Synthesizing Social Laws with ATL Conditions

Extended Abstract

Rustam Galimullin
Department of Information Science and Media Studies
University of Bergen
Bergen, Norway
rustam.galimullin@uib.no

Louwe B. Kuijer
Department of Computer Science
University of Liverpool
Liverpool, United Kingdom
louwe.kuijer@liverpool.ac.uk

ABSTRACT

We introduce a formalism called SLAM (Social Laws on ATL Models) for defining social laws. Such social laws can constrain the behaviour of a multi-agent system. Importantly, these social laws can use any ATL formula as the condition under which an action is allowed. We show that the synthesis problem for these social laws is NP-complete. This generalizes a known result that synthesis of social laws that use only Boolean conditions is NP-complete.

KEYWORDS

ATL; Social Laws; Normative Systems; Synthesis

ACM Reference Format:
Rustam Galimullin and Louwe B. Kuijer. 2024. Synthesizing Social Laws with ATL Conditions: Extended Abstract. In *Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), Auckland, New Zealand, May 6 – 10, 2024*, IFAAMAS, 3 pages.

1 INTRODUCTION

In a multi-agent system, the agents typically have the ability to take actions that would result in an outcome that is considered detrimental. It is therefore wise to constrain the agents' behaviour, using so-called *social laws*. Simply put, a social law divides the set of possible actions in any given context into the actions that are allowed (by the social law) and those that are not.

We can then reason about the effects of a social law. In particular, if we want the agents to achieve a given goal, we can check whether that goal will be achieved if the agents follow the law. Or, given the goal, we can search for a social law that will satisfy that goal.

Multi-agent systems constrained by social laws are also known as *normative systems*, see, e.g., [2, 6, 12]. The seminal paper on social laws is [10], which considers the problem of *synthesizing* social laws, i.e., the problem to find a law that will bring about a desired goal. More precisely, [10] shows that in their definition of social laws, determining whether a satisfactory law exists is NP-complete.

Here, we use *concurrent game models* (CGMs) as our model of agency, and a variant of *alternating-time temporal logic* (ATL) [7] to reason about that agency, define goals and social laws. We assume that social laws describe the condition under which an action may be taken, with this condition also being defined in ATL. Specifically, social laws are sets of pairs of (a, φ) with a being an action and φ being an ATL formula. The other approach in the literature [2, 11, 12] is to treat social laws as purely a set of permissible actions. We, however, follow and generalize the approach from [10].

We refer to our formalism as *SLAM*, which is a backronym for "social laws on ATL-style models". SLAM is a considerably more general formalism than that of [10], yet we show that synthesis remains NP-complete, which is the main technical result of this work. The result is non-trivial as it requires a novel notion of bisimulation in order to show that we can always construct a sufficiently succinct law, i.e. of polynomial size. Moreover, our approach to synthesis allows for synthesizing several social laws at once.

Proc. of the 23rd International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2024), N. Alechina, V. Dignum, M. Dastani, J.S. Sichman (eds.), May 6 – 10, 2024, Auckland, New Zealand. © 2024 International Foundation for Autonomous Agents and Multiagent Systems (www.ifaamas.org).

2 SLAM

Let $Ag = \{1, \cdots, n\}$ be a finite set of agents, At a countable set of atoms and Act a finite set of actions.

A *concurrent game model* (CGM) is a tuple $M = (S, En, \tau, V)$, where S is a non-empty set of *states*, $En : S \times Ag \to 2^{Act}$ is an *action enabling function* with the property that $En(s, i) \neq \emptyset$ for all $s \in S$ and $i \in Ag$, $\tau : \{(s, a_1, \cdots, a_n) \mid s \in S, a_i \in En(s, i)\} \to S$ is an *outcome function*, and $V : S \to 2^{At}$ is a *valuation function*.

Intuitively, $En(s, i)$ is the set of actions that agent i is able to carry out at state s. Note that this set is required to be non-empty. If $G \subseteq Ag$, we write $En(s, G)$ for $\Pi_{i \in G} En(s, i)$. Moreover, we denote the set of all tuples $(a_1, ..., a_n)$ as Act^G, and call elements of Act^G *action profiles*. An action profile A for G is *enabled* in s if $A \in En(s, G)$.

A *memoryless strategy profile* (or a *strategy*) for G is a function $\sigma_G : S \to Act^G$ with $\sigma_G(s, i)$ being an action that agent i takes in state s according to the strategy profile. A strategy profile σ_G is *enabled* if $\sigma_G(s, i) \in En(s, i)$ for all s and i, denoted by $\sigma_G \in En(G)$.

Given a CGM $M = (S, En, \tau, V)$, a state $s_0 \in S$ and a strategy profile $\sigma = (\sigma_1, \cdots, \sigma_n)$ for Ag, the *outcome* $O(s_0, \sigma)$ of executing σ in s_0 is the sequence $s_0, s_1, \cdots$ where for all $t \in \mathbb{N}$, $s_{t+1} = \tau(s_t, \sigma_1(s_t), \cdots, \sigma_n(s_t))$. The k-th element of $O(s_0, \sigma)$ is $O_k(s_0, \sigma)$.

The *language* $\mathcal{L}$ of SLAM is defined as:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \varphi \mid \langle\langle N : G \rangle\rangle \bigcirc \varphi \mid \langle\langle N : G \rangle\rangle \square \varphi \mid$$
$$\mid \langle\langle N : G \rangle\rangle \varphi \mathcal{U} \varphi \mid [N]\varphi$$
$$N ::= \eta \mid \overline{N} \mid N + N \mid N \times N \mid N \circ N$$

where $p \in At$ and $\eta : Ag \times Act \hookrightarrow \mathcal{L}$ is a partial function.

We refer to η as *atomic laws*. If $\eta(i, a) = \varphi$, then agent i is allowed to take action a in exactly those states where φ is true. The reason we use a *partial* function is that a laws may constrain only certain agents and actions. We assume every action is allowed unless explicitly forbidden, so if $\eta(i, a)$ is undefined then i may perform a.

Atomic laws can then be combined into complex laws, which we generally denote by N. The negation $\overline{N}$ allows an action if and only

if it is disallowed by N, the addition $N_1 + N_2$ allows an action if and only if it is allowed by either N_1 or N_2, and the product $N_1 \times N_2$ allows an action if and only if it is allowed by both N_1 and N_2. Finally, $N_1 \circ N_2$ allows an action if and only if (i) it is allowed by N_1, and (ii) it is allowed by N_2 after N_1 has been applied.

Given a CGM $M = (S, En, \tau, V)$ and $s \in S$, the *satisfaction relation* $\models$ is defined similarly to the one for ATL with the only difference that $En(G)$ is now parametrized by N. We provide the definition of only $\langle\langle N : G\rangle\rangle \bigcirc \varphi$ as an example.

$$M, s \models \langle\langle N : G\rangle\rangle \bigcirc \varphi \quad \text{iff} \quad \exists \sigma_G \in En^N(G), \forall \sigma_{-G} \in En^N(Ag \setminus G) : M, O_1(s, \sigma_G \cup \sigma_{-G}) \models \varphi$$

$$M, s \models [N]\varphi \quad \text{iff} \quad M^N, s \models \varphi$$

Where, given a model $M = (S, En, \tau, V)$, the *updated model* $M^N = (S, En^N, \tau^N, V)$ is as follows.

- If $N = \eta$, then for every a, $a \in En^N(s, i)$ if and only if $a \in En(s, i)$ and one of the following: (1) $\eta(i, a)$ is undefined, (2) $M, s \models \eta(i, a)$, (3) for every $b \in En^N(s, i)$, $\eta(i, b)$ is defined and $M, s \not\models \eta(i, b)$.
- If $N = \overline{N_1}$ then

$$En^N(s, i) = \begin{cases} En(s, i) & \text{if } En(s, i) = En^{N_1}(s, i) \\ En(s, i) \setminus En^{N_1}(s, i) & \text{otherwise} \end{cases}$$

- If $N = N_1 + N_2$ then $En^N(s, i) = En^{N_1}(s, i) \cup En^{N_2}(s, i)$.
- If $N = N_1 \times N_2$ then

$$En^N(s, i) = \begin{cases} En^{N_1}(s, i) \cap En^{N_2}(s, i) & \text{if } En^{N_1}(s, i) \cap En^{N_2}(s, i) \neq \emptyset \\ En(s, i) & \text{otherwise} \end{cases}$$

- If $N = N_1 \circ N_2$ then $En^N(s, i) = (En^{N_1})^{N_2}$.

Furthermore, τ^N is the restriction of τ to En^N.

The case distinctions in the definition of En^N are required because we assume that every social law will inherently allow at least one action for each agent in each state. This assumption is often called *reasonableness* [2–5]. If, in a state s, a law forbids every possible action that an agent i could take (subcase (3)), then that law is considered inapplicable to that agent in that state, and thus that agent may instead take any action.

3 SYNTHESIS OF SOCIAL LAWS

The *social law synthesis problem* takes as input a CGM M, a state s of M, and a goal formula φ with variables $X_1, \cdots, X_k$. It has as output either a collection $N_1, \cdots, N_k$ of social laws such that

$$M, s \models \varphi(X_1 \mapsto N_1, \cdots, X_k \mapsto N_k)$$

or, if no such social laws exist, output "NO".

Our definition of the synthesis problem is quite general, and it allows synthesizing multiple social laws at once that can occur in any place inside the goal formula. We show that even with this more flexible definition, social law existence is NP-complete.

First, observe that model checking SLAM can be done by a modification of the classic algorithm [7] for ATL.

Theorem 3.1. *Model checking SLAM is P-complete.*

We can also show the NP-hardness of the synthesis problem by a reduction from 3-SAT.

Lemma 3.2. *The social law synthesis problem is NP-hard.*

Membership in NP takes a little bit more work. First, we introduce a novel notion of bisimulation that is aware of action names.

Let $M_1 = (S_1, En_1, \tau_1, V_1)$ and $M_2 = (S_2, En_2, \tau_2, V_2)$ be CGMs. A relation $\sim \subseteq S_1 \times S_2$ is a *bisimulation* if for every $s_1 \sim s_2$ there is a partition $Ag = Ag^+ \cup Ag^-$ such that,

Atomic agreement $V_1(s_1) = V_2(s_2)$,
Action agreement for every $i \in Ag^+$, $En_1(s_1, i) = En_2(s_2, i)$,
Forth and Back For all action profiles $A^+ \in En_1(s_1, Ag^+)$, $A^- \in En_1(s_1, Ag^-)$ and $B^- \in En_2(s_2, Ag^-)$, we have that $\tau_1(s_1, A^+ \cup A^-) \sim \tau_2(s_2, A^+ \cup B^-)$.

If there is a bisimulation $\sim$ such that $M_1, s_1 \sim M_2, s_2$ we say that they are *bisimilar*, denoted $M_1, s_1 \approx M_2, s_2$.

This definition of bisimulation is rather unusual, and differs from the classic one for ATL [1]. We have a special condition, **action agreement**, because the labels of actions are relevant, if the agent has a meaningful choice. We therefore partition Ag into the agents that have a meaningful choice, Ag^+, and the agents that do not, Ag^-. An agent i has a meaningful choice if there are an action profile $A^{-i} \in En(s, Ag \setminus \{i\})$ and actions $a, b \in En(s, i)$ such that $A^{-i} \cup \{a\}$ and $A^{-i} \cup \{b\}$ have non-bisimilar outcomes.

While our definition of bisimulation is non-standard, it does satisfy the properties one would expect: the bisimulation relation is an equivalence relation, and bisimilar states satisfy the same formulas. Moreover, it plays the crucial role in the next theorem.

Theorem 3.3. *Let M_1 and M_2 be finite CGMs. If $M_1, s_1 \not\approx M_2, s_2$, then there is a formula $\varphi \in \mathcal{L}$ such that $M_1, s_1 \models \varphi$, $M_2, s_2 \not\models \varphi$ and $|\varphi|$ is of polynomial size with respect to $|M_1| + |M_2|$.*

The proof is by a variation of the Paige-Tarjan algorithm [9], where while computing refinements we also simultaneously generate sets of formulas that uniquely identify each bisimulation class. The size of φ is taken as the size of the corresponding DAG. Together, Theorem 3.3 and Lemma 3.2, imply the desired result.

Theorem 3.4. *The social law synthesis problem is NP-complete.*

4 DISCUSSION

We have introduced SLAM, a formalism for reasoning about social laws, where both the goals to be achieved by the laws and the laws themselves are formulated in ATL. We have shown that model checking for SLAM is in P, while the social law existence problem is NP-complete, which considerably generalizes the classic result [10] that deals with social laws that use only Boolean conditions.

The main direction for further research is generalizations to more expressive languages. Goals and laws expressed in ATL suffice for memoryless agents, but for more capable agents other languages may be useful. In particular, ATL* for goals and LTL$^-$ (i.e. LTL with past [8, Chapter 6.5]) for social laws. We expect, however, that synthesis for such social laws to be far harder than for ATL.

Given that social law existence is already NP-complete for the much simpler social laws from [10], there does not seem to be a pressing need to further investigate the complexity of reasoning about social laws for languages that are less expressive than ATL.

Another interesting direction is *global synthesis*. In that problem, we search for a social law that makes a goal true not merely in a single given model, but in every model. This is likely to be far more computationally expensive than the problem discussed here.

REFERENCES

[1] Thomas Ågotnes, Valentin Goranko, and Wojciech Jamroga. 2007. Alternating-time temporal logics with irrevocable strategies. In *Proceedings of the 11th TARK*, Dov Samet (Ed.). 15–24. https://doi.org/10.1145/1324249.1324256

[2] Thomas Ågotnes, Wiebe van der Hoek, Juan A. Rodríguez-Aguilar, Carles Sierra, and Michael Wooldridge. 2007. On the logic of normative systems. In *Proceedings of the 20th IJCAI*. 1175–1180.

[3] Thomas Ågotnes, Wiebe van der Hoek, and Michael J. Wooldridge. 2010. Robust normative systems and a logic of norm compliance. *Logic Journal of the IGPL* 18, 1 (2010), 4–30. https://doi.org/10.1093/jigpal/jzp070

[4] Thomas Ågotnes and Michael J. Wooldridge. 2010. Optimal social laws. In *Proceedings of the 9th AAMAS*, Wiebe van der Hoek, Gal A. Kaminka, Yves Lespérance, Michael Luck, and Sandip Sen (Eds.). IFAAMAS, 667–674.

[5] Natasha Alechina, Mehdi Dastani, and Brian Logan. 2013. Reasoning about Normative Update. In *Proceedings of the 23rd IJCAI*, Francesca Rossi (Ed.). 20–26.

[6] Natasha Alechina, Mehdi Dastani, and Brian Logan. 2018. Norm Specification and Verification in Multi-agent Systems. *Journal of Applied Logics* 5, 2 (2018), 457–491.

[7] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. 2002. Alternating-time temporal logic. *J. ACM* 49 (2002), 672–713. https://doi.org/10.1145/585265.585270

[8] Stéphane Demri, Valentin Goranko, and Martin Lange. 2016. *Temporal Logics in Computer Science: Finite-State Systems*. Cambridge Tracts in Theoretical Computer Science, Vol. 58. CUP. https://doi.org/10.1017/CBO9781139236119

[9] Robert Paige and Robert E. Tarjan. 1987. Three Partition Refinement Algorithms. *SIAM J. Comput.* 16, 6 (1987), 973–989. https://doi.org/10.1137/0216062

[10] Yoav Shoham and Moshe Tennenholtz. 1995. On social laws for artificial agent societies: off-line design. *Artificial Intelligence* 73, 1 (1995), 231–252. https://doi.org/10.1016/0004-3702(94)00007-N

[11] Wiebe van der Hoek, Mark Roberts, and Michael Wooldridge. 2007. Social laws in alternating time: effectivenss, feasibility and synthesis. *Synthese* 156 (2007), 1–19.

[12] Michael Wooldridge and Wiebe van der Hoek. 2005. On obligations and normative ability. *Journal of Applied Logic* 4 (2005), 396–420.