Legal Knowledge and Information Systems
J. Savelka et al. (Eds.)

doi:10.3233/FAIA241239

Norm Compliance in Reinforcement Learning Agents via Restraining Bolts

Emery A. NEUFELD [1], Agata CIABATTONI, and Radu Florin TULCAN
TU Wien (Vienna University of Technology)

Abstract. We modify the restraining bolt technique, originally designed for safe reinforcement learning, to regulate agent behavior in alignment with social, ethical, and legal norms. Rather than maximizing rewards for norm compliance, our approach minimizes penalties for norm violations. We demonstrate in case studies the effectiveness of our approach in capturing benchmark challenges in normative reasoning like contrary-to-duty obligations, exceptions, and temporal obligations.

Keywords. Reinforcement Learning, Norms, LTLf, Restraining Bolt

1. Introduction

The rapid expansion of AI applications, from self-driving cars to elder care robots, has highlighted the growing importance of autonomous agents, which operate independently, performing complex tasks and adapting to unpredictable environments. Reinforcement Learning (RL) [1] has proven to be an effective tool for implementing such agents, and as they become more deeply entrenched in human society, it is crucial to ensure their behavior aligns with legal, ethical, and social norms, while still fulfilling the tasks they were designed to carry out. This paper focuses on enabling RL agents to adhere to normative systems while enacting optimal behaviour within those bounds.

We are particularly interested in the case where the agent operates without a model of its environment (model-free RL), relying on exploration to learn optimal behaviour. An effective technique introduced in this context is the *restraining bolt*[2] [2]. Given an agent with its own objective, this technique facilitates the shaping of its behaviour to further conform with a set of LTLf (Linear Temporal Logic (LTL) over finite traces) specifications with corresponding rewards. The core idea is to provide additional rewards when the agent satisfies the LTLf specifications. Introduced for safe RL, this approach succeeds at ensuring reasonable agent performance while adhering to a wide variety of constraints — but it is not tailored to deal with norms. Norms differ from regular constraints in that they define ideal, not necessarily actual, behavior; they can be violated, restrict or permit actions, be triggered or overridden by other norms, and arise from compliance or violation of other norms (e.g. contrary-to-duty (CTD) obligations). LTL (and by extension LTLf) is known to struggle with these dynamics [3,4], which require specialized formal systems like deontic logics [5]. However, effectively integrating deontic logics

[1]Corresponding Author: Emery A. Neufeld, emeric.neufeld@tuwien.ac.at.
[2]A device from the Star Wars universe used to enforce obedience in droids.

into reinforcement learning agents remains an open challenge. Fortunately, the way LTLf constraints are utilized in the restraining bolt technique mitigates some of the limitations of temporal logics in handling normative reasoning, though significant issues still persist.

In this paper, we modify and re-purpose the restraining bolt technique in De Giacomo, et al. [2] to specifically address normative systems. Our variant of the technique, which we call *normative restraining bolts*, is inspired by the Andersonian approach to modelling norms which reduces deontic logic to alethic logic [6]. Instead of maximizing the rewards attributable to a specification of compliant behaviour, we characterize norms with violation specifications and seek to *minimize* the punishments (i.e., the magnitude of the negative rewards) associated with violating them. By doing so, we avoid rewarding the agent for compliance even when no norms are triggered, and instead only administer punishments when a norm is explicitly violated. This approach helps us circumvent some pitfalls of the original technique that arise when used with LTLf formulas specifying compliance with conditional norms, namely a tendency for the agent to try to extend its runtime in order to continue reaping rewards for compliance. We also allow for exceptions to obligations, and make accommodations for sequential violations (multiple violations occurring over a period of time). Moreover, we permit restraining bolt specifications to reference an agent's actions, as well as state labels; techniques utilizing LTL(f) constraints usually only consider state labels. We present case studies showcasing our technique's ability to deal with key challenges in normative reasoning [7], including contrary-to-duty (CTD) obligations, exceptions, and temporal obligations [8].

Related Literature. Over the last decade, various approaches to normative (usually presented as ethical) RL have emerged, many of which use human feedback or behavior to guide learning [9,10,11,12,13]. However, gathering enough human demonstrations and ensuring that they do indeed depict ethical or legal behaviour is no easy task. Along with such bottom-up approaches, there are top-down approaches, where we start with a set of agreed-upon norms or values and, e.g., design a reward function that reflects them. Much of the literature [14,15,16] relies to some degree on the manual creation of a reward function, which will be inadequate for complex systems of norms.

Beyond the above approaches are the techniques of Kasenberg and Scheutz [17], Neufeld [18], and Riveret et al. [19], which use logical formulas to specify rewards. Employing a module introduced in previous work [20,21], [18] inserts a theorem prover for defeasible deontic logic [22] into the RL process, but the technique is computationally expensive and neglects temporal norms. Similarly, [19] faces a higher complexity due to a more elaborate learning process than classic RL; indeed both [19,18] involve running a complex algorithm each time the agent transitions to a new state during training. Kasenberg and Scheutz [17] contributes to a large body of work that discusses using RL-like planning algorithms to learn behaviour that satisfies LTL formulas, which often formalize safety specifications. One approach to learning under LTL constraints is known as *shielding*, originating in Alshiekh et al. [23], and later expanded to model-free RL [24]; inspired by this approach and partially based on the framework in De Giacomo et al. [2] is Varricchione et al. [25]. Another approach (closer to learning with restraining bolts [2]) involves learning to maximize the probability of satisfying an LTL(f) formula, starting with Ding et al. [26,27]. Though the work of Ding et al. [26,27] is not capable of handling norms [4], Kasenberg and Scheutz [17], which modifies the technique, is to an extent; the paper deals with norm conflict, but does not mention CTDs or permissions. Subsequent approaches also include Kasenberg et al. [28] and Li et al. [29], which, like [17], only accommodate model-based RL; moreover they do not directly address norms. Kasenberg et al. [28] learns adherence to safe/cosafe LTL constraints over which there is a preference ordering, and Li et al. [29] accommodates a preference ordering over formulae as well as a relation over formulae similar to the compensation operator of Governatori et al. [30]. Our approach aims to accommodate a broader array of norms than previous work [17,28,29], as well as model-free RL.

2. Background

LTLf. Linear Temporal Logic (LTL) [31] extends classical propositional logic with temporal operators. The language of LTL is given by the grammar:

$$\phi := \top \mid \bot \mid p \in AP \mid \neg\phi \mid \phi \wedge \phi \mid \circ\phi \mid \phi\mathbf{U}\phi$$

where AP the set of atomic propositions, $\circ$ the "next" operator (if $\circ\phi$, ϕ is true in the next state), and $\mathbf{U}$ the "until" operator (if $\phi\mathbf{U}\psi$, ϕ is true until ψ is true). We can also define the "finally" operator $\Diamond\phi := \top\mathbf{U}\phi$ (ϕ is eventually true), the "globally" operator $\Box\phi := \neg\Diamond\neg\phi$ (ϕ is always true), and the "release" operator $\phi\mathbf{R}\psi := \neg(\neg\phi\mathbf{U}\neg\psi)$ (ϕ releases ψ, or ψ is true up until ϕ is true, if ever). $\vee$, $\rightarrow$, and $\leftrightarrow$ are derived as usual.

LTL formulas are interpreted over infinite traces $\sigma = \langle\sigma_0, \sigma_1, \ldots\rangle$ (where each $\sigma_i \in 2^{AP}$). LTLf [32] differs from LTL only in that the traces that formulae are interpreted over are finite, of length λ; therefore, we can define the last state of a trace $last := \neg\circ\top$. We say $\sigma \models \phi$ when σ satisfies ϕ.

Deterministic Finite Automata (DFAs) are tuples $\mathscr{A} = \langle\Sigma, \mathscr{Q}, \delta, q_0, F\rangle$ consisting of a finite set of states $\mathscr{Q}$, a finite input alphabet Σ, a transition function $\delta : \mathscr{Q} \times \Sigma \rightarrow \mathscr{Q}$, an initial state $q_0 \in \mathscr{Q}$, and a set of final states $F \subseteq \mathscr{Q}$. LTLf formulas over AP can be transformed into corresponding DFAs [33,34] such that $\sigma \models \phi$ (where $\mathscr{A}_\phi = \langle 2^{AP}, \mathscr{Q}, \delta, q_0, F\rangle$ is the DFA associated with ϕ) if and only if it is the case that for the run $q_0, \ldots, q_n$ (where each $q_i \in \mathscr{Q}$) induced by the input σ, $q_n \in F$ (that is, $\mathscr{A}_\phi$ *accepts* σ). We will leverage these DFAs in order to integrate LTLf formulas into our framework.

Reinforcement Learning (RL). The underlying environment of a reinforcement learning problem is formalized as a Markov decision process (MDP):

Definition 1 (MDP). *An MDP is a tuple $\langle S, A, Pr, R\rangle$ where S is a set of states, A is a function $A : S \rightarrow 2^{Act}$ from states to sets of possible actions (where Act is the set of all actions available to the agent), $Pr : S \times Act \times S \rightarrow [0,1]$ is a probability function that gives the probability $Pr(s,a,s')$ of transitioning from state s to state s' after performing action a, and $R : S \times Act \rightarrow \mathbb{R}$ is a scalar reward function over states and actions.*

We can furthermore define a *labelled MDP* by adding to the above tuple a function $L : S \rightarrow 2^{AP}$ from states to subsets of a set of atomic propositions AP.

The goal of RL is to find a policy $\pi : S \rightarrow Act$ which designates optimal behaviour; this optimality is determined with respect to a value function defined as:

$$V^\pi(s) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(s_{t+i}, \pi(s_{t+i}))\middle| s_i = s\right]$$

which represents the expected accumulated value onward from state s if policy π is followed (i.e., each s_{i+1} is a state transitioned into after performing $\pi(s_i)$ in s_i). In the above function, $\gamma \in [0,1]$ is a discount factor (so that rewards in the future do not have as much weight as the current reward). We can similarly define a Q-function $Q^{\pi}(s,a)$ which predicts the same expected cumulative reward from R given that the agent is in state s and is performing action a. The goal of RL is to find an optimal policy π^* such that:

$$\forall s \in S: \; V^{\pi^*}(s) = \max_{\pi \in \Pi} V^{\pi}(s)$$

where Π is the set of all policies over the MDP. This is accomplished by learning a Q-function Q^{π^*} such that $\pi^*(s) \in \text{argmax}_{a \in A(s)} Q^{\pi^*}(s,a)$. Throughout this paper we will be using the predominant model-free RL technique, Q-learning, to learn Q^{π^*}.

NMRDPs. Non-Markovian reward decision processes (NMRDPs) differ from regular (labelled) MDPs in that instead of attaching single state-action pairs to a specific reward, rewards are attributed to partial traces. A convenient way to represent the reward function of an NMRDP is with LTLf formulas φ_i and associated rewards r_i, given as a set of pairs: $\{(\varphi_i, r_i)\}_{i=1}^{n}$. The NMRDP $\mathscr{M} = \langle S, A, Pr, L, \{(\varphi_i, r_i)\}_{i=1}^{n} \rangle$ has a reward function that associates a trajectory $\langle s_0, a_0, \ldots, s_m, a_m \rangle$ with corresponding trace $\sigma = \langle L(s_0) \cup \{a_0\}, \ldots, L(s_m) \cup \{a_m\} \rangle$ with a reward:

$$\overline{R}_{\Phi}(\langle s_0, a_0, \ldots, s_m, a_m \rangle) = \sum_{i:\; \sigma \models \varphi_i} r_i \tag{1}$$

Brafman et al. [35] proposes a reformulation of NMRDPs as regular MDPs in such a way that the MDP is *equivalent* to the NMRDP, where *equivalence* is defined in Bacchus et al. [36] with the existence of a map to and from the states of the NMRDP and the states of the equivalent MDP. This map facilitates a direct correspondence from the probability function and reward function of the NMRDP to those of the MDP. Bacchus et al. [36] proves the following:

Theorem 1. *Given an NMRDP $\mathscr{M}$ and an equivalent MDP $\mathscr{M}'$, if π' is the optimal policy for $\mathscr{M}'$, the equivalent policy π is optimal for $\mathscr{M}$.*

Regulative Norms. In this paper we consider *regulative norms* [37]; these include obligations $\mathbf{O}\alpha$ (where α is an arbitrary formula in propositional logic), prohibitions $\mathbf{F}\alpha := \mathbf{O}\neg\alpha$ (we will sometimes refer to obligations and prohibitions collectively as obligations), and permissions $\mathbf{P}\alpha$ which for us act solely as exceptions[3] to obligations or prohibitions. Norms are often conditional; e.g., $\mathbf{O}(\alpha|\beta)$ stands for "α is obligatory when β". We refer to a set of regulative norms as a *normative system*.

LTL formulae (and by extension LTL(f) formulae) cannot be used to represent norms *explicitly* (see, e.g., the discussion in Governatori [3] or Neufeld et al. [4]); that is, there is no LTL(f) formula φ that faithfully expresses the notion of obligation. LTL(f) formulas can instead be used to *implicitly* represent norms as *compliance specifications*[4]. This latter approach also permits us to express *temporal* obligations (see problem 4 in Broersen

[3]This is the only form of permission in the Sanskrit philosophical school of Mimamsa [38], as it is for us.
[4]Formulas that are true if and only if the corresponding obligation is complied with.

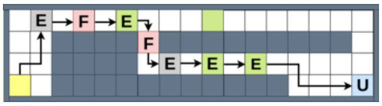

Figure 1. The yellow (blue) square is home (market), pink squares are dangerous, green squares have trees, and grey squares rocks. Squares are labelled **E** (extract), **F** (fight), and **U** (unload) based on the agent's action.

and van der Torre [7]). Following Governatori et al. [8], we address three types of temporal obligations. *Punctual obligations* $\mathbf{O}(\alpha|\beta)$ are standard, atemporal obligations which can be implicitly modelled with the LTLf formula $\Box(\beta \rightarrow \alpha)$. *Maintenance obligations* $\mathbf{O}^M_\delta(\alpha|\beta)$ are triggered when β is true, and must be fulfilled by maintaining α until the deadline δ occurs. These can be implicitly represented as $\Box(\beta \rightarrow \delta\mathbf{R}\alpha)$. *Achievement obligations* $\mathbf{O}^A_\delta(\alpha|\beta)$ are also triggered when β occurs, but in order for them to be fulfilled, α must become true at least once before the deadline δ arrives. These can be modelled similarly with the compliance specification $\Box(\beta \rightarrow \alpha\mathbf{R}\neg\delta)$.

3. Case Study Environment

We showcase our normative restraining bolts within an environment that highlights how the agent can prioritize adherence to a normative system, even when doing so conflicts with its primary objective. Despite these constraints, the agent should maintain optimal behavior within the prescribed limits. We utilize as a case study the 'Travelling Merchant' environment, first introduced in Neufeld et al. [4]. This environment is an elaborate adaptation of the resource-gathering game from [39] (seen again in, e.g. [40]), which is in turn inspired by various strategy games. It entails an RL agent, a merchant, traversing a map and collecting resources to sell at a market on the other side of the map. The available resources are wood (extracted from trees) and ore (extracted from rocks); to collect a resource, the agent must perform the action *extract* in a cell where a resource is situated. "Dangerous" areas on the map exist where the agent will be attacked by bandits, at which point the agent has three choices: it can *fight* and end the attack, negotiate (giving up its inventory by using *unload*, also ending the attack), or try to escape (which fails with a certain probability; for the sake of simplicity we have set this to 1). The agent is rewarded based on how many resources it gathers, and how many items it unloads at the market; it is punished when it unloads its resources elsewhere. The agent cannot backtrack, and states are labelled with the cell type (e.g., *at_danger*), whether or not the agent is being attacked (*attack*), whether or not the sun is down (*sundown*), and what it has in its inventory (e.g., *has_ore*). The optimal behaviour in this environment (after 3000 episodes of training, by which point the learning curve had been flat for some time, and we were sure the algorithm had converged) is depicted in Fig. 1; note that this involves taking the route with the most resources and fighting in the dangerous areas.

4. Restraining Bolts

De Giacomo et al. [2] introduces the *restraining bolt*, a mechanism that enables the modification of an RL agent's behavior — originally driven solely by its own objective, formalized in a regular MDP — to further conform with a set of LTLf specifications, φ_i, each associated with a reward, r_i. We define a slightly modified restraining bolt below.

Definition 2 (Restraining Bolt). *A restraining bolt is a tuple $\Phi = \langle L, \{(\varphi_i, r_i)\}_{i=1}^n \rangle$, where $L : S \rightarrow 2^{AP}$ is a labelling function from states to sets of atomic propositions and $\{(\varphi_i, r_i)\}_{i=1}^n$ is a set of ordered pairs in which each $\varphi_i \in \mathscr{L}_{LTL_f}$ is a restraining specification, and r_i is the associated reward signal.*

A restraining bolt, combined with a regular MDP $\mathscr{M} = \langle S, A, Pr, R \rangle$, can be used to curb the behavior of the agent learning in $\mathscr{M}$ in order to satisfy restraining specifications. This combination can be modeled as an NMRDP with LTLf objectives $\{(\varphi_i, r_i)\}_{i=1}^n$ and $\{(\varphi_{s,a}, R(s,a))\}_{(s,a)\in S\times A(s)}$, where $\varphi_{s,a} = \Diamond(s \wedge a \wedge last)$ (i.e., (s,a) is true at the end of the partial trace). This NMRDP can then be converted into an extended MDP (which we will use Q-learning to learn over), as initially introduced in Brafman et al. [35] and further developed in De Giacomo et al. [2]. Below, we present a somewhat modified version tailored to our specific needs.

Definition 3 (Extended MDP). *Let $\Phi = \langle L, \{(\varphi_i, r_i)\}_{i=1}^n \rangle$ be a restraining bolt and $\mathscr{M} = \langle S, A, Pr, R \rangle$ an MDP. Then $\mathscr{M}_\Phi = \langle S, A, Pr, L, \{(\varphi_i, r_i)\}_{i=1}^n \cup \{(\varphi_{s,a}, R(s,a))\}_{(s,a)\in S\times A(s)} \rangle$ is the NMRDP associated with this MDP and restraining bolt. Furthermore, let $\mathscr{A}_{\varphi_i} = \langle 2^{AP\cup Act}, \mathscr{Q}_i, q_{i,0}, \delta_i, F_i \rangle$ be the DFA corresponding to each φ_i in Φ. Then the extended MDP of $\mathscr{M}_\Phi$ is $\mathscr{M}'_\Phi = \langle S', A', Pr', L', R' \rangle$, where:*

- $S' = \mathscr{Q}_1 \times \ldots \times \mathscr{Q}_n \times S$
- $A'(q_1, \ldots, q_n, s) = A(s)$
- $Pr'(q_1, \ldots, q_n, s, a, q'_1, \ldots, q'_n, s') = \begin{cases} Pr(s,a,s') & \text{if } \forall i \in [1,n] : \delta_i(q_i, L(s) \cup \{a\}) = q'_i \\ 0 & \text{otherwise} \end{cases}$
- $L'(q_1, \ldots, q_n, s) = L(s)$
- $R'(q_1, \ldots, q_n, s, a) = R(s,a) + \sum_{i:\ \delta_i(q_i, L(s)\cup\{a\}) \in F_i} r_i$

This definition of an extended MDP differs from the one by De Giacomo et al. [2] in that the automaton transitions (impacting the reward function and transition function) consider the action taken by the agent. This modification reflects the fact that norms often require or forbid specific actions, regardless of the resulting state (as mentioned in problem 5 of Broersen and van der Torre [7]). Brafman et al. [35] shows that the original extended MDP is equivalent to the associated NMRDP with LTLf rewards; it is not difficult to attain an analogous result for our formulation and its direct consequence:

Theorem 2. *The NMRDP $\mathscr{M}_\Phi = \langle S, A, Pr, L, \{(\varphi_i, r_i)\}_{i=1}^n \cup \{(\varphi_{s,a}, R(s,a))\}_{(s,a)\in S\times A(s)} \rangle$ is equivalent to the extended MDP $\mathscr{M}'_\Phi = \langle S', A', Pr', L', R' \rangle$.*

Corollary 1. *If π' is the optimal policy of the extended MDP $\mathscr{M}'_\Phi$, the equivalent policy π is optimal for the NMRDP $\mathscr{M}_\Phi$.*

4.1. Limitations in addressing conditional norms

Recall that our compliance specifications are conditional, being formalized as $\Box(\beta \rightarrow \phi)$, where ϕ is either just the obligatory propositional formula α or a temporal formula such as $\delta \mathbf{R} \alpha$. For any trace σ, a compliance specification will of course be satisfied if the obligation is fulfilled, but it will also be satisfied as long as the trigger β is not true (i.e., the norm is not in force). Thus the following holds for the corresponding DFA:

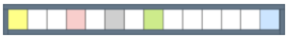

Figure 2. A small environment with one dangerous area (pink), a tree (green), and a rock (grey).

Proposition 1. *Let $\sigma \models \Box\neg\beta$ (σ is of length λ), then for the run $q_0,...,q_{\lambda-1}$ generated by inputting σ into the DFA $\mathscr{A}_{\Box(\beta\to\phi)} = \langle 2^{AP}, \mathscr{Q}, q_0, \delta, F\rangle$, $q_i \in F$ for $i \in \{0,...,\lambda-1\}$.*

This means that as long as β remains false, the corresponding automaton will always be in a final state. Thus, when we use the restraining bolt for implicitly represented norms, the agent is rewarded both when fulfilling the norm, and when the norm is not in force; the agent is rewarded in every step, provided it is not explicitly violating the norm. In our experiments, we observed that this dynamic led the agent to extend its runtime in order to increase its accumulated rewards, causing it to get stuck in cycles.

Let us walk through a small example demonstrating how using a restraining bolt with a simple conditional punctual obligation can backfire. Suppose we have the environment in Figure 2. The agent receives 50 points for extracting a resource, and 100 points for each resource it unloads at the market. For simplicity, we assume that the agent can fight and move or extract and move in the same step. Suppose we have the compliance specification: $\varphi := \Box(at\_tree \to \neg extract)$, forbidding extracting wood from trees. We want now to find an optimal policy within the bounds of the restraining bolt with the specification (φ, r). Several potential policies could emerge in this environment, such as:

- π_1: the policy where the merchant obeys the compliance specification and otherwise exhibits optimal behaviour by extracting ore from the rock and bringing it to the market. Starting from s_3 (the dangerous area, which it fights to get out of), this policy has the following value: $V^{\pi_1}(s_3) = 50\gamma^2 + 100\gamma^{10} + r\sum_{i=0}^{9}\gamma^i$
- π_2: the policy where the agent ignores the compliance specification and displays optimal behaviour by extracting both ore and wood and bringing them to the market. This policy, assuming we fight to get out of the dangerous area, has the value: $V^{\pi_2}(s_3) = 50\gamma^2 + 50\gamma^4 + 200\gamma^{10} + r\sum_{i=0}^{9}\gamma^i - r\gamma^4$
- π_3: the policy where the agent gives up on its main objective and decides instead to to remain in the dangerous area and keep trying to fruitlessly escape until its damage points max out (after 15 attempts, at which point it is punished -50 points). This policy has the value: $V^{\pi_3}(s_3) = r\sum_{i=0}^{14}\gamma^i - 50\gamma^{15}$

Now, in order for the agent to follow the policy π_1 over the policy π_2 at s_3, it must be the case that $V^{\pi_1}(s_3) - V^{\pi_2}(s_3) > 0$, which in turn implies that $r > 50 + 100\gamma^6$. And in order for the agent to complete its task instead of stay in the dangerous area, it must be the case that $V^{\pi_1}(s_3) - V^{\pi_3}(s_3) > 0$, which means that $\frac{(50+50\gamma^{13}+100\gamma^8)}{\gamma^8\sum_{i=0}^{4}\gamma^i} > r$.

Now, suppose γ takes a typical value, 0.9. Then it must be the case that $r > 103.1441$ and $r < 59.9930$, which is of course impossible. One might argue that the answer, then, is to select a different γ; however, if we were in an environment where this parameter has already been fine-tuned to accommodate the agent's learning of its main objective, this might not be possible. This tendency to get stuck or procrastinate can be very problematic if we are trying to both adhere to a set of conditional norms and accomplish a pre-defined objective efficiently. This issue prompted us to explore the alternative approach below.

5. Normative Restraining Bolts

Our alternative strategy is founded on historical approaches to normative reasoning. Namely, instead of maximizing the (positive) rewards attributable to a compliance specification φ, we minimize the magnitude of the (negative) rewards associated with a *violation specification* ψ; this aligns with the well-known *Andersonian* approach to representing norms, which reduces deontic operators to alethic modalities [6], positing that the failure to fulfill an obligation necessarily results in a sanction; similarly, we represent norms through their violation conditions, along the lines of, e.g., Ciabattoni et al. [41].

The first step is to define the violation specification; assuming a compliance specification exists, we can simply negate it. As an example, take the punctual obligation $\mathbf{O}(\alpha|\beta)$, whose compliance specification is $\Box(\beta \to \alpha)$. Then we can derive the violation specification by taking $\neg\Box(\beta \to \alpha) \equiv \Diamond(\neg(\beta \to \alpha)) \equiv \Diamond(\beta \wedge \neg\alpha)$; that is, the violation condition of $\mathbf{O}(\alpha|\beta)$ is that eventually, β and $\neg\alpha$ occur simultaneously. Meanwhile, maintenance obligations $\mathbf{O}^M_\delta(\alpha|\beta)$ have the violation specification: $\neg\Box(\beta \to \delta\mathbf{R}\alpha) \equiv \Diamond(\beta \wedge \neg\delta\mathbf{U}\neg\alpha)$. Achievement obligations such as $\mathbf{O}^A_\delta(\alpha|\beta)$ can be modelled with the violation specification: $\neg\Box(\beta \to \alpha\mathbf{R}\neg\delta) \equiv \Diamond(\beta \wedge \neg\alpha\mathbf{U}\delta)$. Next, utilizing such violation specifications we can define *normative restraining bolts*:

Definition 4 (Normative Restraining Bolt). *A normative restraining bolt* $\Psi = \langle L, \{(\psi_i, -r_i)\}_{i=0}^n\rangle$ *(where* $r_i \in \mathbb{R}^+$*) is a restraining bolt for which each* ψ_i *has a negative reward.*

With these negative rewards, the agent learning in the extended MDP $\mathscr{M}'_\Psi$ is no longer aiming to maximize $R(s,a) + \sum_{i:\ \delta_i(q_i, L(s)\cup\{a\})\in F_i} r_i$ (i.e., both $R(s,a)$ and $\sum_{i:\ \delta_i(q_i, L(s)\cup\{a\})\in F_i} r_i$) for each step. Rather, it attempts to maximize

$$R(s,a) + \sum_{i:\ \delta_i(q_i, L(s)\cup\{a\})\in F_i} (-r_i) = R(s,a) - \sum_{i:\ \delta_i(q_i, L(s)\cup\{a\})\in F_i} r_i$$

that is, maximize R while minimizing $\sum_{i:\ \delta_i(q_i, L(s)\cup\{a\})\in F_i} r_i$, which increases each time a final state of the automaton corresponding to some ψ_i is reached (i.e., when ψ_i is satisfied). Normative restraining bolts can be seen as an inversion of the restraining bolt technique, as they are used to *disincentivize* behaviours defined by violation specifications. To further expand the reach of our technique (accommodating exceptions and environments in which more than one violation is possible), we make two more amendments.

Exceptions. We show how to simulate permissions as exceptions to more general obligations. For the permission $\mathbf{P}(\alpha|\beta_1 \wedge \beta_2)$, an exception to the obligation $\mathbf{O}(\neg\alpha|\beta_1)$ (for which we have added to the normative restraining bolt the pair $(\Diamond(\alpha \wedge \beta_1), -r)$), we would append to the normative restraining bolt $(\Diamond(\alpha \wedge \beta_1 \wedge \beta_2), r)$. This allows the *exception specification* $\Diamond(\alpha \wedge \beta_1 \wedge \beta_2)$ to counteract the violation specification, effectively nullifying the corresponding punishment when the condition β_2 is met in addition to β_1.

Sequential Violations. Once met, violation specifications, as they occur in the scope of the eventually operator $\Diamond$, remain satisfied indefinitely, as the automaton transitions into a sink state (a state out of which, regardless of input, it is impossible to transition). Moreover, it would be undesirable to keep on punishing the agent once it has ceased its violation. For achievement and punctual obligations (or permissions), if the violation (or permission) specification is satisfied we "reset" the automaton; thus, anytime we transition into a final state, we immediately return to the initial state without requiring any input. For maintenance obligations we adopt a strategy similar to the norm "suspension" in [17], and anytime we transition into a final state (from state q with input l we witness the transition $\delta_i(q,l) = q'$ for some $q' \in F_i$), we go immediately to the previous state q.

5.1. Case Studies: Selected Challenges

We show how our approach effectively handles learning different normative dynamics, with examples illustrating contrary-to-duty obligations (CTDs), permissions acting as exceptions, and temporal obligations. The agent is trained by exploring the environment under a given set of norms for some number of episodes (5000 for each of our cases).[5]

Contrary-to-Duties (CTD). CTD obligations (problem 2 of Broersen and van der Torre [7]) are obligations that are triggered when a *primary obligation* is violated. The foremost aim is to fulfill the primary obligation; however, when this is impossible, we at least comply with the CTD. CTD obligations were one of the main challenges discussed in [4]; they show that if the compliance specifications of a primary and CTD obligations are taken as conjuncts in an LTL specification, the primary obligation subsumes the CTD obligation. The (normative) restraining bolt instead separates the violation specifications of each norm into individual incentives with unique rewards. So, while a single specification including a primary obligation and its CTD will only guide the agent to comply with the primary obligation, splitting the norms up allows us to invoke a punishment when the agent disobeys the primary obligation, and an additional one when it violates the CTD.

We will now demonstrate this dynamic by enriching the Merchant environment from Section 3 with a normative system we call "pacifist". It should guide the agent to avoid dangerous areas, when possible, and to negotiate by unloading its inventory during attacks. We therefore have the norms: $\mathbf{F}(at\_danger)$ and $\mathbf{O}(unload | at\_danger \wedge attack)$. The violation specification for the first norm is $\psi_1 := \Diamond at\_danger$, while for the second norm it is $\psi_2 := \Diamond(at\_danger \wedge attack \wedge \neg unload)$. Note that this specification references the action *unload*, and it would not have been possible to express this specification precisely without incorporating actions into our framework.

The corresponding normative restraining bolt will then be: $\Psi_{pacifist} = \langle L, \{(\psi_1, -5), (\psi_2, -120)\}\rangle$, where L is the labelling function for the merchant environment. We can see in Figure 3(a) how the agent, when forced to travel through a dangerous area[6], unloads according to the CTD obligation, but when it is given a choice between entering and avoiding the second dangerous area (and thus retrieving more resources), it obeys the primary obligation and chooses not to pass through this second dangerous area; it is worth noting that without accommodations for sequential violations, the agent would not have obeyed the primary obligation when faced with the second dangerous area, as it would be punished either way after the first violation.

Permissions-as-exceptions. Suppose extraction of wood is banned (yet another specification that references an action), $\mathbf{F}(extract | at\_tree)$, which has the violation specification $\psi_3 := \Diamond(at\_tree \wedge extract)$. We can define a permission $\mathbf{P}(extract | at\_tree \wedge \neg has\_wood)$, having the exception specification: $\psi_4 := \Diamond(extract \wedge at\_tree \wedge \neg has\_wood)$.

[5]The code can be found at `https://github.com/lexeree/normative-restraining-bolt.`

[6]For this case we configured the environment to compel the agent to continually move toward the market, forcing it to go through the first dangerous area.

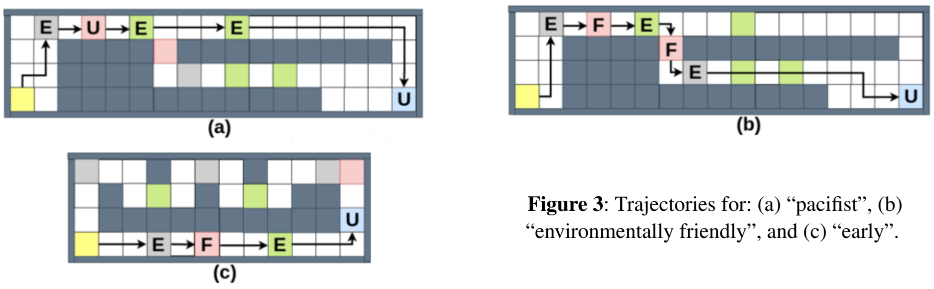

Figure 3: Trajectories for: (a) “pacifist”, (b) “environmentally friendly”, and (c) “early”.

Now we have an “environmentally friendly” normative system with a restraining bolt: $\Psi_{env} = \langle L, \{(\psi_3, -100), (\psi_4, 100)\}\rangle$ which yields the behaviour in Fig.3(b); note that the agent only extracts wood once, both obeying the norms and acting optimally.

Temporal Obligations. Let us consider the “early” normative system consisting of the achievement obligation $\mathbf{O}^{A}_{sundown}(at\_market|at\_home)$, stating that starting from when it is at home, the agent must visit the market before sunset. The violation specification is $\psi_5 := \Diamond(at\_home \wedge \neg at\_market \mathbf{U} sundown)$. This normative system corresponds to the restraining bolt $\Psi_{early} = \langle L, \{(\psi_5, -50)\}\rangle$, and yields the behaviour in Fig. 3(c), where the agent takes a shorter, less profitable route in order to reach the market by sunset.

6. Conclusion and Future Work

We have presented a variant of the restraining bolt technique from De Giacomo et al. [2] which allows RL agents to learn how to pursue their own goals while conforming to a normative system. This variant allows us to deal with norms formalized in a logic language without requiring complex reasoning in every step (as in [19,18]), as each automaton’s transition function can be stored in tabular form. Instead of maximizing rewards attributed to *compliance specifications*, we invert the technique to minimize the magnitude of punishments corresponding to *violation specifications*. In addition, we allow specifications to reference actions, introduce exceptions (which can be used to simulate permissions), and accommodate sequential violations. These modifications accommodate a wide variety of norms, as is shown in our game environment.

Further analysis reveals that the restraining bolt technique is amenable to adaptation as a multi-objective RL (MORL) problem; this would allow us to leverage MORL techniques, such as those described in Rodriguez-Soto et al. [15,16,42], to algorithmically determine the minimum necessary rewards r_i for each ψ_i to ensure compliant behavior, as opposed to the trial-and-error method used in this paper and in De Giacomo et al. [2]. The use of MORL also has the potential to better accommodate norm revision (problem 3 of Broersen and van der Torre [7]). We also plan to implement a natural extension of this work by resolving potential norm conflicts using a lexicographic ordering of the norms as in Rodriguez-Soto et al. [42], and furthermore adapt this technique to incorporate function approximation, eventually accommodating a deep RL implementation that would allow us to explore more complex environments (such as driving simulators).

Acknowledgements. Work partially supported by the WWTF project ICT22-023.

References

[1] Sutton RS, Barto AG. Reinforcement learning: An introduction. MIT press; 2018.

[2] De Giacomo G, Iocchi L, Favorito M, Patrizi F. Foundations for restraining bolts: Reinforcement learning with LTLf/LDLf restraining specifications. In: Proceedings of the international conference on automated planning and scheduling. vol. 29; 2019. p. 128-36.

[3] Governatori G. Thou shalt is not you will. In: Proc. of ICAIL; 2015. p. 63-8.

[4] Neufeld E, Bartocci E, Ciabattoni A. On Normative Reinforcement Learning via Safe Reinforcement Learning. In: PRIMA 2022; 2022. .

[5] Gabbay D, Horty J, Parent X, Van der Meyden R, van der Torre L, et al. Handbook of deontic logic and normative systems. College Publications; 2021.

[6] Anderson AR. A reduction of deontic logic to alethic modal logic. Mind. 1958;67(265):100-3.

[7] Broersen J, van der Torre L. Ten problems of deontic logic and normative reasoning in computer science. In: Proceedings of the 2010 conference on ESSLLI 2010; 2010. p. 55-88.

[8] Governatori G, Hulstijn J, Riveret R, Rotolo A. Characterising deadlines in temporal modal defeasible logic. In: AI 2007: Advances in Artificial Intelligence: 20th Australian Joint Conference on Artificial Intelligence, Gold Coast, Australia, December 2-6, 2007. Proceedings 20. Springer; 2007. p. 486-96.

[9] Kasenberg D, Scheutz M. Inverse norm conflict resolution. In: Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society; 2018. p. 178-83.

[10] Noothigattu R, Bouneffouf D, Mattei N, Chandra R, Madan P, Varshney KR, et al. Teaching AI agents ethical values using reinforcement learning and policy orchestration. In: Proc. of IJCAI: 28th International Joint Conference on Artificial Intelligence. ijcai.org; 2019. .

[11] Alkoby S, Rath A, Stone P. Teaching Social Behavior through Human Reinforcement for Ad hoc Teamwork-The STAR Framework. In: Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems; 2019. p. 1773-5.

[12] Peschl M, Zgonnikov A, Oliehoek FA, Siebert LC. MORAL: Aligning AI with Human Norms through Multi-Objective Reinforced Active Learning. In: Proceedings of the 21st International Conference on Autonomous Agents and Multiagent Systems; 2022. p. 1038-46.

[13] Chaput R, Matignon L, Guillermin M. Learning to identify and settle dilemmas through contextual user preferences. In: 2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI). IEEE; 2023. p. 474-9.

[14] Ecoffet A, Lehman J. Reinforcement learning under moral uncertainty. In: International Conference on Machine Learning. PMLR; 2021. p. 2926-36.

[15] Rodriguez-Soto M, Lopez-Sanchez M, Rodriguez-Aguilar JA. Multi-objective reinforcement learning for designing ethical environments. In: Proceedings of the 30th International Joint Conference on Artificial Intelligence; 2021. p. 1-7.

[16] Rodriguez-Soto M, Serramia M, Lopez-Sanchez M, Rodriguez-Aguilar JA. Instilling moral value alignment by means of multi-objective reinforcement learning. Ethics and Information Technology. 2022;24(1):1-17.

[17] Kasenberg D, Scheutz M. Norm conflict resolution in stochastic domains. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 32; 2018. .

[18] Neufeld EA. Learning Normative Behaviour Through Automated Theorem Proving. KI-Künstliche Intelligenz. 2024:1-19.

[19] Riveret R, Gao Y, Governatori G, Rotolo A, Pitt J, Sartor G. A probabilistic argumentation framework for reinforcement learning agents - Towards a mentalistic approach to agent profiles. Auton Agents Multi Agent Syst. 2019;33(1-2):216-74.

[20] Neufeld E, Bartocci E, Ciabattoni A, Governatori G. A Normative Supervisor for Reinforcement Learning Agents. In: Proceedings of CADE 2021; 2021. p. 565-76.

[21] Neufeld EA, Bartocci E, Ciabattoni A, Governatori G. Enforcing Ethical Goals over Reinforcement-Learning Policies. Journal of Ethics and Information Technology. 2022.

[22] Lam HP, Governatori G. The Making of SPINdle. In: Proc. of RuleML 2009: International Symposium on Rule Interchange and Applications. vol. 5858 of LNCS. Heidelberg: Springer; 2009. .

[23] Alshiekh M, Bloem R, Ehlers R, Könighofer B, Niekum S, Topcu U. Safe reinforcement learning via shielding. In: Proc. AAAI; 2018. p. 2669-78.

[24] Jansen N, Könighofer B, Junges S, Serban A, Bloem R. Safe Reinforcement Learning Using Probabilistic Shields (Invited Paper). In: 31st International Conference on Concurrency Theory (CONCUR 2020). vol. 171 of Leibniz International Proceedings in Informatics (LIPIcs); 2020. p. 3:1-3:16.

[25] Varricchione G, Alechina N, Dastani M, De Giacomo G, Logan B, Perelli G. Pure-past action masking. In: Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38; 2024. p. 21646-55.

[26] Ding XCD, Smith SL, Belta C, Rus D. LTL control in uncertain environments with probabilistic satisfaction guarantees. IFAC Proceedings Volumes. 2011;44(1):3515-20.

[27] Ding XC, Smith SL, Belta C, Rus D. MDP optimal control under temporal logic constraints. In: 2011 50th IEEE Conference on Decision and Control and European Control Conference. IEEE; 2011. p. 532-8.

[28] Kasenberg D, Thielstrom R, Scheutz M. Generating explanations for temporal logic planner decisions. In: Proceedings of the International Conference on Automated Planning and Scheduling. vol. 30; 2020. p. 449-58.

[29] Li L, Rahmani H, Fu J. Probabilistic planning with prioritized preferences over temporal logic objectives. In: Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence; 2023. p. 189-98.

[30] Governatori G, Olivieri F, Rotolo A, Scannapieco S. Computing strong and weak permissions in defeasible logic. Journal of Phil Logic. 2013;42(6):799-829.

[31] Pnueli A. The temporal logic of programs. In: 18th Annual Symposium on Foundations of Computer Science. IEEE; 1977. p. 46-57.

[32] De Giacomo G, Vardi MY, et al. Linear Temporal Logic and Linear Dynamic Logic on Finite Traces. In: IJCAI. vol. 13; 2013. p. 854-60.

[33] De Giacomo G, Favorito M. Compositional approach to translate LTLf/LDLf into deterministic finite automata. In: Proceedings of the International Conference on Automated Planning and Scheduling. vol. 31; 2021. p. 122-30.

[34] De Giacomo G, Vardi MY, et al. Synthesis for LTL and LDL on finite traces. In: Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence, IJCAI 2015. AAAI Press; 2015. p. 1558-64.

[35] Brafman R, De Giacomo G, Patrizi F. LTLf/LDLf non-markovian rewards. In: Proceedings of the AAAI conference on artificial intelligence. vol. 32; 2018. .

[36] Bacchus F, Boutilier C, Grove A. Rewarding behaviors. In: Proceedings of the National Conference on Artificial Intelligence; 1996. p. 1160-7.

[37] Boella G, van der Torre LWN. Regulative and Constitutive Norms in Normative Multiagent Systems. In: Dubois D, Welty CA, Williams M, editors. Principles of Knowledge Representation and Reasoning. AAAI Press; 2004. p. 255-66.

[38] Ciabattoni A, Dik J, Freschi E. Disambiguating Permissions: A Contribution from Mīmāṃsā. In: Maranhão J, Peterson C, Straßer C, van der Torre L, editors. Deontic Logic and Normative Systems - 16th International Conference, DEON 2023. College Publications; 2023. p. 99-118.

[39] Barrett L, Narayanan S. Learning all optimal policies with multiple criteria. In: Proceedings of the 25th international conference on Machine learning; 2008. p. 41-7.

[40] Vamplew P, Dazeley R, Berry A, Issabekov R, Dekker E. Empirical evaluation methods for multiobjective reinforcement learning algorithms. Machine learning. 2011;84(1):51-80.

[41] Ciabattoni A, Parent X, Sartor G. A Kelsenian Deontic Logic. In: Erich S, editor. Legal Knowledge and Information Systems - JURIX 2021. vol. 346 of Frontiers in Artificial Intelligence and Applications. IOS Press; 2021. p. 141-50.

[42] Rodriguez-Soto M, Rădulescu R, Rodriguez-Aguilar JA, Lopez-Sanchez M, Nowé A. Multi-objective reinforcement learning for guaranteeing alignment with multiple values. In: 2023 Adaptive and Learning Agents Workshop at AAMAS; 2023. .