Synthesising Reward Machines for Cooperative Multi-Agent Reinforcement Learning

G. Varricchione N. A. Alechina M. M. Dastani B. S. Logan

Information and Computing Sciences, Utrecht University

Highlights

Objective

- Automatically synthesise Reward Machines (RMs) to train agents independently in a multi-agent reinforcement learning (MARL) setting

Methodology

- Abstract the low-level environment using an Epistemic Concurrent Game Structure (ECGS)
- Synthesise a joint plan from an Alternating-time Temporal Logic (ATL) specification
- Transform the plan into an RM, and automatically decompose it into individual RMs to train agents independently as in [1]

Reward Machines

- Reward machines are a tool recently introduced in the reinforcement learning literature [2] to specify non-Markovian reward functions
- An RM is a Mealy automaton, where to each transition between states a reward is associated
- At each timestep, the RM transitions to a new state based on the high-level event that it has observed and gives a reward to the agent(s)

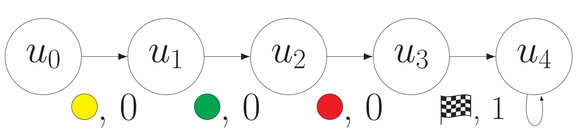


Figure: Example of an RM for the CooperativeButtons environment.

The Neary et al. Approach

- In the context of MARL, [1] propose (under certain assumptions) to train agents independently
- Each agent observes only a subset of events from the environment, needed to complete its subtask
- By projecting the team's RM onto the set of observables of each agent, we obtain an RM for each of them, which can then be used to train them independently

Alternating-time Temporal Logic

ATL [3] is a logic used to specify the high-level behaviour of agents in a multi-agent system. It is derived from Linear-time Temporal Logic by adding so-called "coalitional modalities" $\langle\langle A \rangle\rangle$ in front of temporal formulas.

$$\varphi ::= p \mid \neg\varphi \mid \varphi_1 \vee \varphi_2 \mid \langle\langle A \rangle\rangle \bigcirc \varphi \mid \langle\langle A \rangle\rangle \Box \varphi \mid \langle\langle A \rangle\rangle \varphi_1 \mathcal{U} \varphi_2$$

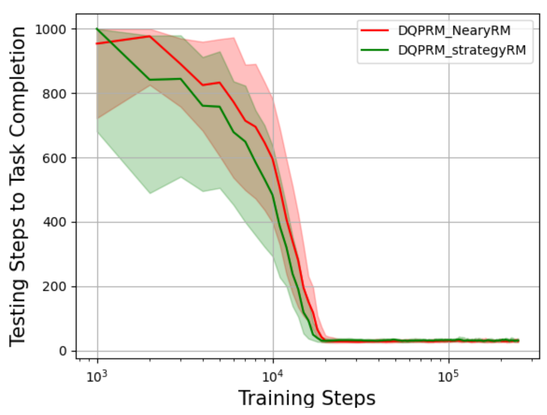
- We use ECGSs as models for ATL to model imperfect information for agents, as it is done in [1]
- A coalitional formula $\langle\langle A \rangle\rangle \varphi$ is true in a state s of the ECGS iff coalition A has a strategy to enforce φ starting from s
- Since we have imperfect information, these strategies must be "feasible" in the sense that if an agent does not distinguish two states then it must perform the same action in those according to the strategy

Synthesising RMs through ATL

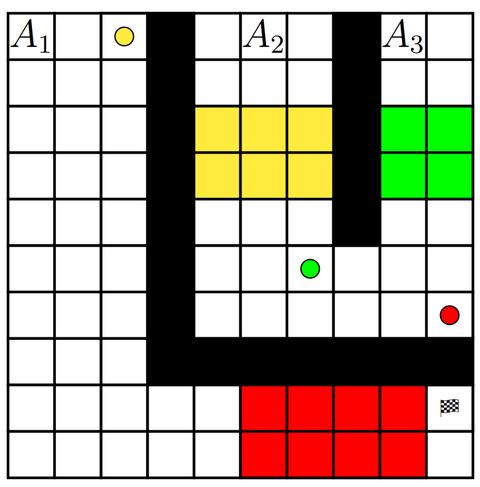
- We are given an ECGS that abstracts the low-level environment in which the agents act
- By specifying the task and behaviours we want the agents to learn through an ATL formula, we can then use the model checker MCMAS [4] to obtain a high-level strategy
- The high-level strategy can be easily transformed into a team RM, on which we can then use the approach of [1] off-the-shelf to train the agents independently in the low-level environment

Evaluation

We evaluate our synthesised RMs against the hand-crafted ones of [1] in the two environments they propose, CooperativeButtons and Rendezvous. In the plots, lines represent median performance of the agents, whereas the shaded areas the 25^{th} and 75^{th} percentiles.

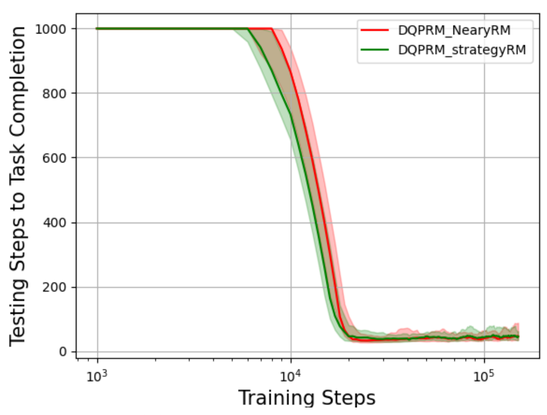


(a) CooperativeButtons evaluation

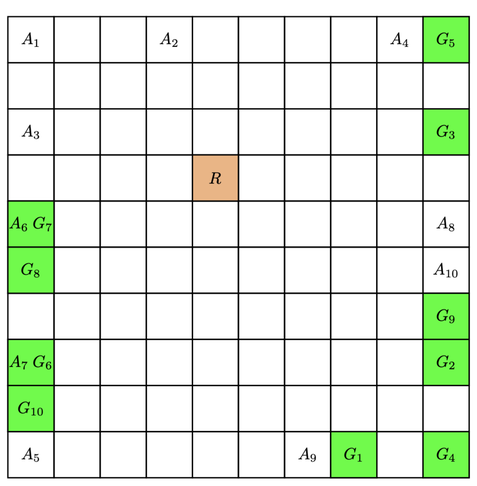


(b) CooperativeButtons domain

Figure: CooperativeButtons



(a) Rendezvous evaluation



(b) Rendezvous domain

Figure: Rendezvous

Future Work

- Instead of generating a single strategy, generate all possible strategies to accomplish the task so that agents are more "versatile" once they are trained
- Consider a non-cooperative setting, i.e., one where there are agents that are not aligned with the coalition of choice. This can be easily done through ATL as coalitional modalities can be defined over any subset of agents
- Extend the approach to support also ATL* – this would allow the expression of more flexible tasks and to generate strategies for several temporal formulas simultaneously

Conclusion

We have

- Presented an approach to synthesise RMs in a multi-agent setting
- By employing ATL, we have also provided a way to easily specify safety constraints, via formulas of the form $\langle\langle A \rangle\rangle \Box \varphi$ or $\langle\langle A \rangle\rangle \varphi \mathcal{U} \psi$
- Performed experiments that show that our RMs perform comparably, if not better, than the hand-crafted ones of [1]

References

[1] Cyrus Neary, Zhe Xu, Bo Wu, and Ufuk Topcu. Reward machines for cooperative multi-agent reinforcement learning. In *Proc. of the 20th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS 2021)*, pages 934–942, 2021.

[2] Rodrigo Toro Icarte, Toryn Q Klassen, Richard Valenzano, and Sheila A McIlraith. Reward machines: Exploiting reward function structure in reinforcement learning. *Journal of Artificial Intelligence Research*, 73:173–208, 2022.

[3] Rajeev Alur, Thomas A. Henzinger, and Orna Kupferman. Alternating-time temporal logic. *J. ACM*, 49(5):672–713, 2002.

[4] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. MCMAS: An open-source model checker for the verification of multi-agent systems. *Int. J. Softw. Tools Technol. Transf.*, 19(1):930, 2017.

Code

github.com/giovannivarr/SynthesisingRMsMARL