the following grammar:

$$\varphi ::= \top \mid p \mid \neg\varphi \mid \varphi \vee \varphi \mid \mathsf{E}\bigcirc\varphi \mid \mathsf{E}(\varphi\,\mathcal{U}\,\varphi) \mid \mathsf{A}\bigcirc\varphi \mid \mathsf{A}(\varphi\,\mathcal{U}\,\varphi)$$

where $p \in \Phi$. We denote the set of CTL formula over Φ by $\mathcal{L}_\Phi$; since Φ is understood, we usually omit reference to it.

The semantics of CTL are given with respect to the satisfaction relation "$\models$", which holds between pairs of the form K, s, (where K is a Kripke structure and s is a state in K), and formulae of the language. The satisfaction relation is defined as follows:

$K, s \models \top$;

$K, s \models p$ iff $p \in V(s)$ (where $p \in \Phi$);

$K, s \models \neg\varphi$ iff not $K, s \models \varphi$;

$K, s \models \varphi \vee \psi$ iff $K, s \models \varphi$ or $K, s \models \psi$;

$K, s \models \mathsf{A}\bigcirc\varphi$ iff $\forall \pi \in \Pi(s) : K, \pi[1] \models \varphi$;

$K, s \models \mathsf{E}\bigcirc\varphi$ iff $\exists \pi \in \Pi(s) : K, \pi[1] \models \varphi$;

$K, s \models \mathsf{A}(\varphi\,\mathcal{U}\,\psi)$ iff $\forall \pi \in \Pi(s), \exists u \in \mathbb{N}$, s.t. $K, \pi[u] \models \psi$ and $\forall v, (0 \leq v < u) : K, \pi[v] \models \varphi$

$K, s \models \mathsf{E}(\varphi\,\mathcal{U}\,\psi)$ iff $\exists \pi \in \Pi(s), \exists u \in \mathbb{N}$, s.t. $K, \pi[u] \models \psi$ and $\forall v, (0 \leq v < u) : K, \pi[v] \models \varphi$

The remaining classical logic connectives ("$\wedge$", "$\to$", "$\leftrightarrow$") are assumed to be defined as abbreviations in terms of $\neg, \vee$, in the conventional manner. The remaining CTL temporal operators are defined:

$$\begin{array}{lclclcl} \mathsf{A}\Diamond\varphi & \equiv & \mathsf{A}(\top\,\mathcal{U}\,\varphi) & \quad & \mathsf{E}\Diamond\varphi & \equiv & \mathsf{E}(\top\,\mathcal{U}\,\varphi) \\ \mathsf{A}\Box\varphi & \equiv & \neg\mathsf{E}\Diamond\neg\varphi & & \mathsf{E}\Box\varphi & \equiv & \neg\mathsf{A}\Diamond\neg\varphi \end{array}$$

We say φ is *satisfiable* if $K, s \models \varphi$ for some Kripke structure K and state s in K; φ is *valid* if $K, s \models \varphi$ for all Kripke structures K and states s in K. The problem of checking whether $K, s \models \varphi$ for given K, s, φ (*model checking*) can be done in deterministic polynomial time, while checking whether a given φ is satisfiable or whether φ is valid is EXPTIME-complete [8]. We write $K \models \varphi$ if $K, s_0 \models \varphi$ for all $s_0 \in S^0$, and $\models \varphi$ if $K \models \varphi$ for all K.

3. NORMATIVE SYSTEMS

For our purposes, a *normative system* is simply *a set of constraints on the behaviour of agents in a system* [1]. More precisely, a normative system defines, for every possible system transition, whether or not that transition is considered to be legal or not. Different normative systems may differ on whether or not a transition is legal. Formally, a normative system η (w.r.t. a Kripke structure $K = \langle S, S^0, R, A, \alpha, V\rangle$) is simply a subset of R, such that $R \setminus \eta$ is a total relation. The requirement that $R \setminus \eta$ is total is a *reasonableness* constraint: it prevents normative systems which lead to states with no successor. Let $N(R) = \{\eta : (\eta \subseteq R) \;\&\; (R \setminus \eta \text{ is total})\}$ be the set of normative systems over R. The intended interpretation of a normative system η is that $(s, s') \in \eta$ means transition (s, s') is forbidden in the context of η; hence $R \setminus \eta$ denotes the *legal* transitions of η. Since it is assumed η is reasonable, we are guaranteed that a legal outward transition exists for every state. We denote the *empty* normative system by $\eta_\emptyset$, so $\eta_\emptyset = \emptyset$. Note that the empty normative system $\eta_\emptyset$ is reasonable with respect to *any* transition relation R.

The effect of *implementing* a normative system on a Kripke structure is to eliminate from it all transitions that are forbidden according to this normative system (see [15, 1]). If K is a Kripke structure, and η is a normative system over K, then $K \dagger \eta$ denotes the Kripke structure obtained from K by deleting transitions forbidden in η. Formally, if $K = \langle S, S^0, R, A, \alpha, V\rangle$, and $\eta \in N(R)$, then let $K \dagger \eta = K'$ be the Kripke structure $K' = \langle S', S^{0'}, R', A', \alpha', V'\rangle$ where:

- $S = S'$, $S^0 = S^{0'}$, $A = A'$, and $V = V'$;
- $R' = R \setminus \eta$; and
- α' is the restriction of α to R':

$$\alpha'(s, s') = \begin{cases} \alpha(s, s') & \text{if } (s, s') \in R' \\ \text{undefined} & \text{otherwise.} \end{cases}$$

Notice that for all K, we have $K \dagger \eta_\emptyset = K$.

EXAMPLE 1. *(continued) When thinking in terms of* fairness*, it seems natural to consider normative systems η that contain (s, s) or (t, t). A normative system with (s, t) would not be fair, in the sense that $\mathsf{A}\Diamond\mathsf{A}\Box\neg p_1 \vee \mathsf{A}\Diamond\mathsf{A}\Box\neg p_2$ holds: in all paths, from some moment on, one agent will have control forever. Let us, for later reference, fix $\eta_1 = \{(s, s)\}$, $\eta_2 = \{(t, t)\}$, and $\eta_3 = \{(s, s), (t, t)\}$.*

Later, we will address the issue of whether or not agents should rationally choose to *comply* with a particular normative system. In this context, it is useful to define operators on normative systems which correspond to groups of agents "defecting" from the normative system. Formally, let $K = \langle S, S^0, R, A, \alpha, V\rangle$ be a Kripke structure, let $C \subseteq A$ be a set of agents over K, and let η be a normative system over K. Then:

- $\eta \upharpoonright C$ denotes the normative system that is the same as η except that it only contains the arcs of η that correspond to the actions of agents in C. We call $\eta \upharpoonright C$ the *restriction of η to C*, and it is defined as:

  $$\eta \upharpoonright C = \{(s, s') : (s, s') \in \eta \;\&\; \alpha(s, s') \in C\}.$$

  Thus $K \dagger (\eta \upharpoonright C)$ is the Kripke structure that results if only the agents in C choose to comply with the normative system.

- $\eta \upharpoonleft C$ denotes the normative system that is the same as η except that it only contains the arcs of η that *do not* correspond to actions of agents in C. We call $\eta \upharpoonleft C$ the *exclusion of C from η*, and it is defined as:

  $$\eta \upharpoonleft C = \{(s, s') : (s, s') \in \eta \;\&\; \alpha(s, s') \notin C\}.$$

  Thus $K \dagger (\eta \upharpoonleft C)$ is the Kripke structure that results if only the agents in C choose *not* to comply with the normative system (i.e., the only ones who comply are those in $A \setminus C$).

Note that we have $\eta \upharpoonleft C = \eta \upharpoonright (A \setminus C)$ and $\eta \upharpoonright C = \eta \upharpoonleft (A \setminus C)$.

EXAMPLE 1. *(Continued) We have $\eta_1 \upharpoonright \{1\} = \eta_1 = \{(s, s)\}$, while $\eta_1 \upharpoonleft \{1\} = \eta_\emptyset = \eta_1 \upharpoonright \{2\}$. Similarly, we have $\eta_3 \upharpoonright \{1\} = \{(s, s)\}$ and $\eta_3 \upharpoonleft \{1\} = \{(t, t)\}$.*

4. GOALS AND UTILITIES

Next, we want to be able to capture the goals that agents have, as these will drive an agent's strategic considerations – particularly, as we will see, considerations about whether or not to comply with a normative system. We will model an agent's goals as a *prioritised list* of CTL formulae, representing increasingly desired properties that the agent wishes to hold. The intended interpretation of such a *goal hierarchy* γ_i for agent $i \in A$ is that the "further up the hierarchy" a goal is, the more it is desired by i. Note that we assume that *if an agent can achieve a goal at a particular level in its goal hierarchy, then it is unconcerned about goals lower down the hierarchy*. Formally, a *goal hierarchy*, γ, (over a Kripke structure K) is a finite, non-empty sequence of CTL formulae

$$\gamma = (\varphi_0, \varphi_1, \ldots, \varphi_k)$$

in which, by convention, $\varphi_0 = \top$. We use a natural number indexing notation to extract the elements of a goal hierarchy, so if $\gamma = (\varphi_0, \varphi_1, \ldots, \varphi_k)$ then $\gamma[0] = \varphi_0$, $\gamma[1] = \varphi_1$, and so on. We denote the largest index of any element in γ by $|\gamma|$.

A particular Kripke structure K is said to satisfy a goal at index x in goal hierarchy γ if $K \models \gamma[x]$, i.e., if $\gamma[x]$ is satisfied in all initial states S^0 of K. An obvious potential property of goal hierarchies is *monotonicity*: where goals at higher levels in the hierarchy logically imply those at lower levels in the hierarchy. Formally, a goal hierarchy γ is monotonic if for all $x \in \{1, \ldots, |\gamma|\} \subseteq \mathbb{N}$, we have $\models \gamma[x] \rightarrow \gamma[x-1]$. The simplest type of monotonic goal hierarchy is where $\gamma[x+1] = \gamma[x] \wedge \psi_{x+1}$ for some ψ_{x+1}, so at each successive level of the hierarchy, we add new constraints to the goal of the previous level. Although this is a natural property of many goal hierarchies, it is not a property we demand of *all* goal hierarchies.

EXAMPLE 1. *(continued) Suppose the agents have similar, but opposing goals: each agent i wants to keep the source as often and long as possible for himself. Define each agent's goal hierarchy as:*

$$\begin{aligned} \gamma_i = (\quad & \varphi_0^i = \top, && \varphi_1^i = \mathsf{E}\Diamond p_i, \\ & \varphi_2^i = \mathsf{E}\Box\mathsf{E}\Diamond p_i, && \varphi_3^i = \mathsf{E}\Diamond\mathsf{E}\Box p_i, \\ & \varphi_4^i = \mathsf{A}\Box\mathsf{E}\Diamond p_i, && \varphi_5^i = \mathsf{E}\Diamond\mathsf{A}\Box p_i \\ & \varphi_6^i = \mathsf{A}\Box\mathsf{A}\Diamond p_i, && \varphi_7^i = \mathsf{A}\Box(\mathsf{A}\Diamond p_i \wedge \mathsf{E}\Box p_i), \\ & \varphi_8^i = \mathsf{A}\Box p_i \;\;) \end{aligned}$$

The most desired goal of agent i is to, in every computation, always have the resource, p_i (this is expressed in φ_8^i). Thanks to our reasonableness constraint, this goal implies φ_7^i which says that, no matter how the computation paths evolve, it will always be that all *continuations will hit a point in which p_i, and, moreover, there is a continuation in which p_i always holds. Goal φ_6^i is a fairness constraint implied by it. Note that $\mathsf{A}\Diamond p_i$ says that every computation eventually reaches a p_i state. This may mean that after p_i has happened, it will never happen again. φ_6^i circumvents this: it says that, no matter where you are, there should be a future p_i state. The goal φ_5^i is like the strong goal φ_8^i but it accepts that this is only achieved in some computation, eventually. φ_4^i requires that in every path, there is always a continuation that eventually gives p_i. Goal φ_3^i says that p_i should be true on some branch, from some moment on. It implies φ_2^i which expresses that there is a computation such that everywhere during it, it is possible to choose a continuation that eventually satisfies p_i. This implies φ_1^i, which says that p_i should at least not be impossible. If we even drop that demand, we have the trivial goal φ_0^i.*

We remark that it may seem more natural to express a fairness constraint φ_6^i as $\mathsf{A}\Box\Diamond p_i$. However, this is not a proper CTL *formula. It is in fact a formula in* CTL* *[9], and in this logic, the two expressions would be equivalent. However, our basic complexity results in the next sections would not hold for the richer language* CTL*[2]*, and the price to pay for this is that we have to formulate our desired goals in a somewhat more cumbersome manner than we might ideally like. Of course, our basic framework does not demand that goals are expressed in* CTL*; they could equally well be expressed in* CTL* *or indeed* ATL *[2] (as in [15]). We comment on the implications of alternative goal representations at the conclusion of the next section.*

A *multi-agent system* collects together a Kripke structure (representing the basic properties of a system under consideration: its state space, and the possible state transitions that may occur in it), together with a goal hierarchy, one for each agent, representing the aspirations of the agents in the system. Formally, a multi-agent system, M, is an $(n+1)$-tuple:

$$M = \langle K, \gamma_1, \ldots, \gamma_n \rangle$$

where K is a Kripke structure, and for each agent i in K, γ_i is a goal hierarchy over K.

4.1 The Utility of Normative Systems

We can now define the *utility of a Kripke structure* for an agent. The idea is that the utility of a Kripke structure is the highest index of any goal that is guaranteed for that agent in the Kripke structure. We make this precise in the function $u_i(\cdot)$:

$$u_i(K) = \max\{j : 0 \leq j \leq |\gamma_i| \;\&\; K \models \gamma_i[j]\}$$

Note that using these definitions of goals and utility, it never makes sense to have a goal φ at index n if there is a logically weaker goal ψ at index $n+k$ in the hierarchy: by definition of utility, it could never be n for any structure K.

EXAMPLE 1. *(continued) Let $M = \langle K, \gamma_1, \gamma_2 \rangle$ be the multi-agent system of Figure 1, with γ_1 and γ_2 as defined earlier in this example. Recall that we have defined S^0 as $\{s, t\}$. Then, $u_1(K) = u_2(K) = 4$: goal φ_4 is true in S^0, but φ_5 is not. To see that $\varphi_4^2 = \mathsf{A}\Box\mathsf{E}\Diamond p_2$ is true in s for instance: note that on ever path it is always the case that there is a transition to t, in which p_2 is true.*

Notice that since for any goal hierarchy γ_i we have $\gamma[0] = \top$, then for all Kripke structures, $u_i(K)$ is well defined, with $u_i(K) \geq$

η	$\delta_1(K,\eta)$	$\delta_2(K,\eta)$
$\eta_\emptyset$	0	0
η_1	0	3
η_2	3	0
η_3	2	2

	C	D
C	$(2,2)$	$(0,3)$
D	$(3,0)$	$(0,0)$

Figure 2: Benefits of implementing a normative system η (left) and pay-offs for the game Σ_M.

0. Note that this is an *ordinal* utility measure: it tells us, for any given agent, the *relative* utility of different Kripke structures, but utility values are not on some standard system-wide scale. The fact that $u_i(K_1) > u_i(K_2)$ certainly means that i strictly prefers K_1 over K_2, but the fact that $u_i(K) > u_j(K)$ does *not* mean that i values K more highly than j. Thus, it does not make sense to compare utility values *between* agents, and so for example, some system wide measures of utility, (notably those measures that aggregate individual utilities, such as social welfare), do not make sense when applied in this setting. However, as we shall see shortly, other measures – such as Pareto efficiency – can be usefully applied.

There are other representations for goals, which would allow us to define cardinal utilities. The simplest would be to specify goals γ for an agent as a finite, non-empty, one-to-one relation: $\gamma \subseteq \mathcal{L} \times \mathbb{R}$. We assume that the x values in pairs $(\varphi, x) \in \gamma$ are specified so that x for agent i means the same as x for agent j, and so we have cardinal utility. We then define the utility for i of a Kripke structure K as$u_i(K) = \max\{x : (\varphi, x) \in \gamma_i \;\&\; K \models \varphi\}$. The results of this paper in fact hold irrespective of which of these representations we actually choose; we fix upon the goal hierarchy approach in the interests of simplicity.

Our next step is to show how, in much the same way, we can lift the utility function from Kripke structures to normative systems. Suppose we are given a multi-agent system $M = \langle K, \gamma_1, \ldots, \gamma_n \rangle$ and an associated normative system η over K. Let for agent i, $\delta_i(K, K')$ be the difference in his utility when moving from K to K': $\delta_i(K, K') = u_i(K') - u_i(K)$. Then the utility of η to agent i wrt K is $\delta_i(K, K \dagger \eta)$. We will sometimes abuse notation and just write $\delta_i(K, \eta)$ for this, and refer to it as the benefit for agent i of implementing η in K. Note that this benefit can be negative.

Summarising, the utility of a normative system to an agent is the difference between the utility of the Kripke structure in which the normative system was implemented and the original Kripke structure. If this value is greater than 0, then the agent would be better off if the normative system were imposed, while if it is less than 0 then the agent would be worse off if η were imposed than in the original system. We say η is *individually rational* for i wrt K if $\delta_i(K, \eta) > 0$, and individually rational *simpliciter* if η is individually rational for every agent.

A *social system* now is a pair

$$\Sigma = \langle M, \eta \rangle$$

where M is a multi-agent system, and η is a normative system over M.

EXAMPLE 1. *The table at the left hand in Figure 2 displays the utilities $\delta_i(K, \eta)$ of implementing η in the Kripke structure of our running example, for the normative systems $\eta = \eta_\emptyset, \eta_1, \eta_2$ and η_3, introduced before. Recall that $u_1(K) = u_2(K) = 4$.*

4.2 Universal and Existential Goals

[2] CTL* model checking is PSPACE-complete, and hence much worse (under standard complexity theoretic assumptions) than model checking CTL [8].

Keeping in mind that a norm η *restricts* the possible transitions of the model under consideration, we make the following observation, borrowing from [15]. Some classes of goals are monotonic or anti-monotonic with respect to adding additional constraints to a system. Let us therefore define two fragments of the language of CTL: the universal language L^u with typical element μ, and the existential fragment L^e with typical element ε.

$$\mu ::= \top \mid p \mid \neg p \mid \mu \vee \mu \mid \mathsf{A}\bigcirc\mu \mid \mathsf{A}\Box\mu \mid \mathsf{A}(\mu\,\mathcal{U}\,\mu)$$

$$\varepsilon ::= \top \mid p \mid \neg p \mid \varepsilon \vee \varepsilon \mid \mathsf{E}\bigcirc\varepsilon \mid \mathsf{E}\Diamond\varepsilon \mid \mathsf{E}(\varepsilon\,\mathcal{U}\,\varepsilon)$$

Let us say, for two Kripke structures $K_1 = \langle S, S^0, R_1, A, \alpha, V\rangle$ and $K_2 = \langle S, S^0, R_2, A, \alpha, V\rangle$ that K_1 is a subsystem of K_2 and K_2 is a supersystem of K_1, written $K_1 \sqsubseteq K_2$ iff $R_1 \subseteq R_2$. Note that typically $K \dagger \eta \sqsubseteq K$. Then we have (cf. [15]).

THEOREM 1. *Suppose $K_1 \sqsubseteq K_2$, and $s \in S$. Then*

$\forall \varepsilon \in L^e : K_1, s \models \varepsilon \Rightarrow K_2, s \models \varepsilon$

$\forall \mu \in L^u : K_2, s \models \mu \Rightarrow K_1, s \models \mu$

This has the following effect on imposing a new norm:

COROLLARY 1. *Let K be a structure, and η a normative system. Let γ_i denote a goal hierarchy for agent i.*

1. *Suppose agent i's utility $u_i(K)$ is n, and $\gamma_i[n] \in L^u$, (i.e., $\gamma_i[n]$ is a universal formula). Then, for any normative system η, $\delta_i(K, \eta) \geq 0$.*
2. *Suppose agent i's utility $u_i(K \dagger \eta)$ is n, and $\gamma_i[n]$ is an existential formula ε. Then, $\delta_i(K \dagger \eta, K) \geq 0$.*

Corollary 1's first item says that an agent whose current maximal goal in a system is a universal formula, need never fear the imposition of a new norm η. The reason is that his current goal will at least remain true (in fact a goal higher up in the hierarchy may *become* true). It follows from this that an agent with only universal goals can only gain from the imposition of normative systems η. The opposite is true for existential goals, according to the second item of the corollary: it can never be bad for an agent to "undo" a norm η. Hence, an agent with only existential goals might well fear any norm η.

However, these observations implicitly assume that all agents in the system will comply with the norm. Whether they will in fact do so, of course, is a strategic decision: it partly depends on what the agent thinks that other agents will do. This motivates us to consider *normative system games*.

5. NORMATIVE SYSTEM GAMES

We now have a principled way of talking about the utility of normative systems for agents, and so we can start to apply the technical apparatus of game theory to analyse them.

Suppose we have a multi-agent system $M = \langle K, \gamma_1, \ldots, \gamma_n\rangle$ and a normative system η over K. It is proposed to the agents in M that η should be imposed on K, (typically to achieve some coordination objective). Our agent – let's say agent i – is then faced with a choice: should it *comply* with the strictures of the normative system, or not? Note that this reasoning takes place *before* the agent is "in" the system – it is a *design time* consideration.

We can understand the reasoning here as a game, as follows. A game in strategic normal form (cf. [11, p.11]) is a structure:

$$\mathcal{G} = \langle \mathcal{AG}, \mathcal{S}_1, \ldots, \mathcal{S}_n, \mathcal{U}_1, \ldots, \mathcal{U}_n\rangle \text{ where:}$$

- $\mathcal{AG} = \{1, \ldots, n\}$ is a set of agents – the players of the game;
- $\mathcal{S}_i$ is the set of strategies for each agent $i \in \mathcal{AG}$ (a strategy for an agent i is nothing else than a choice between alternative actions); and
- $\mathcal{U}_i : (\mathcal{S}_1 \times \cdots \times \mathcal{S}_n) \to \mathbb{R}$ is the utility function for agent $i \in \mathcal{AG}$, which assigns a utility to every combination of strategy choices for the agents.

Now, suppose we are given a social system $\Sigma = \langle M, \eta\rangle$ where $M = \langle K, \gamma_1, \ldots, \gamma_n\rangle$. Then we can associate a game – the *normative system game* – $\mathcal{G}_\Sigma$ with Σ, as follows. The agents $\mathcal{AG}$ in $\mathcal{G}_\Sigma$ are as in Σ. Each agent i has just two strategies available to it:

- C – *comply* (*cooperate*) with the normative system; and
- D – *do not comply with* (*defect from*) the normative system.

If S is a tuple of strategies, one for each agent, and $x \in \{C, D\}$, then we denote by $\mathcal{AG}^x_S$ the subset of agents that play strategy x in S. Hence, for a social system $\Sigma = \langle M, \eta\rangle$, the normative system $\eta \upharpoonright \mathcal{AG}^C_S$ only implements the restrictions for those agents that choose to cooperate in $\mathcal{G}_\Sigma$. Note that this is the same as $\eta \upharpoonright \mathcal{AG}^D_S$: the normative system that excludes all the restrictions of agents that play D in $\mathcal{G}_\Sigma$. We then define the utility functions U_i for each $i \in \mathcal{AG}$ as:

$$\mathcal{U}_i(S) = \delta_i(K, \eta \upharpoonright \mathcal{AG}^C_S).$$

So, for example, if S_D is a collection of strategies in which *every* agent defects (i.e., does not comply with the norm), then

$$\mathcal{U}_i(S_D) = \delta_i(K, (\eta \upharpoonright \mathcal{AG}^D_{S_D})) = u_i(K \dagger \eta_\emptyset) - u_i(K) = 0.$$

In the same way, if S_C is a collection of strategies in which *every agent cooperates* (i.e., complies with the norm), then

$$\mathcal{U}_i(S_C) = \delta_i(K, (\eta \upharpoonright \mathcal{AG}^D_{S_C})) = u_i(K \dagger (\eta \upharpoonright \emptyset)) = u_i(K \dagger \eta).$$

We can now start to investigate some properties of normative system games.

EXAMPLE 1. *(continued) For our example system, we have displayed the different $\mathcal{U}$ values for our multi agent system with the norm η_3, i.e., $\{(s,s),(t,t)\}$ as the second table of Figure 2. For instance, the pair $(0,3)$ in the matrix under the entry $S = \langle C, D\rangle$ is obtained as follows. $\mathcal{U}_1(\langle C, D\rangle) = \delta_1(K, \eta_3 \upharpoonright \mathcal{AG}^C_{\langle C,D\rangle}) = u_1(K \dagger \eta_3 \upharpoonright \mathcal{AG}^C_{\langle C,D\rangle}) - u_1(K)$. The first term of this is the utility of 1 in the system K where we implement η_3 for the cooperating agent, i.e., 1, only. This means that the transitions are $R \setminus \{(s,s)\}$. In this system, still $\varphi^1_4 = \mathsf{A}\Box\mathsf{E}\Diamond p_1$ is the highest goal for agent 1. This is the same utility for 1 as in K, and hence, $\delta_1(K, \eta_3 \upharpoonright \mathcal{AG}^C_{\langle C,D\rangle}) = 0$. Agent 2 of course benefits if agent 1 complies with η_3 while 2 does not. His utility would be 3, since $\eta_3 \upharpoonright \mathcal{AG}^C_{\langle C,D\rangle}$ is in fact η_1.*

5.1 Individually Rational Normative Systems

A normative system is individually rational if every agent would fare better if the normative system were imposed than otherwise. This is a necessary, although not sufficient condition on a norm to expect that everybody respects it. Note that η_3 of our example is individually rational for both 1 and 2, although this is not a stable situation: given that the other plays C, i is better of by playing D. We can easily characterise individually rationality with respect to the corresponding game in strategic form, as follows. Let $\Sigma = \langle M, \eta\rangle$ be a social system. Then the following are equivalent:

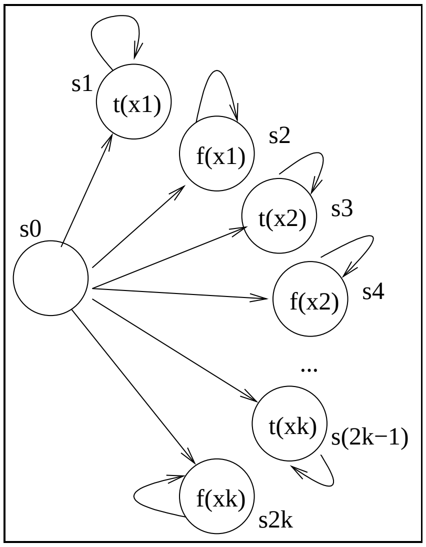

Figure 3: The Kripke structure produced in the reduction of Theorem 2; all transitions are associated with agent 1, the only initial state is s_0.

1. η is individually rational in M;
2. $\forall i \in \mathcal{AG}, \mathcal{U}_i(S_C) > \mathcal{U}_i(S_D)$ in the game $\mathcal{G}_\Sigma$.

The decision problem associated with individually rational normative systems is as follows:

> INDIVIDUALLY RATIONAL NORMATIVE SYSTEM (IRNS):
> *Given*: Multi-agent system M.
> *Question*: Does there exist an individually rational normative system for M?

THEOREM 2. IRNS *is* NP*-complete, even in one-agent systems.*

PROOF. For membership of NP, guess a normative system η, and verify that it is individually rational. Since $\eta \subseteq R$, we will be able to guess it in nondeterministic polynomial time. To verify that it is individually rational, we check that for all i, we have $u_i(K \dagger \eta) > u_i(K)$; computing $K \dagger \eta$ is just set subtraction, so can be done in polynomial time, while determining the value of $u_i(K)$ for any K can be done with a polynomial number of model checking calls, each of which requires only time polynomial in the K and γ. Hence verifying that $u_i(K \dagger \eta) > u_i(K)$ requires only polynomial time.

For NP-hardness, we reduce SAT [12, p.77]. Given a SAT instance φ over Boolean variables $x_1, \ldots, x_k$, we produce an instance of IRNS as follows. First, we define a single agent $A = \{1\}$. For each Boolean variable x_i in the SAT instance, we create two Boolean variables $t(x_i)$ and $f(x_i)$ in the IRNS instance. We then create a Kripke structure K_φ with $2k+1$ states, as shown in Figure 3: arcs in this graph correspond to transitions in K_φ. Let φ^* be the result of systematically substituting for every Boolean variable x_i in φ the CTL expression $(\mathsf{E}\bigcirc t(x_i))$. Next, consider the following formulae:

$$\bigwedge_{i=1}^{k} \mathsf{E}\bigcirc(t(x_i) \vee f(x_i)) \tag{1}$$

$$\bigwedge_{i=1}^{k} \neg((\mathsf{E}\bigcirc t(x_i)) \wedge (\mathsf{E}\bigcirc f(x_i))) \tag{2}$$

We then define the goal hierarchy for all agent 1 as follows:

$$\begin{aligned} \gamma_1[0] &= \top \\ \gamma_1[1] &= (1) \wedge (2) \wedge \varphi^* \end{aligned}$$

We claim there is an individually rational normative system for the instance so constructed iff φ is satisfiable. First, notice that any individually rational normative system must force $\gamma_1[1]$ to be true, since in the original system, we do not have $\gamma_1[1]$.

For the $\Rightarrow$ direction, if there is an individually rational normative system η, then we construct a satisfying assignment for φ by considering the arcs that are forbidden by η: formula (1) ensures that we must forbid an arc to either a $t(x_i)$ or a $f(x_i)$ state for all variables x_i, but (2) ensures that we cannot forbid arcs to both. So, if we forbid an arc to a $t(x_i)$ state then in the corresponding valuation for φ we make x_i false, while if we forbid an arc to a $f(x_i)$ state then we make x_i true. The fact that φ^* is part of the goal ensures that the normative system is indeed a valuation for φ.

For $\Leftarrow$, note that for any satisfying valuation for φ we can construct an individually rational normative system η, as follows: if the valuation makes x_i true, we forbid the arc to the $f(x_i)$ state, while if the valuation makes x_i false, we forbid the arc to the $t(x_i)$ state. The resulting normative system ensures $\gamma_1[1]$, and is thus individually rational.

Notice that the Kripke structure constructed in the reduction contains just a single agent, and so the Theorem is proven. □

5.2 Pareto Efficient Normative Systems

Pareto efficiency is a basic measure of how good a particular outcome is for a group of agents [11, p.7]. Intuitively, an outcome is Pareto efficient if there is no other outcome that makes every agent better off. In our framework, suppose we are given a social system $\Sigma = \langle M, \eta \rangle$, and asked whether η is Pareto efficient. This amounts to asking whether or not there is some other normative system η' such that every agent would be better off under η' than with η. If η' makes every agent better off than η, then we say η' *Pareto dominates* η. The decision problem is as follows:

> PARETO EFFICIENT NORMATIVE SYSTEM (PENS):
> *Given*: Multi-agent system M and normative system η over M.
> *Question*: Is η Pareto efficient for M?

THEOREM 3. PENS *is co-*NP*-complete, even for one-agent systems.*

PROOF. Let M and η be as in the Theorem. We show that the complement problem to PENS, which we refer to as PARETO DOMINATED, is NP-complete. In this problem, we are given M and η, and we are asked whether η is Pareto dominated, i.e., whether or not there exists some η' over M such that η' makes every agent better off than η. For membership of NP, simply guess a normative system η', and verify that for all $i \in A$, we have $u_i(K \dagger \eta') > u_i(K \dagger \eta)$ – verifying requires a polynomial number of model checking problems, each of which takes polynomial time. Since $\eta' \subseteq R$, the normative system can be guessed in non-deterministic polynomial time. For NP-hardness, we reduce IRNS, which we know to be NP-complete from Theorem 2. Given an instance M of IRNS, we let M in the instance of PARETO DOMINATED be as in the IRNS instance, and define the normative system for PARETO DOMINATED to be $\eta_\emptyset$, the empty normative system. Now, it is straightforward that there exists a normative system η' which Pareto dominates $\eta_\emptyset$ in M iff there exist an individually rational normative system in M. Since the complement problem is NP-complete, it follows that PENS is co-NP-complete. □

	η_0	η_1	η_2	η_3	η_4	η_5	η_6	η_7	η_8
$u_1(K \dagger \eta)$	4	4	7	6	5	0	0	8	0
$u_2(K \dagger \eta)$	4	7	4	6	0	5	8	0	0

Table 1: Utilities for all possible norms in our example

How about Pareto efficient norms for our toy example? Settling this question amounts to finding the dominant normative systems among $\eta_0 = \eta_\emptyset, \eta_1, \eta_2, \eta_3$ defined before, and $\eta_4 = \{(s,t)\}, \eta_5 = \{(t,s)\}$, $\eta_6 = \{(s,s),(t,s)\}$, $\eta_7 = \{(t,t),(s,t)\}$ and $\eta_8 = \{(s,t),(t,s)\}$. The utilities for each system are given in Table 1. From this, we infer that the Pareto efficient norms are $\eta_1, \eta_2, \eta_3, \eta_6$ and η_7. Note that η_8 prohibits the resource to be passed from one agent to another, and this is not good for any agent (since we have chosen $S^0 = \{s,t\}$, no agent can be sure to ever get the resource, i.e., goal φ_1^i is not true in $K \dagger \eta_8$).

5.3 Nash Implementation Normative Systems

The most famous solution concept in game theory is of course Nash equilibrium [11, p.14]. A collection of strategies, one for each agent, is said to form a Nash equilibrium if no agent can benefit by doing anything other than playing its strategy, under the assumption that the other agents play theirs. Nash equilibria are important because they provide stable solutions to the problem of what strategy an agent should play. Note that in our toy example, although η_3 is individually rational for each agent, it is not a Nash equilibrium, since given this norm, it would be beneficial for agent 1 to deviate (and likewise for 2). In our framework, we say a social system $\Sigma = \langle M, \eta \rangle$ (where $\eta \neq \eta_\emptyset$) is a *Nash implementation* if S_C (i.e., everyone complying with the normative system) forms a Nash equilibrium in the game $\mathcal{G}_\Sigma$. The intuition is that if Σ is a Nash implementation, then complying with the normative system is a reasonable solution for all concerned: there can be no benefit to deviating from it, indeed, there is a positive incentive for all to comply. If Σ is *not* a Nash implementation, then the normative system is unlikely to succeed, since compliance is not rational for some agents. (Our choice of terminology is deliberately chosen to reflect the way the term "Nash implementation" is used in implementation theory, or mechanism design [11, p.185], where a game designer seeks to achieve some outcomes by designing the rules of the game such that these outcomes are equilibria.)

> NASH IMPLEMENTATION (NI) :
> *Given*: Multi-agent system M.
> *Question*: Does there exist a non-empty normative system η over M such that $\langle M, \eta \rangle$ forms a Nash implementation?

Verifying that a particular social system forms a Nash implementation can be done in polynomial time – it amounts to checking:

$$\forall i \in A : u_i(K \dagger \eta) \geq u_i(K \dagger (\eta \upharpoonright \{i\})).$$

This, clearly requires only a polynomial number of model checking calls, each of which requires only polynomial time.

THEOREM 4. *The* NI *problem is* NP*-complete, even for two-agent systems.*

PROOF. For membership of NP, simply guess a normative system η and check that it forms a Nash implementation; since $\eta \subseteq R$, guessing can be done in non-deterministic polynomial time, and as we argued above, verifying that it forms a Nash implementation can be done in polynomial time.

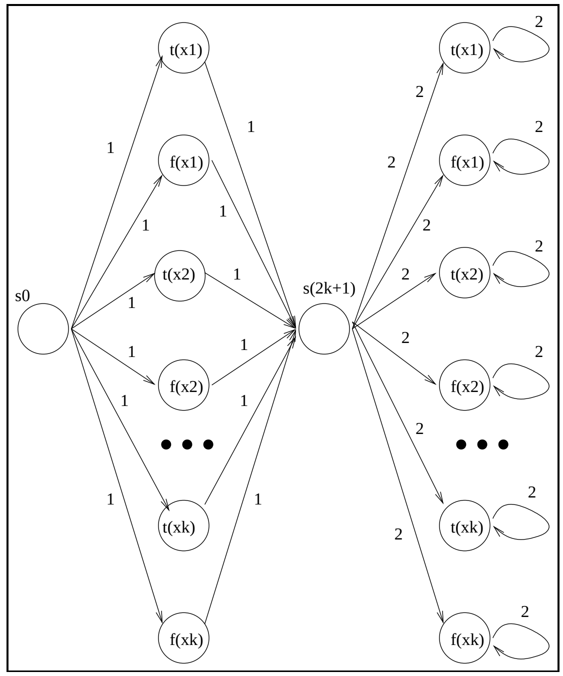

Figure 4: Reduction for Theorem 4.

For NP-hardness, we reduce SAT. Suppose we are given a SAT instance φ over Boolean variables $x_1, \ldots, x_k$. Then we construct an instance of NI as follows. We create two agents, $A = \{1,2\}$. For each Boolean variable x_i we create two Boolean variables, $t(x_i)$ and $f(x_i)$, and we then define a Kripke structure as shown in Figure 4, with s_0 being the only initial state; the arc labelling in Figure 4 gives the α function, and each state is labelled with the propositions that are true in that state. For each Boolean variable x_i, we define the formulae $x_i^\top$ and $x_i^\bot$ as follows:

$$\begin{aligned} x_i^\top &= \mathsf{E}\bigcirc(t(x_i) \wedge \mathsf{E}\bigcirc((\mathsf{E}\bigcirc(t(x_i))) \wedge \mathsf{A}\bigcirc(\neg f(x_i)))) \\ x_i^\bot &= \mathsf{E}\bigcirc(f(x_i) \wedge \mathsf{E}\bigcirc((\mathsf{E}\bigcirc(f(x_i))) \wedge \mathsf{A}\bigcirc(\neg t(x_i)))) \end{aligned}$$

Let φ^* be the formula obtained from φ by systematically substituting $x_i^\top$ for x_i. Each agent has three goals: $\gamma_i[0] = \top$ for both $i \in \{1,2\}$, while

$$\begin{aligned} \gamma_1[1] &= \bigwedge_{i=1}^{k}((\mathsf{E}\bigcirc(t(x_i))) \wedge (\mathsf{E}\bigcirc(f(x_i)))) \\ \gamma_2[1] &= \mathsf{E}\bigcirc\mathsf{E}\bigcirc\bigwedge_{i=1}^{k}((\mathsf{E}\bigcirc(t(x_i))) \wedge (\mathsf{E}\bigcirc(f(x_i)))) \end{aligned}$$

and finally, for both agents, $\gamma_i[2]$ being the conjunction of the following formulae:

$$\bigwedge_{i=1}^{k}(x_i^{\top} \vee x_i^{\perp}) \tag{3}$$

$$\bigwedge_{i=1}^{k}\neg(x_i^{\top} \wedge x_i^{\perp}) \tag{4}$$

$$\bigwedge_{i=1}^{k}\neg(\mathsf{E}\bigcirc(t(x_i)) \wedge \mathsf{E}\bigcirc(f(x_i))) \tag{5}$$

$$\varphi^{*} \tag{6}$$

We denote the multi-agent system so constructed by M_φ. Now, we prove that the SAT instance φ is satisfiable iff M_φ has a Nash implementation normative system:

For the $\Rightarrow$ direction, suppose φ is satisfiable, and let X be a satisfying valuation, i.e., a set of Boolean variables making φ true. We can extract from X a Nash implementation normative system η as follows: if $x_i \in X$, then η includes the arc from s_0 to the state in which $f(x_i)$ is true, and also includes the arc from $s(2k+1)$ to the state in which $f(x_i)$ is true; if $x_i \notin X$, then η includes the arc from s_0 to the state in which $t(x_i)$ is true, and also includes the arc from $s(2k+1)$ to the state in which $t(x_i)$ is true. No other arcs, apart from those so defined, as included in η. Notice that η is individually rational for both agents: if they both comply with the normative system, then they will have their $\gamma_i[2]$ goals achieved, which they do not in the basic system. To see that η forms a Nash implementation, observe that if *either* agent defects from η, then *neither* will have their $\gamma_i[2]$ goals achieved: agent 1 strictly prefers (C, C) over (D, C), and agent 2 strictly prefers (C, C) over (C, D).

For the $\Leftarrow$ direction, suppose there exists a Nash implementation normative system η, in which case $\eta \neq \emptyset$. Then φ is satisfiable; for suppose not. Then the goals $\gamma_i[2]$ are not achievable by any normative system, (by construction). Now, since η must forbid at least one transition, then at least one agent would fail to have its $\gamma_i[1]$ goal achieved if it complied, so at least one would do better by defecting, i.e., not complying with η. But this contradicts the assumption that η is a Nash implementation, i.e., that (C, C) forms a Nash equilibrium. $\square$

This result is perhaps of some technical interest beyond the specific concerns of the present paper, since it is related to two problems that are of wider interest: the complexity of mechanism design [5], and the complexity of computing Nash equilibria [6, 7]

5.4 Richer Goal Languages

It is interesting to consider what happens to the complexity of the problems we consider above if we allow richer languages for goals: in particular, CTL* [9]. The main difference is that determining $u_i(K)$ in a given multi-agent system M when such a goal language is used involves solving a PSPACE-complete problem (since model checking for CTL* is PSPACE-complete [8]). In fact, it seems that for each of the three problems we consider above, the corresponding problem under the assumption of a CTL* representation for goals is also PSPACE-complete. It cannot be any easier, since determining the utility of a particular Kripke structure involves solving a PSPACE-complete problem. To see membership in PSPACE we can exploit the fact that PSPACE = NPSPACE [12, p.150], and so we can "guess" the desired normative system, applying a PSPACE verification procedure to check that it has the desired properties.

6. CONCLUSIONS

Social norms are supposed to restrict our behaviour. Of course, such a restriction does not have to be bad: the fact that an agent's behaviour is restricted may seem a limitation, but there may be benefits if he can assume that others will also constrain their behaviour. The question then, for an agent is, how to be sure that others will comply with a norm. And, for a system designer, how to be sure that the system will behave socially, that is, according to its norm. Game theory is a very natural tool to analyse and answer these questions, which involve strategic considerations, and we have proposed a way to translate key questions concerning logic-based normative systems to game theoretical questions. We have proposed a logical framework to reason about such scenarios, and we have given some computational costs for settling some of the main questions about them. Of course, our approach is in many senses open for extension or enrichment. An obvious issue is to consider is the complexity of the questions we give for more practical representations of models (cf. [1]), and to consider other classes of allowable goals.

7. REFERENCES

[1] T. Agotnes, W. van der Hoek, J. A. Rodriguez-Aguilar, C. Sierra, and M. Wooldridge. On the logic of normative systems. In *Proc. IJCAI-07*, Hyderabad, India, 2007.

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

[3] K. Binmore. *Game Theory and the Social Contract Volume 1: Playing Fair*. The MIT Press: Cambridge, MA, 1994.

[4] K. Binmore. *Game Theory and the Social Contract Volume 2: Just Playing*. The MIT Press: Cambridge, MA, 1998.

[5] V. Conitzer and T. Sandholm. Complexity of mechanism design. In *Proc. UAI*, Edmonton, Canada, 2002.

[6] V. Conitzer and T. Sandholm. Complexity results about nash equilibria. In *Proc. IJCAI-03*, pp. 765–771, Acapulco, Mexico, 2003.

[7] C. Daskalakis, P. W. Goldberg, and C. H. Papadimitriou. The complexity of computing a Nash equilibrium. In *Proc. STOC*, Seattle, WA, 2006.

[8] E. A. Emerson. Temporal and modal logic. In *Handbook of Theor. Comp. Sci. Vol. B*, pages 996–1072. Elsevier, 1990.

[9] E. A. Emerson and J. Y. Halpern. 'Sometimes' and 'not never' revisited: on branching time versus linear time temporal logic. *Jnl. of the ACM*, 33(1):151–178, 1986.

[10] D. Fitoussi and M. Tennenholtz. Choosing social laws for multi-agent systems: Minimality and simplicity. *Artificial Intelligence*, 119(1-2):61–101, 2000.

[11] M. J. Osborne and A. Rubinstein. *A Course in Game Theory*. The MIT Press: Cambridge, MA, 1994.

[12] C. H. Papadimitriou. *Computational Complexity*. Addison-Wesley: Reading, MA, 1994.

[13] Y. Shoham and M. Tennenholtz. On the synthesis of useful social laws for artificial agent societies. In *Proc. AAAI*, San Diego, CA, 1992.

[14] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. In *Computational Theories of Interaction and Agency*, pages 597–618. The MIT Press: Cambridge, MA, 1996.

[15] W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: Effectiveness, feasibility, and synthesis. *Synthese*, 2007.

[16] M. Wooldridge and W. van der Hoek. On obligations and normative ability. *Jnl. of Appl. Logic*, 3:396–420, 2005.