Foundations for Restraining Bolts: Reinforcement Learning for LTL_f/LDL_f Goals

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

Introduction

In reasoning about action it is becoming of great interest to develop mechanisms that include components with forms of decision making that are based on stochastic models as in Markovian and Non-Markovian Decision Processes (MDPs/NMDPs) and Reinforcement Learning (RL) (Puterman 1994; Sutton and Barto 1998), possibly coupled with deep learning techniques (Silver et al. 2017). However, while the actual execution in such components could be chosen stochastically, we do want to have *safety guarantees* on all possible generated executions (Amodei et al. 2016; Hadfield-Menell et al. 2017).

In this work we aim at starting the development of the foundation for the concept of "*restraining bolt*"[1], as envisioned in science fiction movies. The concept of restraining bolt is based on two distinct representations of the world: one by the agent and one by the authority imposing the bolt. The two representations are different since developed by independent parties. However they both model (aspects of) the same world. We want the mechanism to conform the restraining rules even if these are not expressed in its original representation.

In this work we present some preliminary results and show that if the mechanism's view of the world is an (unknown) MDP and the restraining rules are expressed in LTL_f/LDL_f over a separate (unknown) model of the world that complements the mechanism's one in a Markovian way, then, under general circumstances, the mechanism can learn to act, while conforming to the LTL_f/LDL_f rules. Notice that keeping track of the satisfaction of the LTL_f/LDL_f rules makes the whole system formed by the mechanism and the restraining bolt non-Markovian.

Recently, interest in NMDPs (Bacchus, Boutilier, and Grove 1996; Whitehead and Lin 1995) 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 temporal logics over finite traces has been independently advocated by (Camacho et al. 2017a) and (Brafman, De Giacomo, and Patrizi 2017; 2018). Both research groups propose to use LTL_f, or its more general extension LDL_f, to model temporal properties of dynamic systems (De Giacomo and Vardi 2013; 2015; De Giacomo and Vardi 2016; Baier et al. 2008; Torres and Baier 2015; Camacho et al. 2017b).

The logic LTL_f is the classical linear time logic LTL (Pnueli 1977) interpreted over finite traces, formed by a finite (instead of infinite as in LTL) sequence of propositional interpretations. Instead, LDL_f is a proper extension of LTL_f, which allows to express regular expressions over such sequences, hence mixing procedural and declarative specifications as advocated in some work in Reasoning about Action and Planning (Levesque et al. 1997; Baier et al. 2008).

The crucial point of both LTL_f and LDL_f is that their formulas can be transformed into finite state automata; this, in turn, allows for transforming an NMDP 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 NMDP and the states of the automaton.

When applied to Reinforcement Learning (RL) with non-Markovian rewards rexpressed in LTL_f/LDL_f, the availability of this transformation allows us to do RL on a fully equivalent MDP whose (optimal) policies are also (optimal) policies for the original problem and viceversa (Brafman, De Giacomo, and Patrizi 2018).

Exploiting such results, we study the "*restraining bolt*" case. We assume to have a learning agent (the mechanism) equipped with sensing procedures to compute a set of features from the world that form its states and with a set of actions that it can perform. We want to use this agent to learn one (or simultaneously many) task whose goal(s) are expressed in LTL_f/LDL_f. Such goals are expressed over a representation of the world that is *not* the one used by the agent (oversimplifying, we may say that the agent has a low-level representation), but a convenient high-level representation suitable to express declaratively temporally extended goals. In other words, we study the possibility of having *two separate representations* of the world:

- one for expressing the dynamics of the learning agent;
- one for expressing the LTL_f/LDL_f goals.

[1] "A restraining bolt is a device that restricts a droid's [mechanism's] actions when connected to its systems. Droid owners install restraining bolts to limit actions to a set of desired behaviors." https://www.starwars.com/databank/restraining-bolt

These two representations use different classes of features from the real world: the first includes the features that the agent can directly access, while the second includes the features needed to evaluate the LTL_f/LDL_f goal

For example, consider a robotic paddle playing the BREAKOUT game. The paddle has to drive the ball to hit a wall of bricks. The robotic paddle perceives its position and the position and velocity of the balls. Though it does not perceive the position and the status of the bricks, the environment gives suitable rewards when they are broken.

Now suppose we want to express in LTL_f or LDL_f the goal: break first the columns on the left, then those at the center, and finally those on the right. To express this goal we do need a representation of bricks' position and their status (broken or not) of the LTL_f/LDL_f formula. A plain application of RL algorithms in the equivalent MDP requires the extension of the state space for the learning agent with memory for keeping track of the stages of the goals, as well as the representation of the bricks' positions and their status. While adding memory is not problematic, keeping track of bricks' positions and their status may require sophisticated sensors[2]. Moreover, what if the bricks are too far for the available sensors to be detected?

For this reason we want to keep the representations separated and we study the problem of RL in the case in which the learning agent cannot access the high-level representation used to express the goals. The interest in having separate representations is manifold:

1. The learning agent feature space can be designed separately from the features needed to express the goal, thus promoting *separation of concerns* which, in turn, facilitates the design; this separation facilitates also the *reuse* of representations already available, possibly developed for the standard setting.
2. A reduced agent's feature space allows for realizing *simpler agents* (think, e.g., of a mobile robot platform, where one can avoid specific sensors and perception routines), while preserving the possibility of tackling complex declarative goals which cannot be represented in the agent's feature space.
3. Reducing the agent's feature space may yield a *reduced state space* to be explored by the learning agent.

Clearly, the two separate representations (i.e., the two sets of features) need to be somehow correlated in reality. The crucial point, however, is that in order to perform RL effectively, *such a correlation does not need to be formalized*. In this work, we set this framework and provide proofs and experimental evidence that an learning agent can learn policies that optimize the conformance to the LTL_f/LDL_f goals without including in the state space representation the features needed to evaluate the corresponding LTL_f/LDL_f formula (more details can be found in (De Giacomo et al. 2018).) Using these results, we can envision that once the agent is equipped with the restraining bolt, by simulating in its mind how to act (i.e., applying RL), it will deliberate a course of actions that automatically conform (as much as possible) to the restraining rules.

[2]Notice it may require equipping the robot with better sensors, e.g., replacing an inexpensive Kinect-like device with full-fledged distance lasers.

References

Amodei, D.; Olah, C.; Steinhardt, J.; Christiano, P.; Schulman, J.; and Mané, D. 2016. Concrete problems in AI safety. *CoRR* abs/1606.06565.

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

Baier, J. A.; Fritz, C.; Bienvenu, M.; and McIlraith, S. A. 2008. Beyond classical planning: Procedural control knowledge and preferences in state-of-the-art planners. In *AAAI*.

Brafman, R. I.; De Giacomo, G.; and Patrizi, F. 2017. Specifying non-markovian rewards in mdps using LDL on finite traces (preliminary version). *CoRR* abs/1706.08100.

Brafman, R. I.; De Giacomo, G.; and Patrizi, F. 2018. LTL_f/LDL_f non-markovian rewards. *AAAI*.

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

Camacho, A.; Triantafillou, E.; Muise, C.; Baier, J. A.; and McIlraith, S. 2017b. Non-deterministic planning with temporally extended goals: LTL over finite and infinite traces. In *AAAI*.

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

De Giacomo, G., and Vardi, M. Y. 2015. Synthesis for LTL and LDL on finite traces. In *IJCAI*.

De Giacomo, G., and Vardi, M. Y. 2016. Ltl_f and ldl_f synthesis under partial observability. In *IJCAI*.

De Giacomo, G.; Iocchi, L.; Favorito, M.; and Patrizi, F. 2018. Reinforcement learning for LTLf/LDLf goals. *CoRR* arXiv:1807.06333.

Hadfield-Menell, D.; Dragan, A. D.; Abbeel, P.; and Russell, S. J. 2017. The off-switch game. In *IJCAI*, 220–227.

Levesque, H. J.; Reiter, R.; Lesperance, Y.; Lin, F.; and Scherl, R. 1997. GOLOG: A logic programming language for dynamic domains. *J. of Logic Programming* 31.

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*.

Pnueli, A. 1977. The temporal logic of programs. In *FOCS*.

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

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.

Torres, J., and Baier, J. A. 2015. Polynomial-time reformulations of LTL temporally extended goals into final-state goals. In *IJCAI*.

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.