Poster Abstract: Structured Reward Functions using STL

Anand Balakrishnan
anandbal@usc.edu
University of Southern California
Los Angeles, CA, USA

Jyotirmoy V. Deshmukh
jdeshmuk@usc.edu
University of Southern California
Los Angeles, CA, USA

ABSTRACT

In this work we present a new method for shaping reward functions to train reinforcement learning agents using signal temporal logic (STL) formulas. The proposed approach uses the robustness metric of partial signal traces against STL specifications to generate *locally shaped rewards*, doing this in a manner that is agnostic of the learning algorithm used by the reinforcement learning agent.

CCS CONCEPTS

• **Computing methodologies** → **Reinforcement learning**; • **Theory of computation** → **Modal and temporal logics**;

ACM Reference format:
Anand Balakrishnan and Jyotirmoy V. Deshmukh. 2019. Poster Abstract: Structured Reward Functions using STL. In *Proceedings of 22nd ACM International Conference on Hybrid Systems: Computation and Control, Montreal, QC, Canada, April 16–18, 2019 (HSCC '19),* 2 pages.
https://doi.org/10.1145/3302504.3313355

1 INTRODUCTION

Recent advancements have shown that reinforcement learning (RL) combined with deep learning can solve highly complex problems, from maximizing high scores in Atari games to learning gait in simulated robots [9, 10, 12]. The ability of deep learning so approximate highly non-linear functions has enabled deep reinforcement learning controllers to optimize policies that maximize rewards using just sensory input, like joint/actuator states in robot simulations. An important problem to address when designing and training reinforcement learning agents is *reward shaping*[5, 13]. Reinforcement learning agents learn policies by iteratively acting on and observing their environment, and for each step performed, they receive a reward, usually through a hand-crafted reward functions. Reward shaping refers to the design of these reward functions.

Reward functions are a way to incorporate domain knowledge in training reinforcement learning agents, and are especially important in model-free learning methods like Q-Learning, DQN, and various other recently developed deep RL models [5, 10]. Thus, reward shaping is an important aspect of reinforcement learning as poorly designed reward functions can lead to poor convergence of the policy. Moreover, in the case of safety critical systems, the agent can learn a policy that performs unsafe or unrealistic actions, even though it maximizes the expected total reward [6].

HSCC '19, April 16–18, 2019, Montreal, QC, Canada

ACM ISBN 978-1-4503-6282-5/19/04.
https://doi.org/10.1145/3302504.3313355

Meanwhile, research on safety and verification of cyber-physical systems (CPS) has extensively used Temporal Logics to define safety specification on the system. Specifically, Signal Temporal Logic (STL) has been used to define temporal properties of signals generated by cyber-physical systems in an expressive manner. Moreover, the quantitative semantics of STL allow us to detect how robustly a signal satisfies a given property, that is, we can calculate how robust a signal is to perturbation against a given STL formula [4].

Our work borrows from the ideas used by the CPS community and leverages it to define reward functions for reinforcement learning system similar to ideas like [1, 8]. Specifically, we use STL specifications defined for an agents learning environment to compute *locally shaped reward functions*. We do this by computing the robustness of partial signal traces generated by the RL agent acting on an environment and perform the learning step in the RL agent in a delayed manner. Our objective is to provide a framework by which reward functions can be specified in an expressive manner using STL formulas, and use these reward functions regardless of the architecture of the reinforcement learning agent.

2 STRUCTURED REWARDING

When monitoring signals for satisfaction of STL specifications, typically there are two ways to do it:

- The more common method is where to let the system run and reach a terminal state, and compute the robustness of the complete signal [3, 4, 11]. This is similar to Monte Carlo methods, and provides the most accurate way to compute how robust a signal is to perturbation.
- The other method is to compute the online robustness metric. This is an approximation as information in the future is not available to compute the STL satisfaction for future temporal operators like $Always$, $Eventually$ and $Until$ [2, 7].

There are disadvantages in using either notion of robustness in the context of reward shaping for RL: Using the classical robustness metric requires us to wait for the training episode to end to calculate rewards, while using the online robustness metric gives rise to rewards that accommodate an arbitrarily long history of the learning episode. The latter is especially incompatible with RL frameworks such as N-step learning [9, 13].

Definition 2.1 (Partial Signal). Given a trajectory of a system $\mathbf{x}$, a partial signal, $\mathbf{x}[i:j]$, is defined by the slice of the trajectory from the i^{th} sample to the j^{th} sample.

To generate a partial signal, we define a buffer that stores the transitions experienced by the RL agent and use the stored states to generate a partial signal when the buffer is full. The maximum size of the buffer is set as an hyper-parameter, τ, for training the RL agent.

Definition 2.2 (Robust Satisfaction Function). The robust satisfaction function is a function $\tilde{\rho}$ that maps a STL property φ, a partial signal $\mathbf{x}[\ldots]$, and a sample step $n \in \mathbb{N}$ to a real number in $R \subseteq \mathbb{R}$.

The robust satisfaction function computes the degree to which a signal is robust to perturbation from a given time step. This means that a signal with a large positive valued robust satisfaction metric satisfies a formula φ better than one with a lower robust satisfaction value. Thus, we can define the reward function for a specific sample as a function of the robust satisfaction values at each point in the signal using the following equation:

$$\mathbf{R}[i] = \tilde{\rho}(\varphi, \mathbf{x}[i:\tau]) \tag{1}$$

where, τ is the length of the partial signal and φ is the STL specification defined on the system.

By doing so, we encode the local satisfaction of an STL property into the reward function for an RL agent, where each reward expressively holds information about satisfaction in future states.

3 RESULTS

To evaluate the new framework, we train Double DQN agents to solve the cart-pole problem, using traditional reward functions for one set and STL reward functions for the rest. In the case of the vanilla rewarding function, we provide the agent with a reward of +1 for every time-step it successfully balances the pole, i.e., $\theta \in [-5^\circ, 5^\circ]$. For the structured reward function, we define the STL property φ that we want the agent to satisfy as

$$\varphi = G\Big(F\left(|\dot{x}| < 0.01\right) \wedge \left(|\theta| < 2^\circ\right) \wedge \left(|x| < 0.5\right)\Big)$$

where $\dot{x}$ is the velocity of the cart, x is the displacement of the cart from the center of the table, and θ is the angle of the inverted pendulum.

In Figure 1, we can see that the Double DQN agent trained using the STL-based reward function improves the duration for which the pole is balanced (max of 500 time-steps), while continuing to improve the total reward obtained. This rewarding strategy can be extended just as easily to actor-critic models for continuous control systems, allowing us to define STL properties on humanoid robots, and other safety-critical systems.

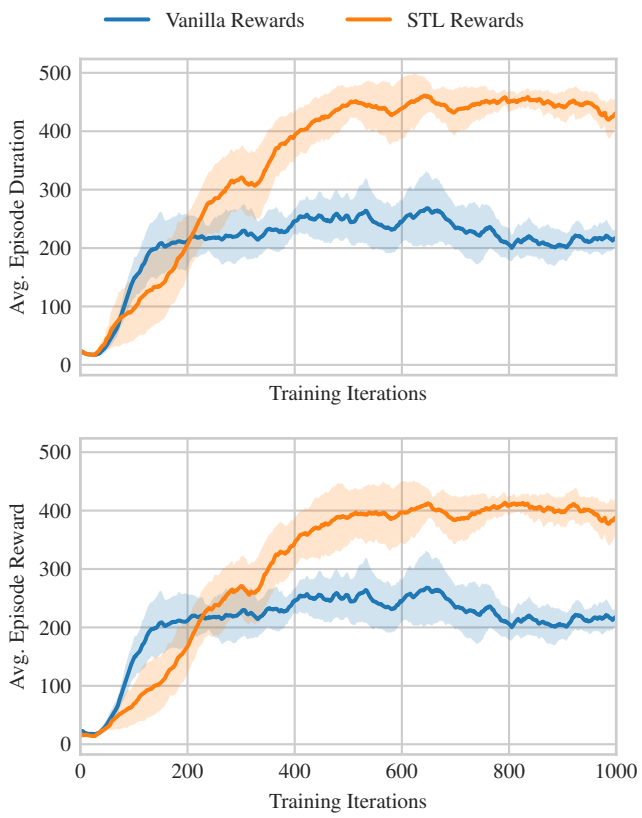

Figure 1: Double DQN training data for cart-pole problem

ACKNOWLEDGMENTS

This work was funded in part by a grant from Toyota Motors North America R&D. Computation for the work described in this paper was supported by the University of Southern CaliforniaâĂŹs Center for High-Performance Computing (hpc.usc.edu).

REFERENCES

[1] Derya Aksaray, Austin Jones, Zhaodan Kong, Mac Schwager, and Calin Belta. 2016. Q-Learning for Robust Satisfaction of Signal Temporal Logic Specifications. (Sept. 2016). arXiv:cs/1609.07409

[2] Jyotirmoy V. Deshmukh, Alexandre Donzé, Shromona Ghosh, Xiaoqing Jin, Garvit Juniwal, and Sanjit A. Seshia. 2017. Robust Online Monitoring of Signal Temporal Logic. *Formal Methods in System Design* 51, 1 (Aug. 2017), 5–30. https://doi.org/10.1007/s10703-017-0286-7

[3] Alexandre Donzé, Thomas Ferrère, and Oded Maler. 2013. Efficient Robust Monitoring for STL. In *Computer Aided Verification (Lecture Notes in Computer Science)*, Natasha Sharygina and Helmut Veith (Eds.). Springer Berlin Heidelberg, 264–279.

[4] Alexandre Donzé and Oded Maler. 2010. Robust Satisfaction of Temporal Logic over Real-Valued Signals. In *International Conference on Formal Modeling and Analysis of Timed Systems*. Springer, 92–106.

[5] Marek Grześ. 2017. Reward Shaping in Episodic Reinforcement Learning. In *Proceedings of the 16th Conference on Autonomous Agents and MultiAgent Systems (AAMAS '17)*. International Foundation for Autonomous Agents and Multiagent Systems, 565–573.

[6] Shixiang Gu, Timothy Lillicrap, Ilya Sutskever, and Sergey Levine. 2016. Continuous Deep Q-Learning with Model-Based Acceleration. (March 2016). arXiv:cs/1603.00748

[7] Stefan Jakšić, Ezio Bartocci, Radu Grosu, Thang Nguyen, and Dejan Ničković. 2018. Quantitative Monitoring of STL with Edit Distance. *Formal Methods in System Design* 53, 1 (Aug. 2018), 83–112. https://doi.org/10.1007/s10703-018-0319-x

[8] X. Li, C. Vasile, and C. Belta. 2017. Reinforcement Learning with Temporal Logic Rewards. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 3834–3839. https://doi.org/10.1109/IROS.2017.8206234

[9] Volodymyr Mnih, Adrià Puigdomènech Badia, Mehdi Mirza, Alex Graves, Timothy P. Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. 2016. Asynchronous Methods for Deep Reinforcement Learning. (Feb. 2016). arXiv:cs/1602.01783

[10] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-Level Control through Deep Reinforcement Learning. *Nature* 518, 7540 (Feb. 2015), 529–533. https://doi.org/10.1038/nature14236

[11] Alena Rodionova, Ezio Bartocci, Dejan Nickovic, and Radu Grosu. 2016. Temporal Logic as Filtering. *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control - HSCC '16* (2016), 11–20. https://doi.org/10.1145/2883817.2883839 arXiv:1510.08079

[12] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. 2017. Proximal Policy Optimization Algorithms. (July 2017). arXiv:cs/1707.06347

[13] Richard S. Sutton and Andrew G. Barto. 2018. *Reinforcement Learning: An Introduction* (second edition ed.). The MIT Press, Cambridge, MA.