Knowledge and Social Laws

Wiebe van der Hoek   Mark Roberts   Michael Wooldridge
Department of Computer Science
University of Liverpool
Liverpool L69 7ZF, UK
{wiebe,mark,mjw}@csc.liv.ac.uk

ABSTRACT

In this paper we combine existing work in the area of *social laws* with a framework for reasoning about *knowledge* in multi-agent systems. The unifying framework in which this is done is based on Alternating-time Temporal Logic (ATL), to which semantics we add epistemic accessibility relations (to deal with the knowledge), actions (in order to naturally talk about allowed and forbidden actions) and updates (to model the effect of the implementation of the constraint in a social law). Apart from a constraint, a social law has an objective: in our formalism, such objectives may refer to the knowledge that agents possess or do not possess. The result is a framework in which we can, for example, express that a desirable property (objective) of a social law is that one agent has the ability to bring about a certain type of knowledge in another agent, or that if one agent knows something, then it should behave in a certain way. We illustrate our approach with a case study, and we use model checking to demonstrate that properties of social laws with respect to this case study.

Categories and Subject Descriptors

I.2.4 [**Artificial Intelligence**]: Knowledge Representation Formalisms and Methods—*Representation languages, Modal logic, Temporal logic*; I.2.11 [**Artificial Intelligence**]: Distributed Artificial Intelligence—*Coherence and coordination, Multiagent systems*

General Terms

Theory, Design, Verification

Keywords

Social Laws, Alternating-time Temporal Epistemic Logic, Knowledge

1. INTRODUCTION

A *social law* is a restriction on the behaviour of agents, to ensure they can work individually in a mutually compatible manner in order to fulfil their individual goals, and possibly joint goals. Social laws work by prohibiting the performance of certain actions in certain situations (states). The social laws paradigm was first proposed by Moses, Shoham and Tennenholtz [11, 10, 12, 8, 13, 14].

In [16], van der Hoek et al extend the social laws framework further. They show that *Alternating-time Temporal Logic* (ATL) provides a rich and natural technical framework within which to investigate social laws and their properties. In this framework, a social law consists of two parts: an objective of the law, written as an ATL specification, and a behavioural constraint, which is a function that for every action returns the set of states where that action is forbidden from being performed. The objective of the law represents what the society aims to achieve by adopting the law, and the behavioural constraint corresponds to the requirements that the law places on the members of society.

With social laws expressed in this way, the problem of determining whether a social law is effective corresponds to an ATL model checking problem. This is advantageous as it allows the use of existing ATL model checkers. Expressing social laws in ATL in this way clearly has other advantages to it. Social laws frequently refer to the *powers* or *rights* (or, conversely, the limits to powers) that agents have, so ATL, with its cooperation modalities, is a very natural choice to express such laws. Also, ATL has the ability to express laws that refer to conditions that must hold over time. Finally, [16] studies various computational problems in their new framework of social laws, and found the complexity of the feasibility problem for social laws in the most general case to be no more complex than the corresponding problem in the Shoham-Tennenholtz framework (it is NP-complete).

However, the framework of [16] is based on quite a strong and unrealistic assumption that agents know everything about the state of the system. This is modelled with a global state space which all agents have access to. In this paper we do away with the need for this assumption by extending the framework further by incorporating the notion of knowledge. Knowledge is important in any non-trivial multi-agent system, and as knowledge will come to play in almost any coordination scenario, we can clearly see a need to extend our framework. Not only will the agents' knowledge of the system be used in order to follow certain laws, but we can have laws in which certain information must become known (or remain private) to an agent or a coalition of agents.

So, by imposing certain social laws, we can force desirable knowledge properties to hold.

Furthermore, we can have laws which are only known to a certain agent or group of agents, and only these agents need follow the law. In this paper, we demonstrate how to 'transform' the objective of a social law in several, individual objectives, which have a feasible format: depending on the information the agent has, he is required to achieve some situation that is under his control.

This paper is structured as follows: We begin by introducing Action-based Alternating Epistemic Transition Systems (AAETSs), based on an essentially equivalent version of the structures of [16] but extended in order to represent the knowledge of the agents; we then introduce Alternating Temporal Epistemic Logic (ATEL) itself, the logic we will use to express social laws. In section 4 we formally define this new extended framework of social laws with respect to AAETS and ATEL; and then in section 5 we present a case study to illustrate the power of our framework and to show how model checking can be performed using standard ATL model checkers. A discussion of related work is presented in section 6, while section 7 concludes.

2. SEMANTIC STRUCTURES

In [16] the semantic structures used are known as *Action-based Alternating Transition Systems* (AATSs) and are essentially equivalent to the alternating transition systems used by Alur and colleagues to give semantics to ATL (see [2]). In this section we extend AATSs into what we call *Action-based Alternating Epistemic Transition Systems* (AAETSs). The main difference is that we introduce an *epistemic accessibility relation* for each agent in the system, which is used to capture each agent's knowledge. The approach is standard in the literature of epistemic logic [5, 7].

We first assume that the systems of interest to us may be in any of a finite set Q of possible *states*, with some $q_0 \in Q$ designated as the *initial state*. Systems are populated by a set Ag of *agents*; a *coalition* of agents is simply a set $G \subseteq Ag$, and the set of all agents is known as the *grand coalition*. Each agent $i \in Ag$ is associated with a set Ac_i of possible actions, and we assume that these sets of actions are pairwise disjoint. We denote the set of actions associated with a coalition $G \subseteq Ag$ by Ac_G, so $Ac_G = \bigcup_{i \in G} Ac_i$. A *joint action* for a coalition G is an indexed tuple $\langle \alpha_1, \ldots, \alpha_k \rangle$, where $\alpha_i \in Ac_i$, for each $i \in G$. We denote the set of all joint actions for coalition G by J_G, so $J_G = \prod_{i \in G} Ac_i$. Given an element j of J_G and agent $i \in G$, we denote i's component of j by j_i. Now, for each agent $i \in Ag$, we also have an epistemic accessibility relation $\sim_i$, which represents indistinguishable states to agent i, used to capture i's knowledge. Thus, an AAETS is an $(2n + 7)$-tuple

$$\langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho, \tau, \Phi, \pi \rangle$$

where:

- Q is a finite, non-empty set of *states*;

- $q_0 \in Q$ is the *initial state*;

- $Ag = \{1, \ldots, n\}$ is a finite, non-empty set of *agents*;

- Ac_i is a finite, non-empty set of *actions*, for each $i \in Ag$, where $Ac_i \cap Ac_j = \emptyset$ for all $i \neq j \in Ag$;

- $\sim_i \subseteq Q \times Q$ is an epistemic accessibility relation for each agent $i \in Ag$. Each $\sim_i$ must be an equivalence relation;

- $\rho : Ac_{Ag} \to 2^Q$ is an *action precondition function*, which for each action $\alpha \in Ac_{Ag}$ defines the set of states $\rho(\alpha)$ from which α may be executed;

- $\tau : Q \times J_{Ag} \to Q$ is a partial *system transition function*, which defines the state $\tau(q, j)$ that would result by the performance of j from state q;

- Φ is a finite, non-empty set of *atomic propositions*; and

- $\pi : Q \to 2^\Phi$ is an interpretation function, which gives the set of primitive propositions satisfied in each state: if $p \in \pi(q)$, then proposition p is true in state q.

AAETSs must satisfy two coherence constraints:

1. *Non-triviality [8].*

   Agents always have at least one legal action: $\forall q \in Q, \forall i \in Ag, \exists \alpha \in Ac_i$ s.t. $q \in \rho(\alpha)$

2. *Consistency.*

   The ρ and τ functions agree on actions that may be performed: $\forall q, \forall j \in J_{Ag}, (q, j) \in \text{dom } \tau$ iff $\forall i \in Ag, q \in \rho(j_i)$

We denote the set of sequences over Q by Q^*, and the set of non-empty sequences over Q by Q^+.

For a coalition of agents $G \subseteq Ag$, we denote the union of G's accessibility relations by $\sim_G^E$, so $\sim_G^E = \left(\bigcup_{i \in G} \sim_i \right)$. The transitive closure of $\sim_G^E$ is denoted by $\sim_G^C$.

Given an agent $i \in Ag$ and a state $q \in Q$, we denote the *options* available to i in q by $options(i, q) = \{\alpha \mid \alpha \in Ac_i$ and $q \in \rho(\alpha)\}$. We then say that a *strategy* for an agent $i \in Ag$ is a function: $\sigma_i : Q \to Ac_i$ which must satisfy the *legality* constraint that $\sigma_i(q) \in options(i, q)$ for all $q \in Q$.

A *strategy profile* for a coalition $G = \{a_1, \ldots, a_k\} \subseteq Ag$ is a tuple of strategies $\langle \sigma_1, \ldots, \sigma_k \rangle$, one for each agent $a_i \in G$. We denote by Σ_G the set of all strategy profiles for coalition $G \subseteq Ag$; if $\sigma_G \in \Sigma_G$ and $i \in G$, then we denote i's component of σ_G by σ_G^i. Given a strategy profile $\sigma_G \in \Sigma_G$ and state $q \in Q$, let $out(\sigma_G, q)$ denote the set of possible states that may result by the members of the coalition G acting as defined by their components of σ_G for one step from q. Formally, $out(\sigma_G, q) =$

$$\{q' \mid \tau(q, j) = q' \text{ where } \sigma_G^i(q) = j_i \text{ for } i \in G\}$$

Notice that, for any grand coalition strategy profile σ_{Ag} and state q, the set $out(\sigma_{Ag}, q)$ will be singleton.

A *computation* λ is an infinite sequence of states $\lambda = q_0, q_1, \ldots$. A computation $\lambda \in Q^+$ starting in state q is referred to as a *q-computation*; if $u \in \mathbb{N}$, then we denote by

$\lambda[u]$ the component indexed by u in λ (thus $\lambda[0]$ denotes the first element, $\lambda[1]$ the second, and so on).

Given a strategy profile σ_G for some coalition G, and a state $q \in Q$, we define $comp(\sigma_G, q)$ to be the set of possible runs that may occur if every agent $a_i \in G$ follows the corresponding strategy σ_i, starting when the system is in state $q \in Q$. Formally, $comp(\sigma_G, q) =$

$$\{\lambda \mid \lambda[0] = q \text{ and } \forall u \in \mathbb{N} : \lambda[u+1] \in out(\sigma_G, \lambda[u])\}.$$

3. ATEL

In this section we define the language used to express social laws in our framework. Alternating temporal epistemic logic (ATEL) [18] is an extension of the well known ATL of Alur, Henzinger, and Kupferman, which in turn can be understood as a generalisation of the branching time temporal logic CTL. As with ATL, we have cooperation modalities, such as, $\langle\!\langle G \rangle\!\rangle \varphi$, which expresses that the coalition G can cooperate to ensure φ; more precisely, that there exists a strategy profile for G such that by following this strategy profile, G can ensure φ.

The "$\square$" temporal operator means "now and forever more": additional temporal connectives in ATEL are "$\Diamond$" ("either now or at some point in the future"), "$\mathcal{U}$" ("until"), and "$\bigcirc$" ("in the next state"). Additionally, we have the following knowledge operators: $K_i \varphi$ ("agent i knows φ"), $E_G \varphi$ ("everyone in coalition G knows φ"), and $C_G \varphi$ ("it is common knowledge to everyone in the coalition G that φ").

Formally, the set of ATEL formulae, formed with respect to a set of agents Ag, and a set of primitive propositions Φ, is given by the following grammar:

$$\varphi \quad ::= \quad p \mid \neg\varphi \mid \varphi \vee \varphi \mid K_i\varphi \mid E_G\varphi \mid C_G\varphi \mid$$
$$\langle\!\langle G \rangle\!\rangle \bigcirc \varphi \mid \langle\!\langle G \rangle\!\rangle \square \varphi \mid \langle\!\langle G \rangle\!\rangle \varphi \mathcal{U} \varphi$$

where $p \in \Phi$ is a propositional variable, $G \subseteq Ag$ is a set of agents, and $i \in Ag$ is an agent.

We now give the truth definition of ATEL formulae on an AAETS S and a state q:

$S, q \models p$ iff $p \in \pi(q)$ $\quad$ (where $p \in \Phi$);

$S, q \models \neg\varphi$ iff $S, q \not\models \varphi$;

$S, q \models \varphi \vee \psi$ iff $S, q \models \varphi$ or $S, q \models \psi$;

$S, q \models \langle\!\langle G \rangle\!\rangle \bigcirc \varphi$ iff $\exists \sigma_G \in \Sigma_G$, such that $\forall \lambda \in comp(q, \sigma_G)$, we have $S, \lambda[1] \models \varphi$;

$S, q \models \langle\!\langle G \rangle\!\rangle \square \varphi$ iff $\exists \sigma_G \in \Sigma_G$, such that $\forall \lambda \in comp(q, \sigma_G)$, we have $S, \lambda[u] \models \varphi$ for all $u \in \mathbb{N}$;

$S, q \models \langle\!\langle G \rangle\!\rangle \varphi \mathcal{U} \psi$ iff $\exists \sigma_G \in \Sigma_G$, such that $\forall \lambda \in comp(q, \sigma_G)$, there exists some $u \in \mathbb{N}$ such that $S, \lambda[u] \models \psi$, and for all $0 \leq v < u$, we have that $S, \lambda[v] \models \varphi$.

$S, q \models K_i \varphi$ iff for all q' such that $q \sim_i q' : S, q' \models \varphi$;

$S, q \models E_G \varphi$ iff for all q' such that $q \sim_G^E q' : S, q' \models \varphi$;

$S, q \models C_G \varphi$ iff for all q' such that $q \sim_G^C q' : S, q' \models \varphi$.

The other connectives ("$\wedge$", "$\rightarrow$", "$\leftrightarrow$") are assumed to be defined as abbreviations in terms of $\neg$, $\vee$. Also, $M_i\varphi$ means $\neg K_i \neg\varphi$, while $\langle\!\langle G \rangle\!\rangle \Diamond \varphi$ is shorthand for $\neg\langle\!\langle G \rangle\!\rangle \square \neg\varphi$. We write $\langle\!\langle i \rangle\!\rangle$ rather than $\langle\!\langle \{i\} \rangle\!\rangle$. The cooperation modality $\langle\!\langle\,\rangle\!\rangle$

asserts that its argument is true on all computations, and thus acts like CTL's universal path quantifier "A". Similarly, $\langle\!\langle Ag \rangle\!\rangle$ asserts that its argument is satisfied on at least one computation, and thus acts like the CTL path quantifier "E".

We now briefly mention two properties of ATEL. For any

$$S = \langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho, \tau, \Phi, \pi \rangle,$$

let the relation R_{Ag} between states in Q be defined as:

$$q_1 R_{Ag} q_2 \text{ iff } \exists j \in J_{Ag} : \tau(q_1, j) = q_2.$$

In other words, $q_1 R_{Ag} q_2$ holds if the grand coalition can enforce a transition from q_1 to q_2. Let

$$S = \langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho, \tau, \Phi, \pi \rangle$$

and

$$S' = \langle Q', q_0', Ag, Ac_1', \ldots, Ac_n', \sim_1', \ldots, \sim_n', \rho', \tau', \Phi, \pi' \rangle$$

be two AAETSs. We say that S' is a *subsystem* of S, (notation $S' \sqsubseteq S$) if there exists a relation

$$\Re \subseteq Q \times Q'$$

such that:

1. $q_0 \Re q_0'$

2. $\forall q \in Q, \ q' \in Q' : q \Re q' \Rightarrow (\pi(q) = \pi(q'))$

3. $\forall q_1 \in Q, \ q_1', q_2' \in Q' :$
   $((q_1 \Re q_1' \ \& \ q_1' R_{Ag}' q_2') \Rightarrow (\exists q_2 \in Q : \ q_1 R_{Ag} q_2 \ \& \ q_2 \Re q_2'))$

The relation $\Re$ is essentially half of a *bisimulation* between two Kripke models [3, p.64]: the final clause represents the "backward clause" for such relations, with the "forward clause" not being present in the conditions on $\Re$. We are interested in which formulas are preserved when going from S to S', where $S' \sqsubseteq S$. To this end, we define a *universal* and an *existential* sublanguage of ATEL, denoted $\mathcal{L}^u$ and $\mathcal{L}^e$, respectively. These languages are defined by the following grammars ($p \in \Phi$):

$$\mathcal{L}^u \quad \upsilon ::= \quad p \mid \neg p \mid \upsilon \wedge \upsilon \mid \upsilon \vee \upsilon \mid K_i\upsilon \mid E_G\upsilon \mid C_G\upsilon \mid$$
$$\langle\!\langle\,\rangle\!\rangle \bigcirc \upsilon \mid \langle\!\langle\,\rangle\!\rangle \Diamond \upsilon \mid \langle\!\langle\,\rangle\!\rangle \square \upsilon \mid \langle\!\langle\,\rangle\!\rangle \upsilon \mathcal{U} \upsilon$$

$$\mathcal{L}^e \quad \epsilon ::= \quad p \mid \neg p \mid \epsilon \wedge \epsilon \mid \epsilon \vee \epsilon \mid M_i\epsilon \mid \langle\!\langle Ag \rangle\!\rangle \bigcirc \epsilon \mid$$
$$\langle\!\langle Ag \rangle\!\rangle \Diamond \epsilon \mid \langle\!\langle Ag \rangle\!\rangle \square \epsilon \mid \langle\!\langle Ag \rangle\!\rangle \epsilon \mathcal{U} \epsilon$$

We now get the following result, which basically says that existential formulas are preserved when moving to a larger structure, whereas universal are preserved when restricting the structure.

LEMMA 1. *Let* $S' \sqsubseteq S, \upsilon \in \mathcal{L}^u, \epsilon \in \mathcal{L}^e$. *Then:*

$$S', q \models \epsilon \Rightarrow S, q \models \epsilon \text{ and } S, q \models \upsilon \Rightarrow S', q \models \upsilon$$

4. KNOWLEDGE AND SOCIAL LAWS

We now present our formal framework of social laws and demonstrate how knowledge is incorporated into the framework by investigating various knowledge properties. As in [16], a social law consists of two parts:

1. An *objective*, which represents what the society aims to achieve by the introduction, or adoption of the law. This objective can now refer to knowledge. In human societies, for example, the objective of a law might be to ensure that bank card PIN numbers remain private to the card holder.

2. A *behavioural constraint*, which corresponds to the requirements that a law places on the members of a society. A behavioural constraint corresponding to the objective of ensuring PIN numbers remain private might be to forbid the action of reading another person's PIN number.

Now we represent an objective for a social law as a formula of ATEL, with the same intuition that a social law is *effective* if it ensures that the objective is satisfied. ATEL inherits all the properties of ATL, allowing us to reason about what certain coalitions can bring about, as well as the ability to express liveness and safety properties from CTL, but now we can also express knowledge properties in order to reason about what agents and coalitions of agents should and shouldn't know.

As in previous work, we model a behavioural constraint, β, as a function $\beta : Ac_{Ag} \to 2^Q$, with the intended interpretation that if $q \in \beta(\alpha)$, then action α may *not* be performed when the system is in state q – that is, α is "forbidden" in state q. We require that any behavioural constraint is "reasonable", in that it always permits an agent to have at least one action left that can be performed in any state:

$$\forall\, i \in Ag, \forall q \in Q, \exists \alpha \in options(i, q) \text{ s.t. } q \notin \beta(\alpha).$$

We can now see what it means for a behavioural constraint β to be *implemented* in an AAETS S: it means eliminating from S all transitions that are forbidden by β. The operation of implementing a behavioural constraint is thus an *update* on AAETSs, in the sense that it results in a new AAETS, which potentially satisfies different formulae. We denote the AAETS obtained from S by implementing β by $S \dagger \beta$. Formally, if

$$S = \langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho, \tau, \Phi, \pi \rangle$$

is an AAETS, and β is a behavioural constraint on S, then $S \dagger \beta =$

$$\langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho', \tau', \Phi, \pi \rangle,$$

where:

1. $\forall \alpha \in Ac$: $\quad \rho'(\alpha) = \rho(\alpha) \setminus \beta(\alpha)$

2. $\forall q \in Q, \forall j \in J_{Ag}$: $\quad \tau'(q, j) =$
   $$\begin{cases} \tau(q, j) & \text{if } (q, j) \in \text{dom } \tau \text{ and } \forall i \in Ag, q \notin \beta(j_i) \\ \text{undefined} & \text{otherwise} \end{cases}$$

3. All other components of $S \dagger \beta$ are as in S.

A *social law* over AAETS S is then a pair (φ, β) where:

- φ is an ATEL formula called the *objective* of the law;

- $\beta : Ac_{Ag} \to 2^Q$ is a *behavioural constraint* on S.

The *feasibility* problem for social laws as defined in [16] is as follows:

> Given an AAETS S and a formula φ, intended to represent an objective, does there exist a behavioural constraint β that is effective for φ in S, i.e., such that $S \dagger \beta \models \langle\langle\rangle\rangle \square \varphi$.

It was proved in [16] that, for objectives expressed in ATL, the feasibility problem is NP-complete. We now show that, for ATEL objectives, the problem is no worse.

THEOREM 1. *The feasibility problem for objectives expressed as arbitrary ATEL formulae is NP-complete.*

PROOF. Since feasibility for ATEL objectives subsumes feasibility for ATL objectives, we only need to prove the upper bound. This follows from the fact that we can guess a behavioural constraint β, which will be polynomial in the size of $S = \langle Q, q_0, Ag, Ac_1, \ldots, Ac_n, \sim_1, \ldots, \sim_n, \rho, \tau, \Phi, \pi \rangle$, and then verify that $S \dagger \beta \models \langle\langle\rangle\rangle \square \varphi$. The process of generating $S \dagger \beta$ can easily be done in time polynomial in the size of S (we simply delete from S all transitions forbidden by β), and since model checking for ATEL is in P, the result follows. $\square$

4.1 Knowledge Properties

There are certain knowledge properties that are of interest to us. A knowledge pre-condition is the information that an agent must have in order to be able to successfully carry out an action or plan. Knowledge pre-condition formulae take the form $K_i\varphi \to \psi$. This formula states that agent i knowing the fact φ implies that the fact (state of affairs) ψ should hold. These knowledge pre-condition formulae can be used as ATEL objectives in our social laws. Let us define $[\![i]\!]\varphi$ as the dual of $\langle\langle i \rangle\rangle\varphi$, so that $[\![i]\!]\varphi$ means that i cannot prevent the others to achieve φ (whichever strategy i choses, the others can complement it in such a way that φ).

Then, a property like

$$K_i\varphi \to [\![i]\!](\bigcirc \psi \vee \bigcirc\bigcirc \psi \vee \bigcirc\bigcirc\bigcirc \psi) \qquad (1)$$

expresses that, if the gate-controller (i) on a crossing knows that the train is waiting to cross (φ), then, whatever the gates do, the train should be able to safely cross (ψ) in the 'near future' (i.e., within the next three time steps). We can extend this example immediately to cases where ψ is itself epistemic: if the train (j) knows that the constraint (1) applies, and he wants to safely cross, he should notify the the gate-controller of this:

$$K_j\varphi \to \langle\langle j \rangle\rangle \bigcirc K_i\varphi \qquad (2)$$

Implied knowledge is knowledge that comes about as a result of the performance of actions, which lead to a change in the current state of affairs: they present themselves as $\varphi \to K_i\psi$. This formula states that the state of affairs φ should bring about the knowledge of the fact ψ to agent i.

This type of formulae can also be used in the objectives to our social laws. For example, the following could express that everytime that train i is able to pass a crossing, the gate at that crossing should know it.

$$\langle\langle i \rangle\rangle \bigcirc \psi \to K_j\varphi \qquad (3)$$

Finally, nested knowledge is information that the agents have about what other agents know. For instance, $[\![i]\!]\diamondsuit\psi \to K_i E_G \varphi$ expresses that if i cannot ensure that ψ will ever become true, he should make sure (know) that everybody in G knows this. Nested knowledge can also play the role of a pre-condition (I should know that you know we have a meeting at the station, before me going there, for instance).

5. A CASE STUDY

In this scenario, there is a train on a circular track, which at one point crosses a road (see Figure 1). The place where the track crosses the road (level crossing) is controlled by gates operated by a gate controller agent. If the train moves on to the level crossing while the gates are closed, the train will be in a crash situation, in which it has crashed into a car on the crossing. Also, if the gates close while the train is on the crossing, a crash will occur. We are interested in social laws that can prevent such situations from arising.

This system consists of two agents called t and g, where t is the agent representing the train and g is the gate controller agent. The train can be in one of four states, "staA" (train is at station A, which is the initial state of the train), "staB" (train stopped at station B), "wait" (waiting to use the crossing) and "oncr" (the train is on the crossing). The gates can only be in two states, either "open" (gates are open to the train) or "closed" (gates are closed to the train).

There is a communication medium by which the train and gate controller agent can exchange messages about their local states. The communication works by signals sent and received via an aerial on each of the agents. Signals received from the gates controller agent are stored in a local variable of the train called $Flag_g$. Similarly, signals received from the train are stored in a local variable of the gate controller agent called $Flag_t$.

The train has six actions available to it: $move_t$, $moveann_t$, $tell\text{-}away_t$, $tell\text{-}wait_t$, $tell\text{-}oncr_t$, and $idle_t$. The $idle_t$ action is the identity which causes no change in the train's state. If the train executes a $move_t$ action while it is at $staA$, then it goes to $staB_t$; executing $move_t$ while at $staB_t$ causes a transition to the $wait_t$ state; executing $move_t$ while $wait_t$ causes a transition to the $oncr_t$ state; and, finally, executing $move_t$ while $oncr_t$ causes a transition to $staA_t$, as long as the gates were not in the $closed_g$ state at the same time as the train was in the $oncr_t$ state, as, if this is the case, the train is said to have crashed and is forced to $idle_t$ indefinitely.

The $moveann_t$ action causes the same state transitions as the $move_t$ action, but additionally the train sends its new location information to g. The actions prefixed with $tell$ have the effect of telling the gates that the location of the train is $away_t$, $wait_t$, and $oncr_t$ respectively. Note that, without any social law in place, it is not guaranteed that the train tells his location *truthfully* (e.g. performing $tell\text{-}wait_t$ when actually in state $oncr_t$)!

The gates controller has five actions available to it: $gates_g$, $gatesann_g$, $tell\text{-}open_g$, $tell\text{-}closed_g$, and $idle_g$. As with the train, this $idle_g$ action causes no change in the state of the gates. The $gates_g$ action causes the position of the gates to be toggled, i.e., performing $gates_g$ when the gates are $closed_g$ will result in the gates being $open_g$ and vice versa. The $gatesann_g$ action causes the same transitions as the $gates_g$ action, but the gate controller agent sends its new status to the train agent. The actions prefixed with $tell$ have the effect of telling the train that the new status of the gates is $closed_g$ and $open_g$ respectively.

A state in our system is defined to be a tuple

$$\langle Loc_t, Flag_g, Status_g, Flag_t \rangle$$

where:

- $Loc_t \in \{staA_t, staB_t, wait_t, oncr_t\}$

- $Flag_g \in \{true, false\}$

- $Status_g \in \{open_g, closed_g\}$

- $Flag_t \in \{a, w, o\}$

We assume that we have atoms in the object language (like $Flag_t = w$) corresponding to these states. We will refer to this tuple generally as $\langle W, X, Y, Z \rangle$. We let q_i denote the ith component of q. Now, $\langle q_1, q_2, q_4 \rangle$ is the local state of agent t and $\langle q_2, q_3, q_4 \rangle$ is the local state of agent g. The initial state (or root) of our system is $\rho = \langle staA_t, false, closed_g, a \rangle$.

Given two states $q, r \in Q$, then the epistemic accessibility relation for agent t is given as: $q \sim_t r$, iff $q_1 = r_1$, $q_2 = r_2$ and $q_4 = r_4$. The epistemic accessibility relation for agent g is given as: $q \sim_g r$, iff $q_2 = r_2$, $q_3 = r_3$ and $q_4 = r_4$. Let us call this overall system S. The transitions are given in Figure 5, where the first line for instance represents that a $move_t$ in any state $\langle staA_t, X, Y, Z \rangle$, when done jointly with any action ac_g of the gates, results in any of $\langle staB_t, X', Y', Z \rangle$ states, where the precise X' and Y' values (denoting the state of the gates), depend on X, Y, and ac_g, the location of the train changes from station A to station B, and the value of $Flag_t$, stored in the variable Z, remains the same.
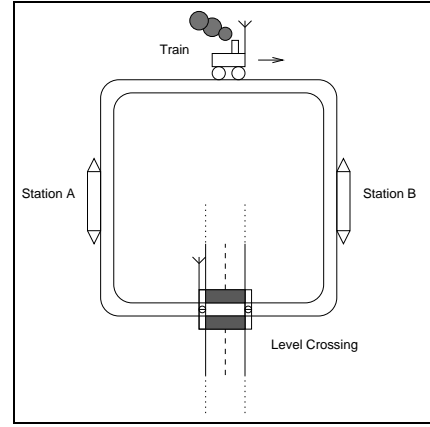

Figure 1: The Level Crossing system.

5.1 Epistemic Social Laws

Now we can formulate some objectives for social laws. They have the general format $K_i\varphi \to \psi$, where ψ is some property under control of agent i. This format makes perfect sense: (only) if an agent knows certain preconditions, can we require from him to take appropriate action. Let us hence say that such an objective is *feasible for agent i*.

$$O_1 = K_t wait_t \to (Flag_t = w) \qquad (4)$$

The objective O_1 is equivalent to $O'_1 = wait_t \to (Flag_t = w)$ (if this is not immediately clear, it should become clear once we have discussed the notion of a *local proposition*). Property (4) denotes that the train is *accurate* with respect to recording his waiting state. We give a 'weakest' constraint β_1 that implements this social law, which can be verified by checking $S \dagger \beta_1, r \models \langle\langle\rangle\rangle \Box O'_1$. The constraint β_1 works

Current State	Actions	Resulting States
$staA_t, X, Y, Z$	$move_t, ac_g$	$staB_t, X', Y', Z$
$staB_t, X, Y, Z$	$move_t, ac_g$	$wait_t, X', Y', Z$
$wait_t, X, Y, Z$	$move_t, ac_g$	$oncr_t, X', Y', Z$
$oncr_t, X, Y, Z$	$move_t, ac_g$	$staA_t, X', Y', Z$
$W, X, closed_g, Z$	$ac_t, gates_g$	$W', X, open_g, Z'$
$W, X, open_g, Z$	$ac_t, gates_g$	$W', X, closed_g, Z'$
W, X, Y, Z	$idle_t, ac_g$	W, X', Y', Z
W, X, Y, Z	$ac_t, idle_g$	W', X, Y, Z'
$staA_t, X, Y, Z$	$moveann_t, ac_g$	$staB_t, X', Y', a$
$staB_t, X, Y, Z$	$moveann_t, ac_g$	$wait_t, X', Y', w$
$wait_t, X, Y, Z$	$moveann_t, ac_g$	$oncr_t, X', Y', o$
$oncr_t, X, Y, Z$	$moveann_t, ac_g$	$staA_t, X', Y', a$
$W, X, closed_g, Z$	$ac_t, gatesann_g$	$W', true, open_g, Z'$
$W, X, open_g, Z$	$ac_t, gatesann_g$	$W', false, closed_g, Z'$
W, X, Y, Z	$tell\text{-}away_t, ac_g$	W, X', Y', a
W, X, Y, Z	$tell\text{-}wait_t, ac_g$	W, X', Y', w
W, X, Y, Z	$tell\text{-}oncr_t, ac_g$	W, X', Y', o
W, X, Y, Z	$ac_t, tell\text{-}closed_g$	$W', false, Y, Z'$
W, X, Y, Z	$ac_t, tell\text{-}open_g$	$W', true, Y, Z'$

Figure 2: State Transitions

as follows: in $\langle staB_t, X, Y, Z \rangle$, if $Z \neq w$, the action for t forbidden by β_1 is $move$ (since this would inaccurately leave the $Flag_t$ to a or o); moreover, in any state $\langle wait_t, X, Y, Z \rangle$, constraint β_1 forbids t to perform any $tell$ action, except when it is $tell\text{-}wait$. Loosely speaking: the train is accurate about his waiting, if he announces it when he starts to wait, and, once waiting, never tells anything otherwise. We claim that moreover β_1 is a weakest constraint for its aims: any constraint β that forbids less than β_1 has the property that $S \dagger \beta, r \models \langle\!\langle \{t, g\} \rangle\!\rangle \diamondsuit \neg O_1$.

The converse of (4) would mean that the train is *truthful* with respect to the waiting state:

$$O_2 = K_t(Flag_t = w) \rightarrow wait_t \tag{5}$$

Again, this property is equal to $O_2' = (Flag_t = w) \rightarrow wait_t$. The behavioural constraint β_2 that makes this social law effective is to forbid t to falsely suggest that it is waiting: t is only allowed to perform $tell\text{-}wait_t$ in any $\langle wait_t, X, Y, Z \rangle$, moreover, β_2 forbids t to perform a $move_t$ in $\langle wait_t, Y, Z, w \rangle$, since otherwise it would falsely suggest that it is still waiting (of course, the train can still leave this state truthfully, by performing a $moveann_t$-action). The fact that indeed β_2 implements O_2' is verified by showing $S \dagger \beta_2, \rho \models \langle\!\langle \rangle\!\rangle \square O_2'$. As an aside, note that the constraint β_4 that *never* allows the train to do a $move_t$ action and only allows t to perform $tell\text{-}wait$ when $wait_t$ is true, is a way to implement the objective $O_4 = ((Flag_t = w) \rightarrow wait_t)$.

Now we observe the following: $S \dagger \beta_1 \dagger \beta_2, r \models wait_t \rightarrow K_g wait_t$. This is interesting, since to make the gates know that the train is waiting, has become a train-feasible objective in $S \dagger \beta_1 \dagger \beta_2$: he just has to accurately and truthfully set $Flag_t$ to w!

Now consider the following objective:

$$O = \langle\!\langle \rangle\!\rangle \square \neg (oncr_t \wedge closed_g) \wedge \langle\!\langle g, t \rangle\!\rangle \diamondsuit onct_t$$

This objective combines a safety property (the train is never on the crossing while it is closed) with a liveness prop-

erty (the train will eventually pass the crossing). The question is whether we can 'break' this objective 'down' into a number of constraints that are feasible for t or g. Here is a high level description: Let O' be the conjunction of:

$(K_t wait_t \rightarrow K_g wait_t) \qquad (K_g wait_t \rightarrow [\![g]\!] \diamondsuit open_g)$

$(K_g open_g \rightarrow K_t open_g) \qquad (K_t open_g \rightarrow \langle\!\langle t \rangle\!\rangle \diamondsuit oncr_t)$

$(K_g closed_g \rightarrow K_t closed_g) \quad (K_t closed_g \rightarrow [\![t]\!] \square \neg oncr_t)$

Objective O' states that t should inform g about waiting, and g should inform t about the states of the gates. Recall that $[\![i]\!] \diamondsuit \varphi$ means that the other agents can ensure $\diamondsuit \varphi$, and, if φ is 'under control' of i, then $[\![i]\!] \diamondsuit \varphi$ means that i cannot avoid that φ will eventually be true. Keeping this in mind, O' then also requires that if g knows that t is waiting, it cannot but avoid that eventually the gates will be open; if t knows the gates to be open, it will eventually pass the crossing, and, finally, if t knows the gates are closed, it will never attempt to cross. We claim that O' can be indeed turned into a set of feasible laws for g and t, and also that the implemented law for O' guarantees O.

We demonstrated how to make the first implication feasible for t by imposing suitable constraints; the same can be done for the other implications. We will now demonstrate how to actually model-check such knowledge properties.

5.2 Model Checking Some Properties

There are certain knowledge properties that need to be satisfied in the level crossing example. We obtain them using social laws. In order to test knowledge properties, we use a model checker called MOCHA, which takes as input an ATS described using a high-level language called REACTIVE MODULES. We programmed the system in this language and also have modified versions of the system, which incorporate social laws. It is important to note that we are not adding to the model checking theory, simply making use of it.

The first knowledge property that we wish to investigate is that the gates always know when the train is waiting. More formally, we want to verify that, given a state q in which $q \models wait_t$, we also have $q \models K_g wait_t$. The intuition behind this knowledge property is that if the train is waiting at the gates for them to be opened, this can only happen if the gate controller agent knows that the train is waiting.

Now we wish to verify properties involving knowledge by using a standard model checker that does not deal with epistemic operators, in our case MOCHA ([1]). To do this, we employ the machinery of *local propositions*, as introduced in [4] and applied to model checking epistemic properties in the context of linear temporal logic in [17]. We give an informal explanation; for details the reader should consult [17].

A proposition φ is i-local in a system S if

$$\forall q, q' \in S : ((q \sim_i q') \Rightarrow (S, q \models \varphi \Leftrightarrow S, q' \models \varphi))$$

that is, an i-local proposition never changes truth value *within* an i-equivalence class of states. This formalises the idea that such propositions depend on what the agent can observe. We immediately see that in the train and gates example, every proposition about the location of the train and the flag of the gates is t-local, likewise are assertions about the status of the gates and the flag of the train local for the gates. Now, generalizing the linear temporal logic analysis from [17] to the branching time context of this paper, we have the following:

THEOREM 2. *Suppose that φ is i-local. Then $S, q \models \varphi$ and $S, q_0 \models \langle\langle\rangle\rangle \Box(\varphi \to \psi)$ are sufficient to prove both $S, q \models K_i\psi$ and $S \models \varphi \to K_i\psi$. In such a case, φ is called i-local for ψ.*

This theorem immediately shows that the laws O_2 and O_2' are equivalent. To show that $q \models K_t wait_t$ in any waiting state q, we must first find a g-local proposition for $wait_t$. We can take $(Flag_t = w)$: To show that $(Flag_t = w)$ is g-local in $S \dagger \beta_2$, we need to verify that $\forall q, r((q \sim_g r) \Rightarrow (q \models Flag_t = w \Leftrightarrow r \models Flag_t = w))$, which is obvious, since $Flag_t$ is part of g's local state. Now, to apply Theorem 2, we must show that $S \dagger \beta_2, r \models \langle\langle\rangle\rangle \Box(Flag_t = w) \to wait_t$ holds. We do this by model checking the following MOCHA ATL specification in $S \dagger \beta_2$:

```
<<>> G ((tFlag = w) => (tState = wait))
```

Since the answer to this is positive, we have verified the desired property in $S \dagger \beta_2$. Note that this is relative to a state q. We now show how we can check this property across *all* states of the system. The only thing we required in q is that $q \models wait_t$. So now we can check the following property across all states, in $S \dagger \beta_2$:

$$wait_t \to K_g wait_t \qquad (6)$$

Now we model check the following:

```
<<>> G ((tState = wait) => (tFlag = w))
```

When this formula is model checked in the original system S it fails: This is as expected, as there is nothing in-built in S to guarantee that the train is truthful. However, model checking the above formula in $S \dagger \beta_2$ passes, which shows that this social law is effective.

Now we look at a nested knowledge formula:

$$wait_t \to K_t K_g wait_t \qquad (7)$$

We briefly sketch how to do this in the system that is constrained with β_4. By Theorem 2, it is sufficient to show that $wait_t$ is t-local for $K_g wait_t$. That is, $wait_t$ is t-local (which is obviously the case) and $S \dagger \beta_4, \rho \models \langle\langle\rangle\rangle \Box(wait_t \to K_g wait_t)$. This can be either model checked or established from (6), the fact that $S \dagger \beta_4 \sqsubseteq S \dagger \beta_2$ and Lemma 1. We can now apply Theorem 2 (take $\varphi = wait_t$ and $\psi = K_g wait_t$), to conclude that $S \dagger \beta_4 \models wait_t \to K_t K_g wait_t$.

The social laws imposed on our system are quite restrictive in that agent t knows everything about the state of agent g, and agent g knows a lot about the state of agent t. The knowledge that each of the agents have is *necessary* to ensure the system runs efficiently and that no crash situations occur. However, when the train is at $staA_t$ or $staB_t$, it is not beneficial to the system that agent g should know which of the two states the train is at, only that the train is away, in order to close the gates. We show this by investigating the following property in S:

$$staA_t \to M_g \neg staA_t \qquad (8)$$

This property states that if the train is at station A, agent g *considers it possible* that in fact, the train is *not* at station A. To check whether (8) holds across all states of our system S, we model check the following formula:

$$\bigwedge_{x,y} (\langle\langle\rangle\rangle \Box(staA_t \wedge x \wedge y) \to \langle\langle\rangle\rangle \Box(\neg staA_t \wedge x \wedge y)) \qquad (9)$$

where the conjunction is taken over all

$$x \in \{open_g, closed_g\},$$

and

$$y \in \{Flag_t = a, Flag_t = w, Flag_t = o\}.$$

This requires performing six different model checking problems, for each of the different values of x and y. After performing this model checking, as the formula passed each time, we have shown that (8) holds. Finally notice that even if the constraints concerning accuracy and truthfulness of both agents are implemented, (8) holds, since the most specific information that t will give when being at station A or B will be that he is 'away'.

6. RELATED WORK

As our framework is an extension of [16], it is clearly related to this work. Our approach is also related to the original frameworks of Moses and Tennenholtz, and that of Shoham and Tennenholtz. Shoham and Tennenholtz were the first to articulate the notion of social laws for multiagent systems, and set up a basic formal framework within which computational questions about social laws could be formulated ([14]).

In [8], Moses and Tennenholtz developed an deontic epistemic logic for representing properties of multiagent systems with normative structures in place. This framework is semantically similar to ours, not least because they also define epistemic accessibility relations, intended to interpret the agents' knowledge. These relations are defined to be equivalence relations representing states that are indistinguishable to agent i. We capture knowledge in precisely this way. However, Moses and Tennenholtz do not go on to make use of these relations in formulating social laws containing knowledge, as we do in this paper.

Another body of work that can be deemed similar to ours, is that of Rohit Parikh et al. In [9], they put forward the notion of *knowledge based obligation*. Here, the idea is that obligations are dependent on what the agent knows. For example, if a doctor does not know that a patient is ill, there is no obligation for the doctor to treat the patient. However, if the doctor does know that the patient is ill, this creates an obligation for the doctor to treat the patient. Obligations can also be over-ridden by more relevant information. Obligations are intended to further some general good and given the choice, agents should do what is best for society. These obligations can be thought of as similar to our social laws, where, for example, if the train moves from the crossing to station A, it is *obliged* to inform the gates of its next state.

7. CONCLUSION

In this paper we have shown how the social laws framework of van der Hoek et al [16] can naturally be extended to incorporate the notion of knowledge. In order to express such social laws we use the language of ATEL, essentially ATL with epistemic extensions. We demonstrated the power of such a framework with the use of a case study, in which many interesting knowledge properties were investigated. In particular, we showed how overall objectives of the system can be broken down into feasible properties for the agents,

involving laws for each agent i of type $K_i\varphi \rightarrow \psi$, where ψ is under control of i, possibly after implementing a social law.

We also demonstrated how several different types of knowledge, including knowledge pre-conditions, implied knowledge, and nested knowledge can be verified using an ATL model checker (MOCHA). We showed how ATEL formulae can be reduced to ATL formulae with the use of local propositions substituted for knowledge. This allowed us to verify such properties with a standard model checker that does not deal with epistemic operators.

Now we have extended the framework further, there are still many avenues for further investigation. It is interesting to incorporate the notions of minimality and simplicity ([6]) into the framework, and investigate them in this setting. The idea behind minimality and simplicity is essentially not to 'over-constrain' the system when achieving its objective. Second, the social laws are not yet expressible in the object language: we state them in ATEL, but then deal with them only semantically.

We are currently extending the framework to allow explicitly in the object language to not only refer to *obligations* and *duties*, but also to specific *actions* or *strategies*, giving an expressive language to reason about *who* should achieve *what* in *which* way. One attempt to add terms of actions in an ATL-like framework is to be found in [15], be it in a setting that does not address knowledge.

Finally, we like to have a general theory that predicts when an overall objective can be translated in individual feasible objectives, a question that is obviously related to the choice of how to partition the states in local substates. We think the analysis of [19] maybe helpful here. In [19], a notion of co-operation and control is studied in which each agent has the power to 'set' or 'unset' certain propositional atoms. This may be relevant for our social laws in the following sense: if φ is the objective of a social law, and all the atoms occurring in φ are $p_1, \ldots p_k$, then the agents that control those atoms have a kind of 'responsibility' to implement the social laws that is greater than those who cannot influence the truth of φ directly.

8. REFERENCES

[1] R. Alur, L. de Alfaro, T. A. Henzinger, S. C. Krishnan, F. Y. C. Mang, S. Qadeer, S. K. Rajamani, and S. Taşiran. MOCHA user manual. University of Berkeley Report, 2000.

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

[3] P. Blackburn, M. de Rijke, and Y. Venema. *Modal Logic*. Cambridge University Press: Cambridge, England, 2001.

[4] K. Engelhardt, R. van der Meyden, and Y. Moses. Knowledge and the logic of local propositions. In *Proceedings of the 1998 Conference on Theoretical Aspects of Reasoning about Knowledge (TARK98)*, pages 29–41, 1998.

[5] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning About Knowledge*. The MIT Press: Cambridge, MA, 1995.

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

[7] J.-J. C. Meyer and W. van der Hoek. *Epistemic Logic for AI and Computer Science*. Cambridge University Press: Cambridge, England, 1995.

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

[9] R. Parikh, E. Pacuit, and E. Cogan. The logic of knowledge based obligation. Presented at DALT 2004, available at www.cs.gc.cuny.edu/~epacuit/docs/kbo.pdf.

[10] Y. Shoham and M. Tennenholtz. Emergent conventions in multi-agent systems. In C. Rich, W. Swartout, and B. Nebel, editors, *Proceedings of Knowledge Representation and Reasoning (KR&R-92)*, pages 225–231, 1992.

[11] Y. Shoham and M. Tennenholtz. On the synthesis of useful social laws for artificial agent societies. In *Proceedings of the Tenth National Conference on Artificial Intelligence (AAAI-92)*, San Diego, CA, 1992.

[12] Y. Shoham and M. Tennenholtz. On social laws for artificial agent societies: Off-line design. *Artificial Intelligence*, 73(1-2):231–252, 1995.

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

[14] Y. Shoham and M. Tennenholtz. On the emergence of social conventions: Modelling, analysis, and simulations. *Artificial Intelligence*, 94(1-2):139–166, July 1997.

[15] W. van der Hoek, W. Jamroga, and M. Wooldridge. A logic for strategic reasoning, 2005. See elsewhere in these proceedings of *AAMAS05*.

[16] W. van der Hoek, M. Roberts, and M. Wooldridge. Social laws in alternating time: Effectiveness, feasibility, and synthesis. Technical Report ULCS-04-017, The University of Liverpool, 2005. To appear in the journal *Synthese*.

[17] W. van der Hoek and M. Wooldridge. Model checking knowledge and time. In D. Bošnački and S. Leue, editors, *Model Checking Software, Proceedings of SPIN 2002 (LNCS Volume 2318)*, pages 95–111. Springer, Germany, 2002.

[18] W. van der Hoek and M. Wooldridge. Time, knowledge, and cooperation: Alternating-time temporal epistemic logic and its applications. *Studia Logica*, 2003.

[19] W. van der Hoek and M. Wooldridge. On the logic of co-operation and propositional control. *Artificial Intelligence*, 64(1-2):81–119, 2005.