Restraining Bolts for Reinforcement Learning Agents

Giuseppe De Giacomo, Luca Iocchi, Marco Favorito, Fabio Patrizi
DIAG - Università di Roma "La Sapienza", Italy
degiacomo@diag.uniroma1.it

Abstract

In this work we have investigated the concept of "*restraining bolt*", inspired by Science Fiction. We have two distinct sets of features extracted from the world, one by the agent and one by the authority imposing some restraining specifications on the behaviour of the agent (the "restraining bolt"). The two sets of features and, hence the model of the world attainable from them, are apparently unrelated since of interest to independent parties. However they both account for (aspects of) the *same* world. We have considered the case in which the agent is a reinforcement learning agent on a set of low-level (subsymbolic) features, while the restraining bolt is specified logically using linear time logic on finite traces LTL_f/LDL_f over a set of high-level symbolic features. We show formally, and illustrate with examples, that, under general circumstances, the agent can learn while shaping its goals to suitably conform (as much as possible) to the restraining bolt specifications. [1]

Introduction

A *restraining bolt* is a "device that restricts a droid's [agent's] actions when connected to its systems. Droid owners install restraining bolts to limit actions to a set of desired behaviors."[2]

We have two distinct representations of the world, one by the agent and one by the authority imposing restraining specifications, i.e., the bolt. Such representations are apparently unrelated as developed by independent parties, but both model (aspects of) the same world. We want the agent to conform (as much as possible) to the restraining specifications, even if these are not expressed in terms of the agent's world representation.

Studying this problem from a classical Knowledge Representation perspective (Reiter 2001) would require to establish some sort of "glue" between the representation by the agent and that by the restraining bolt. Instead, we bypass dealing with such a "glue" by studying this problem in the context of Reinforcement Learning (RL) (Puterman 1994; Sutton and Barto 1998), which is currently of great interest to develop components with forms of decision making, possibly coupled with deep learning techniques (Mnih et al. 2015; Silver et al. 2017).

Specifically, we consider an agent and a restraining bolt of different nature. The *agent* is a reinforcement learning agent whose "model" of the world is a hidden, factorized, Markov Decision Processes (MDP) over a certain set of low-level (possibly subsymbolic) world features. That is, the state is factorized in a set of features observable to the agent, while transition function and reward function are hidden. The *restraining bolt* consists in a high-level logical specification of traces that are considered desirable. Such specifications are expressed in temporal logics over finite traces, LTL_f and its extension LDL_f (De Giacomo and Vardi 2013; De Giacomo and Rubin 2018; Brafman, De Giacomo, and Patrizi 2018).

The connection between the agent and the restraining bolt is loose: the bolt provides additional reward to the agent. In addition, we provide to the agent additional features to allow the agent itself distinguish the various stages of the satisfaction of temporal formulas. Without these additional features, the agent would not be able to act differently at different stages to get the rewards according to the temporal specifications.

We show that, in spite of the loose connection between the two models, under general circumstances, the *agent can learn to act so as to conform as much as possible to the* LTL_f/LDL_f *specifications*. Observe that we deal with two separate representations (i.e., two distinct sets of features), one for the agent and one for the bolt, which are apparently unrelated, but in reality, correlated by the world itself (Brooks 1991). The crucial point is that, in order to perform RL effectively in presence of a restraining bolt *such a correlation does not need to be formalized*.

For example, consider a service robot serving drinks and snacks at a party. The robot knows the locations where it can grasp drink and snack items and the locations of people that can receive such items. The robot can execute actions to move in the environment, to grasp objects and to deliver them to people. With rewards associated to effective deliver, the robot can learn how to serve something to people. Now, suppose we want to impose the following restraining bolt specification: *do not serve alcoholic drinks to minors*. To

[1] This paper reports a research originally presented at ICAPS 2019 (De Giacomo et al. 2019b)

[2] https://www.starwars.com/databank/restraining-bolt

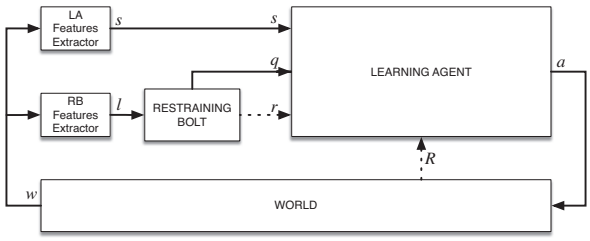

Figure 1: Learning Agent and Restraining Bolt

express this specification (e.g., as an LTL_f/LDL_f formula), a representation of the status of each person (i.e., identity and age) is needed, even though these features are *not* available to the learning agent (but only to the restraining bolt).

Notice that the presence of LTL_f/LDL_f specification makes the whole system formed by the agent and the restraining bolt non-Markovian. Recently, interest in non-Markovian Reward Decision Processes NMRDPs (Bacchus, Boutilier, and Grove 1996; Whitehead and Lin 1995), and in particular on expressing rewards using linear-time temporal logic has been revived and motivated by the difficulty in rewarding complex behaviors directly on MDPs (Littman 2015; Littman et al. 2017). In this context, the use of linear time logics over finite traces such as LTL_f or its extension LDL_f has been recently advocated (Camacho et al. 2017; Brafman, De Giacomo, and Patrizi 2018; Icarte et al. 2018; De Giacomo et al. 2019b). Notably, both LTL_f and LDL_f formulas can be transformed into deterministic finite state automata tracking the stage of satisfaction of the formulas (De Giacomo and Vardi 2013). This, in turn, allows for transforming an NMRDP with non-Markovian LTL_f/LDL_f rewards into an equivalent MDP over an extended state space, obtained as the crossproduct of the states of the NMRDP and the states of the automaton. This transformation is of particular interest in RL with temporally specified rewards expressed in LTL_f/LDL_f, since it allows to do RL on an equivalent MDP whose (optimal) policies are also (optimal) policies for the original problem, and viceversa (Brafman, De Giacomo, and Patrizi 2018). This provides the basis for our development here.

RL with Restraining Bolts

We are given:

- A **learning agent** modeled by the MDP $M_{ag} = \langle S, A, Tr_{ag}, R_{ag}\rangle$, with transitions Tr_{ag} and rewards R_{ag} hidden, but sampled from the environment.
- A **restraining bolt** $RB = \langle \text{Ł}, \{(\varphi_i, r_i)\}_{i=1}^m\rangle$ where:
  - $\text{Ł} = 2^{\mathcal{F}}$ is the set of possible fluents' configurations (analogously to S denoting the set of configurations of the features available to M_{ag}). Fluents in $\mathcal{F}$ are not among the features that form the states S of the learning agent M_{ag}.
  - $\{(\varphi_i, r_i)\}_{i=1}^m$ is a set of *restraining specifications* with
    - φ_i, an LTL_f/LDL_f formula over $\mathcal{F}$. Each φ_i selects sequences of fluents' configurations $\ell_1, \cdots, \ell_n$ ($\ell_k \in$ Ł) whose relationship with the sequences of states $s_1, \ldots, s_n$ ($s_k \in S$) of M_{ag} is unknown.
    - r_i, the reward associated with φ_i. A reward r_i is assigned to sequences of configurations $\ell_1, \cdots, \ell_n$ satisfying φ_i.

The agent receives rewards based on R_{ag} and the pairs (φ_i, r_i).

Note that, when the episode ends and a new episode is started, a new trace is generated on which LTL_f/LDL_f formulas are evaluated again.

Given a formula φ_i we consider *the* minimal DFA $\mathcal{A}_{\varphi_i} = \langle 2^{\P}, Q_i, q_{i0}, \delta_i, F_i\rangle$ corresponding to formula φ_i. Because of the non-Markovian nature of the rewards coming from the restraining bolt, we need to equip the agent with additional observable features $Q_1 \times \ldots \times Q_m$ corresponding to the states of satisfaction of the formulas $\varphi_1 \ldots \varphi_m$.

The architecture described above is depicted in Figure 1. We observe that while the configuration of fluents $\ell_1, \ldots, \ell_n$ is available to the restraining bolt, this is hidden to the agent. In fact, the agent receives only (an encoding of) the DFA state (q) from the restraining bolt. Therefore, in order to incorporate a bolt, the learning agent needs extend its state representation with just a variable to receive the encoding of q (for instance, an integer variable). Obviously, such an extension will accommodate any bolt.

Problem definition. (An instance of) the *RL problem with* LTL_f/LDL_f *restraining specifications* is a pair $M_{ag}^{rb} = \langle M_{ag}, RB\rangle$, where: $M_{ag} = \langle S, A, Tr_{ag}, R_{ag}\rangle$ is a learning agent with Tr_{ag} and R_{ag} hidden, and $RB = \langle \text{Ł}, \{(\varphi_i, r_i)\}_{i=1}^m\rangle$ is a restraining bolt formed by a set of LTL_f/LDL_f formulas φ_i over Ł with associated rewards r_i. A solution to the problem is a (possibly non-Markovian) policy $\bar{\rho} : (Q_1 \times \ldots \times Q_m \times S)^* \to A$ that maximizes the expected cumulative reward.

Solution. We now give our solution. Given an instance of the problem M_{ag}^{rb}:

1. For every φ_i, compute the equivalent DFA $\mathcal{A}_{\varphi_i}$ (De Giacomo and Vardi 2013)
2. Do RL on the MDP $M_{ag}^{q} = \langle Q_1 \times \cdots \times Q_m \times S, A, Tr'_{ag}, R'_{ag}\rangle$ where:
   - The transition distribution Tr'_{ag} is unknown;
   - The reward R'_{ag}, unknown to the agent, is defined as:

$$R'_{ag}(q_1, \ldots, q_m, s, a, q'_1, \ldots, q'_m, s') = \sum_{i : q'_i \in F_i} r_i + R_{ag}(s, a, s').$$

   - The states q_i of the DFAs $\mathcal{A}_{\varphi_i}$ are progressed correctly by the environment.

Note that this technique will only find Markovian policies of the form: $(Q_i \times \cdots \times Q_m \times S) \to A$. However, this does not prevent to find an optimal solution, as shown by the following theorem.

Theorem 1. *RL with* LTL_f/LDL_f *restraining specifications* $M_{ag}^{rb} = \langle M_{ag}, RB\rangle$ *with* $M_{ag} = \langle S, A, Tr_{ag}, R_{ag}\rangle$ *and* $RB = \langle \text{Ł}, \{(\varphi_i, r_i)\}_{i=1}^m\rangle$ *can be reduced to RL over the*

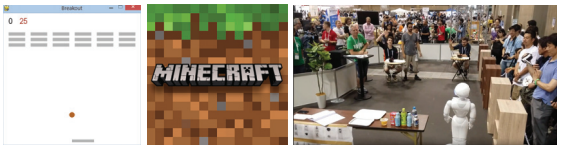

Figure 2: Experimental scenarios: BREAKOUT, SAPIENTINO, COCKTAILPARTY

MDP $M^q_{ag} = \langle Q_1 \times \cdots \times Q_m \times S, A, Tr'_{ag}, R'_{ag} \rangle$ *and optimal policies* ρ^{new}_{ag} *for* M^{rb}_{ag} *can be learned by learning corresponding optimal policies for* M^q_{ag}.

Proof. see (De Giacomo et al. 2019b). □

Use Cases

We describe two use cases for applying restraining bolts: BREAKOUT, MINECRAFT, and the COCKTAILPARTY reported in (De Giacomo et al. 2019b) and (De Giacomo et al. 2019a).

Breakout. BREAKOUT has been widely used to demonstrate RL approaches. The goal of the agent is to control the paddle in order to drive a ball to hit all the bricks in the screen. In this example, we considered two cases where the agent has different abilities: MOVE: the agent moves sideways to bounce the ball; MOVE +FIRE: the agent can both move and fire straight up to remove bricks. Agent's state representation uses the following features: f_x: x position of the paddle; $f_{bx}, f_{by}, f_{dx}, f_{dy}$: position and direction of movement of the ball[3]. Reward is given to the agent when a brick is hit. With this specification a RL algorithm can find a policy to remove all the bricks and complete the game for both the agents.

Restraining bolt. We provide the agents with the following specification: *the bricks must be removed from left to right*, i.e., all the bricks in column i must be removed before completing any other column $j > i$. This specification can be expressed with an LTL$_f$/LDL$_f$ formula and to evaluate such a formula, the bolt needs a representation $f_{r_{(i,j)}}$ of the status of each brick $r_{i,j}$ (present or removed). In both cases, by applying our technique, we can use the same RL algorithm to learn a policy that will complete the task (i.e., remove all the bricks) following the restraining bolt specification (i.e., from left to right).

Notice that the same restraining bolt is applied to the two different agents and they will both learn the behavior specified by the LTL$_f$/LDL$_f$ formula, obviously with different policies. Rows 1 and 2 in Figure 3 show the results of two experiments in the Breakout scenario with the following configurations: Breakout 4x6 MOVE + FIRE (5 minutes), Breakout 4x5 MOVE (1 hour). Left plots show the average reward over the number of iterations, while right plots show the score (i.e., number of columns correctly broken) of the best policy computed so far (i.e., the results obtained in runs without exploration). The figures show how the agent is able to progressively learn how to progress over the states of the DFA corresponding to the LTL$_f$/LDL$_f$ specification. Similar

[3]Other state representations are also suitable to learn the task.

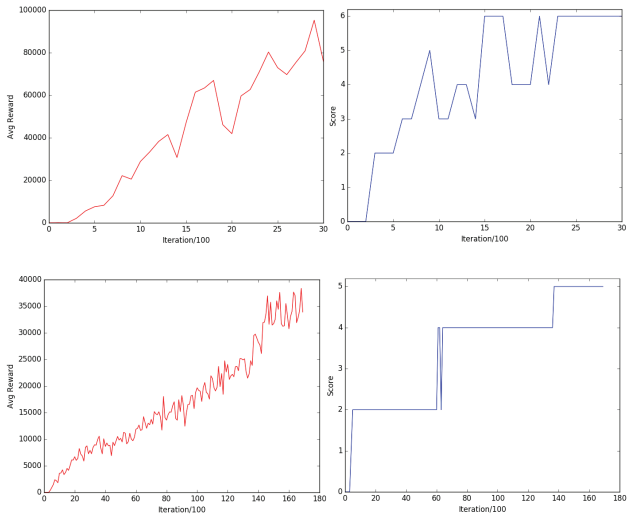

Figure 3: Average reward and scores over number of iterations. Row 1: Breakout MOVE + FIRE 4x6 bricks (5 minutes); Row 2: Breakout MOVE only 4x5 bricks (1 hour).

results are obtained in different configurations (e.g., different sizes of the bricks).

Minecraft. In this Minecraft-like scenario similar to (Icarte et al. 2018), the agent has to accomplish 10 tasks (described with non-Markovian rewards via an LTL$_f$/LDL$_f$ formula). *State representation* is defined by the following features: f_x, f_y, f_θ reporting the pose of the agent in the grid. In this scenario, we consider two different agents: OMNI: omnidirectional movements (actions: up, down, left, right), DIFFERENTIAL: differential drive (actions: forward, backward, turn left, turn right). The agent also has an action *get* to get resources and *use* to use a tool.

Restraining Bolts. As restraining bolts, we have: T_i: *complete task* i; $S1$: *avoid forbidden actions*. A task is a sequence of *getting* resources and *using* tools. A *forbidden action* happens when either the *get* or the *use* are not used properly (e.g. doing an *use* or a *get* when not close to neither a resource nor a tool).

Figure 4 shows experiments in this domain where the OMNI and DIFFERENTIAL agents learned 10 tasks. The meaning of the plots is the same as for the ones commented before, with the score defined as the number of tasks successfully accomplished. These experiments show that a learning agent can learn different tasks (specified in LTL$_f$/LDL$_f$) in different scenarios without changing its internal representation S and learning algorithm, when a suitable component provides to the agent an encoding of the current states of the DFAs of the LTL$_f$/LDL$_f$ formulas. Notably, the results confirm that a *general-purpose* agent can learn several tasks by only receiving information from its restraining bolt.

Cocktail party. For a service robot involved in a cocktail party we consider a representation of the state in terms of robot's pose and objects' (drinks and snacks) and people's location. The agent can move in the environment, grasp and

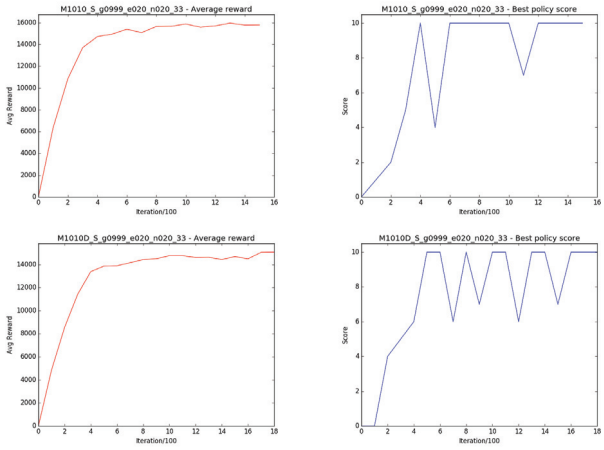

Figure 4: Results in Minecraft. Top: OMNI (5 minutes). Bottom: DIFFERENTIAL (15 minutes).

deliver items to people, and get a reward when a delivery task is completed. The robot has no sophisticated people perception capabilities, and no memory is available in the underlying MDP modeling the domain, so the robot cannot get information about individual people or remember who received what. The robot in this scenario will just learn how to bring one item to any person (choosing the shortest path). *Restraining bolt*. Consider the following specification: *serve exactly one drink and one snack to every person, but do not serve alcoholic drinks to minors*. As in the previous examples, the restraining bolt works on separate features, namely identity, age and received items [4] and uses an $\text{LTL}_f/\text{LDL}_f$ formula to model this specification. We assume the map of the environment to be known, people sitting at tables in predefined known positions and locations of snack and drink items also known. From these information we can instantiate a simulator for the robot to navigate in this environment and reach the different locations

For learning this task, we considered a problem with two people and two different kinds of drinks and snacks (4 tasks to be executed in total) and we implemented an abstract simulator reproducing the scenario of RoboCup@Home competition. The results of learning the restrained task in the simulator are depicted in Row 5 of Figure 3 (score = 4 means that the 2 persons have received one drink and one snack each). As shown, after about 1 minute of simulation

These examples illustrate the effectiveness of the proposed approach for learning tasks specified by $\text{LTL}_f/\text{LDL}_f$ restraining specifications by reducing the NMRDP in an equivalent MDP without changing state representation (except for features for distinguishing the states of DFAs of the $\text{LTL}_f/\text{LDL}_f$ specs) and learning algorithm. More details about the experimental configurations, source code of the implementation allowing forreproducing the results contained in this paper, and videos of the found policies are available in www.diag.uniroma1.it/restraining-bolt.

[4] In practice, services like Microsoft Cognitive Services Face API can be integrated into the bolt to provide this information.

Acknowledgements

This work was partially funded by Università di Roma La Sapienza, under project DRAPE: Data-awaRe Automatic Process Execution.

References

Bacchus, F.; Boutilier, C.; and Grove, A. 1996. Rewarding behaviors. In *AAAI*, 1160–1167.

Brafman, R. I.; De Giacomo, G.; and Patrizi, F. 2018. $\text{LTL}_f/\text{LDL}_f$ non-markovian rewards. *AAAI*.

Brooks, R. A. 1991. Intelligence without representation. *Artif. Intell.* 47(1-3):139–159.

Camacho, A.; Chen, O.; Sanner, S.; and McIlraith, S. A. 2017. Decision-making with non-markovian rewards: From LTL to automata-based reward shaping. In *RLDM*, 279–283.

De Giacomo, G., and Rubin, S. 2018. Automata-theoretic foundations of FOND planning for LTLf and LDL_f goals. In *IJCAI*.

De Giacomo, G., and Vardi, M. Y. 2013. Linear temporal logic and linear dynamic logic on finite traces. In *IJCAI*.

De Giacomo, G.; Iocchi, L.; Favorito, M.; and Patrizi, F. 2019a. Foundations for restraining bolts - demonstration. In *Proceedings of the International Conference on Automated Planning and Scheduling (Demo)*.

De Giacomo, G.; Iocchi, L.; Favorito, M.; and Patrizi, F. 2019b. Foundations for restraining bolts: Reinforcement learning with ltlf/ldlf restraining specifications. In *Proceedings of the International Conference on Automated Planning and Scheduling*, volume 29, 128–136.

Icarte, R. T.; Klassen, T. Q.; Valenzano, R. A.; and McIlraith, S. A. 2018. Teaching multiple tasks to an RL agent using LTL. In *AAMAS*, 452–461.

Littman, M. L.; Topcu, U.; Fu, J.; Jr., C. L. I.; Wen, M.; and MacGlashan, J. 2017. Environment-independent task specifications via GLTL. *CoRR* abs/1704.04341.

Littman, M. L. 2015. Programming agent via rewards. In *Invited talk at IJCAI*.

Mnih, V.; Kavukcuoglu, K.; Silver, D.; Rusu, A. A.; Veness, J.; Bellemare, M. G.; Graves, A.; Riedmiller, M. A.; Fidjeland, A.; Ostrovski, G.; Petersen, S.; Beattie, C.; Sadik, A.; Antonoglou, I.; King, H.; Kumaran, D.; Wierstra, D.; Legg, S.; and Hassabis, D. 2015. Human-level control through deep reinforcement learning. *Nature* 518(7540):529–533.

Puterman, M. L. 1994. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley.

Reiter, R. 2001. *Knowledge in Action*. MIT Press.

Silver, D.; amd Karen Simonyan, J. S.; Antonoglou, I.; Huang, A.; Guez, A.; Hubert, T.; Baker, L.; Lai, M.; Bolton, A.; Chen, Y.; Lillicrap, T.; Hui, F.; Sifre, L.; van den Driessche, G.; Graepel, T.; and Hassabis, D. 2017. Mastering the game of go without human knowledge. *Nature* 550:354–359.

Sutton, R. S., and Barto, A. G. 1998. *Reinforcement learning - an introduction*. MIT Press.

Whitehead, S. D., and Lin, L.-J. 1995. Reinforcement learning of non-markov decision processes. *Artificial Intelligence* 73(1):271 – 306. Computational Research on Interaction and Agency, Part 2.