Influence-Focused Asymmetric Island Model

Andrew Festa
Oregon State University
Corvallis, United States
festaa@oregonstate.edu

Gaurav Dixit
Oregon State University
Corvallis, United States
dixitg@oregonstate.edu

Kagan Tumer
Oregon State University
Corvallis, United States
ktumer@oregonstate.edu

ABSTRACT

KEYWORDS

Multiagent Learning, Evolutionary Learning

1 BACKGROUND

1.1 Cooperative Co-Evolution

Cooperative Coevolutionary Algorithms (CCEAs) make it possible to apply an evolutionary approach to learning joint-policies for a multiagent system. Whereas an Evolutionary Algorithm (EA) evolves a population of policies for a single agent, a CCEA co-evolves several sub-populations of policies, one sub-population for each agent in the multiagent system. This essentially splits the learning problem into several sub-problems, where learning each agent's policy is a specific sub-problem. A primary consequence of splitting up the learning problem this way is that evaluating fitnesses becomes more complex, as a policy can only be evaluated as part of a joint-policy, not on its own. As such, a policy is always evaluated in the context of how it performs with policies from other sub-populations.

1.2 Temporal Abstractions

A promising approach of designing systems for sequential task coordination relies on augmenting the agents with the ability to perform behaviors rather than actions [11, 13, 17, 21]. In a generalized form, a behavior, sometimes called a temporally extended action (e.g. macro action), is closely related to a sequence of actions with the added idea that it is meant to exhibit progress towards some goal. For instance, a behavior may be for an agent to "pick up a ball" or "move towards a door." However, these methods generally differ in how the behaviors are designed and how they are used during execution.

1.2.1 Option Learning. Options were designed with the idea that each action need not be a single time step, but may be a closed-loop policy executed for a desired number of steps [17]. In this way, they are a form of temporal abstraction in that they allow for an action to operate over an extended period of time and reduce the difficulty in assigning rewards to temporally distant actions. However, the original formulation left open how these extended actions could actually be used in a learning framework [15]. One way this is addressed is through the insight that intermediary states may be sub-goals of a task, particularly those that are frequently visited [12]. [12] uses this insight to identify the key points in states that may be useful to visit in order to accomplish a desired task.

Additionally, many approaches that use options only update a single option at a time, rather than analogously updating the value of many actions at once. Updating multiple options simultaneously is challenged by that a single option is not a single action, and the agent actions are not discretized along those action sequences. That is, one options may contain part of another option and options may be different number of steps. [6] addresses this through use of a option-critic to extend intra-option learning [16] which allows them to update multiple options at once in the context of a deep reinforcement learning algorithm.

It is also worth noting that this type of formulation allows for more easily reusing prior experience for lifelong learning [2, 22]. Research regarding options typically looks at how to automatically build options [19, 20], how to identify bottleneck states [8], which are those that connect different connect regions of a state space (e.g. doorways) [4, 14], or how to balance the exploration-exploitation trade-off common in reinforcement learning [5]. But in building these extended actions, they more readily allow for transferring the capabilities of the agent to new solve new, more complex tasks.

One of the limitations with options learning is a necessity for executing a chosen policy for a specified number of time steps. Additionally, it generally operates in a reinforcement learning paradigm that makes extensive use of the gradients both to learn the behaviors and to optimize the agents to complete a task. Finally, options research is generally limited to a single-agent learning problem, though there is some work towards extending this framework to the multiagent case [3, 7].

1.2.2 Dynamic Skill Selection. An alternate approach to sequential decision making, particularly in a multiagent context, is presented as dynamic skill selection [13, 21, 23]. This approach makes use of a multiagent evolutionary paradigm to form effective joint strategies among agents. It makes an explicit effort to handle the issue that arise in multiagent learning, especially the structural credit assignment problem discussed earlier. Over the course of an episode, the agents learn to optimize local skills using policy gradients during an episode and uses an evolutionary learner to optimize the delayed rewards from the agents switching between these local rewards over a single episode.

Both MAEDyS [13] and Multi-fitness learning (MFL) [21] use a similar idea as the options framework in that they seek to form a behavioral abstraction about the capabilities of an agent. However, where MAEDyS switches between learned skills to optimize the team fitness, MFL leverages multiple fitness functions to learn which fitness is best to maximize at any given time. MFL alleviates the challenges of agents learning to cooperate to achieve complex, sequential tasks by having the agents focus on learning *which objective matters when*. This has an added benefit of being extremely robust to changes in the environment, population size, and task complexity. It is important to distinguish this approach from multi-objective learning, as each skill is maximizes a single objective, and during training, the system seeks to satisfy a single objective at any given point.

1.3 Challenges in Multiagent Learning

Multi-agent learning shares many of the issues inherent in single agent learning, Along with a set of challenges that are unique to the. Multiple agents simultaneously learning in a domain. In both single and multi agent learning, the systems are challenged in the presence of sparse rewards, which simply make it. For agents to learn which actions. Might lead to a positive feedback. Additionally, in the single agent setting. the agent can be challenged by a high dimensional state-action space. However, when you move to a multi agent setting, this problem becomes inherently more difficult, simply because each agent has their own state-action space. In essence this is the curse of dimensionality that is not specific to reinforcement, evolutionary, or multiagent learning, but nevertheless, it still impacts the learnability of the problem just as does in other domains of machine learning.

Non-Stationarity. Another key difficulty in a multiagent setting is due to the assumption of the system being represented as a Markov Decision Process. Specifically, the issue arises due to the state-transition function not being dependent on the action of a given agent. Instead, it is dependent on the joint-actions of all the agents in the system. Thus, from the perspective of any agent in the system, the environment appears to be non-stationary [1]. While even independent learners can yield powerful results in such domains [9, 10], this non-stationarity invalidates the theoretical convergence guarantees in single-agent reinforcement learning [18].

2 INFLUENCE-FOCUSED ASYMMETRIC ISLAND MODEL

In this work, we introduce the Influence-Focused Asymmetric Island Model, an extension of the Asymmetric Island Model that aims to improve the efficiency of agents learning to interact with other types of agents in environments that are asymmetrically coupled. This is done by making two alterations to the base asymmetric island model. First, while each island is still tasked with optimizing a particular type of agent based on an agent-specific skill, the population of learning agents are not the only population present on the island. The island also contains a population of non-learning agents of each other type of agent. The second change is we allow for periodic inter-island migrations based on some migration criteria.

2.1 Island Optimization

Before migrating a population, each sub-island is tasked with optimizing a particular type of agent based on an agent-specific objective. In this work, there is a single type of sub-island solely responsible for optimizing a specific type of agent. The population of learning agents is then optimized using a CCEA optimizer with 50 agents in each population that are selected for evaluation using a hall of fame selection. The non-learning agents act in the environment just the same as the learning agents. However, they are never mutated or altered between generations in the CCEA loop. In effect, as there are only just enough non-learning agents to fill out the requirements of the environment, the populations of non-learning agents act as champions of their agent type. Thus, the learning agents are always learning against amongst a stationary set of non-learning agents of other types. While they other learning agents acting simultaneously in the system also contributes to the non-stationarity challenge present in multi-agent systems, the population of non-learning agents being static helps alleviate that issue.

2.2 Island-Island Migrations

Along with allowing for the presence of non-learning agents on each island, we also introduce inter-island migrations. Instead of only migrating population to and from the mainland, each island also migrates a set of candidate solutions to other islands that are learning to optimize agent specific rewards. As previously discussed, the asymmetric island models looks to train a population of candidates on separate islands that are then sent to a mainland to learn to effectively collaborate on some team objective. While this works well for asymmetric agents whose individual tasks or behaviors are able to be learned and accomplished independently of one another, it struggles when these behaviors require, or benefit, from the direct contribution of multiple types of agents, or where another type of agent opens up a set of possibilities that may not be present to an agent without the presence of the other asymmetric agent. The asymmetric island model may be able to eventually able to learn these types of behaviors, but it requires multiple island-mainland-island migrations for one sub-island to cause an evolutionary pressure on another sub-island.

A key factor of these inter-island migrations is what criteria is used to determine when an island will perform the migrations, called the migration schedule. If this migrations occurs too frequently, then it is essentially akin to a typical CCEA implementation. Each island is learning with agents that are all simultaneously learning, and so this process would not aid in reinforcing agent-specific skills or alleviate any non-stationarity issues in multi-agent systems. However, if the islands perform these migrations too infrequently, then while the island are still learning in the presence of non-learning agents, they are not as able to make effective use of the behaviors of other agents when directly optimizing their agent-specific tasks.

As a balance between these two extremes, in this work, this migration schedule is defined as shown in equation **??**, where an island migrates its learning population to all neighbor islands if the $m_i = i$, where m_i is a migration index defining the generations when a migration occurs, and i is current generation on the island.

$$m_i = \lfloor \frac{i}{50} \rfloor * 25 + 50 \tag{1}$$

This acts as a decaying schedule where the time between migrations increases as the optimization process goes on. In this work, the first migration occurs after 50 generations. The next occurs at generation 125, an additional 25 generations more than the time it took for the first generation. This pattern then continues until the entire optimizer has finished.

3 HARVEST ENVIRONMENT

In this work, we use a version of the continuous rover domain problem that has been altered to allow for asymmetric interactions between agents. Notably, this includes introducing multiple types of points-of-interests (POIs) in the form of resources and obstacles.

These can be observed by different types of agents with differing capabilities and tasks. Harvesters are capable of collecting resources while excavators are capable of removing obstacles.

3.1 Environmental Dynamics

Past versions of the continuous rover domain problem have explored how introducing tight-coupling makes the learning problem more challenging, and what can be done to guide learning in such problems. This serves as a form of observational coupling where multiple agents must simultaneously observe a POI to collect a positive system reward. This makes the learning problem more difficult as multiple agents must cooperate at particular times, meaning that this feedback is tied not only to the actions of a singular agent, but also to the actions of other agents in the system.

In this alteration of the environment, we are primarily concerned with how the agents respond to increasing detrimental environmental dynamics that can only be removed by other types of agents. That is, a harvester must rely on an excavator to remove an obstacle that may be blocking the path to a resource. Rather than tight coupling across agent observation, we explore tight coupling across asymmetric agent interactions and across temporal separation, where one type of agent must coordinate with another type of agent in order to make progress towards its task. The temporal component comes from feedback not being given when a beneficial action is done by a different type of agent. Instead, this interaction must occur, then the agent must continue on to ultimately collect the correct type of POI before the positive feedback is given to the agent.

The interesting portions of this environment come when considering how agents interact with POIs. As previously mentioned, the goal of the agents is to collect the corresponding type of POI for its type: harvester are incentivized to collect resources and excavators are incentivized to collect obstacles. However, there are additional adverse effects when one type of agent attempts to collect the wrong type of POI. e.g. a harvester attempting to collect an obstacle. When a harvester is near an obstacle, it's movement is reduced, and when an excavator is near a resource, its movement is likewise reduced. This is meant to force the agent types to distinguish between different types of POIs and determine which are beneficial or detrimental to it.

3.2 State Configuration

At the start of every training episode, the agents and POIs are distributed random throughout an unbounded space. However, this is constrained to ensure that every resource and obstacle is reachable by every other agent in less than 75% of the episode length. Additionally, agents are more likely to be distributed towards the center of the space while obstacle and resources are more likely to be distributed towards the edges of the reachable area, with obstacles having a slightly lower distributional radius than resources.

Figure 1 shows an example starting configuration of agents and POIs in the system. Green dots correspond to harvesters, blue dots correspond to excavators, black dots correspond to obstacles, and red dots correspond to resources.

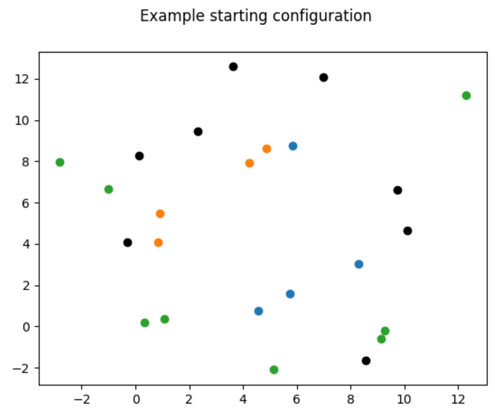


Figure 1: An example of agent and POI distributions at the start of an episode. The green dots represent harvesters, the blue dots excavators, the black dots obstacles, and the red dots POIs. While they are all randomly distributed around the center of the space, the agents are more likely to be more centrally located than the POIs. This is meant to try and create situations where agents are more likely to require depending on other type of agents to accomplish their agent-specific task.

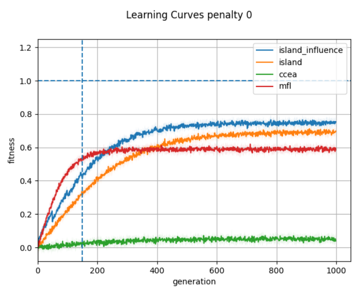
4 EXPERIMENTAL RESULTS

In our experiments we examine how our approach compares against several baselines: CCEA, MFL, and Asymmetric Island Model. CCEA and MFL act as two extremes of the temporal abstraction framework that we bound ourselves within. Additionally, for these experiments, MFL is provided with pre-trained behaviors that are expected to be beneficial to this task. Specifically, these behaviors are to go towards, or away from, the densest region of each type of object. As there are 4 types of objects in the system, this translates to each MFL agent having access to 8 behaviors at the start of training. Note that the size of the action space does not depend on the number of regions the agent uses for its state space. Instead, it selects the region with the highest density. The action is then to go towards, or away from, that direction.

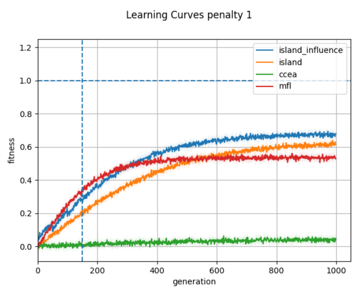
4.1 Team Objective Performance

When examining the performance of the approach, we look not just at how well did the system learn, but also how long did learning take. in these experiments there are eight harvesters, 8 ups, 8 excavators, sixteen obstacles and eight resources, distributed as discussed in the previous section. in such a scenario, it is expected that the harvesters must heavily rely on the excavators to remove any potential obstacle in its way otherwise. it may be hindered to collect resources gathered around the environment.

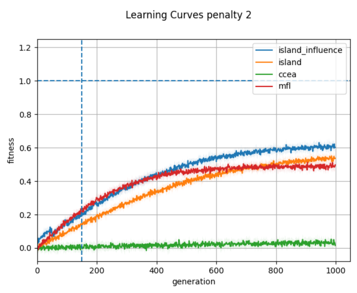
When we look at the configuration where there is no negative reward directly associated with colliding with an obstacle, MFL is able to quickly learn to optimize the team objective because the only negative effect of a collision is a slowing of the agents movement. In such a scenario, it might actually be beneficial to move through the hazardous regions because it allows the harvester to collect the



(a) cap a



(b) cap b



(c) cap c

Figure 2: The proper caption

resources more efficiently. Since MFL is directly selecting between these sets of behaviors, it's able to learn this much more quickly. However, when we introduce an actual penalty for these kind of incursions, we see that MFL degrades in 2 ways rather notably. First, it's learning speed degrades much more than either of the other two approaches, along with the degradation in performance that we see in both the island-based approaches. Specifically, we see this when we look at the influence-focused island model. Not only does it start to about match the speed of MFL, it is able outperforms all of the other methods after 1000 generations.

An important detail to consider is that the MFL behaviors were expected to be beneficial in this environment. Thus, it was expected that MFL would perform the best in all the scenarios while reaching its maximum performance quickly. However, we can see that this isn't necessarily the case, and it's partly because of the behaviors. Recall that all of the behaviors are based on the observation layer of a single type of agent, meaning that MFL was not able to make a decision based on a state that simultaneously considers multiple types of objects. This highlights that while one set of behaviors may be effective in a task, if we alter the task or environment, even just slightly, these behaviors may become less useful. So designing these behaviors can be a difficult design problem, whereas while the other approaches may not learn as quickly in some scenarios, they are at least able to perform more consistently across variations of the same environment.

4.2 Difference in Early Learning

When we zoom in on the learned policies before any approach has finished learning, we can examine an interesting characteristic that emerges in the distributions of harvesters when placed in an enclosing circle of obstacles. Figure 3 shows this distribution, where the black dots are obstacles and the green dots are harvesters. These figures were examined using harvesters trained with a collision penalty of 2.

The harvesters trained using CCEA do not seem to have learned anything useful, as can also be corroborated with the fitness curve previously shown. They have moved a bit from their starting configuration. In this particular scenario, this is not such a bad policy as there are no actual rewards to collect, and thus they are also avoiding any potential hazards.

On the opposite side of the temporal abstraction spectrum, the MFL agents are much more willing to navigate towards hazardous regions, which may incur the negative reward. However, the behaviors provided to the agents only choose to go towards a region of highest density of a singular type. Typically, in the environments these agents were trained in, resources often lay in a similar direction as obstacles, relative to the starting location of the harvesters. Additionally, there were often excavators around to remove the hazardous regions.

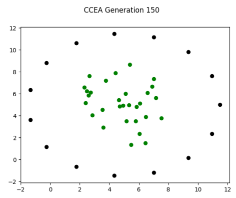
However, in this configuration, there are neither resources to gather nor other excavators to remove the obstacles, causing the MFL agent to make potentially poor decisions based on the training it received from its previous training. Additionally, it highlights the sensitivity of MFL to the efficacy of its provided behaviors. In the overall team objective, these behaviors may be effective behaviors to choose from. However, if the scenario is modified slightly, these behaviors may not readily translate to a new environment.

The asymmetric island agents learns a bit of a conservative policy that looks to set up the agents to move past the obstacles, should a hole open up. Note that due to the observation-space of the agents being divided into the four quadrants around the agent, they do not have a very fine-grained view of the world. And so, a gap might have to be positioned correctly, or the agent must be much closer, before it might realize there is room for it to navigate through two obstacles. This seems like a more appropriate for this environmental configuration from the perspective of the agent. It doesn't see any rewards, and so it doesn't move in such a way as to move into a hazardous region, but it also tries to set itself up in case an excavator were able to remove the obstacles.

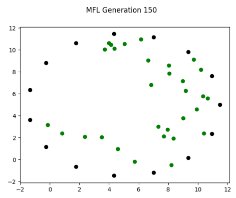
The Influenced-Learning Island agents are making a similar type of positioning as the island agents, although they are slightly more aggressive in their posturing in case an excavator were to come along and remove an obstacle.

5 CONCLUSION

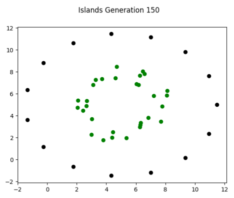
In this work, we presented a framework for learning inter-agent dependencies that arise due to environmental dynamics as related to the desired objective. By allowing for distributed optimization processes that intermittently incorporate the learning of other types of agents, the non-stationarity issue in multi-agent learning can be alleviated, leading to faster convergence along with more optimal policies as the degree of this reliance is increased compared to a standard cooperative co-evolutionary optimization loop. Additionally, when presented with an altered environment, the agents trained



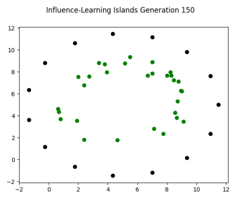
(a) Caption



(b) Caption



(c) Caption



(d) Caption

Figure 3: Three simple graphs

using the influenced-learning island model were able to adapt to the new environment much more quickly that agents trained using other methods.

5.1 Key Findings

5.2 Limitations and Future Work

REFERENCES

[1] Craig Boutilier. 1996. Planning, Learning and Coordination in Multiagent Decision Processes. In *Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge* (The Netherlands) *(TARK '96)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 195–210.

[2] Emma Brunskill and Lihong Li. 2014. PAC-inspired Option Discovery in Lifelong Reinforcement Learning. In *Proceedings of the 31st International Conference on Machine Learning (Proceedings of Machine Learning Research, Vol. 32)*, Eric P. Xing and Tony Jebara (Eds.). PMLR, Bejing, China, 316–324. https://proceedings.mlr.press/v32/brunskill14.html

[3] Jiayu Chen, Jingdi Chen, Tian Lan, and Vaneet Aggarwal. 2022. Learning Multi-agent Options for Tabular Reinforcement Learning using Factor Graphs. https://doi.org/10.48550/ARXIV.2201.08227

[4] Özgür Şimşek and Andrew G. Barto. 2004. Using Relative Novelty to Identify Useful Temporal Abstractions in Reinforcement Learning. In *Proceedings of the Twenty-First International Conference on Machine Learning* (Banff, Alberta, Canada) *(ICML '04)*. Association for Computing Machinery, New York, NY, USA, 95. https://doi.org/10.1145/1015330.1015353

[5] Ronan Fruit and Alessandro Lazaric. 2017. Exploration-Exploitation in MDPs with Options. *ArXiv* abs/1703.08667 (2017).

[6] Martin Klissarov and Doina Precup. 2021. Flexible Option Learning. https://doi.org/10.48550/ARXIV.2112.03097

[7] Xingjie Liu, Guolei Wang, and Ken Chen. 2022. Option-Based Multi-Agent Reinforcement Learning for Painting With Multiple Large-Sized Robots. *IEEE Transactions on Intelligent Transportation Systems* 23, 9 (2022), 15707–15715. https://doi.org/10.1109/TITS.2022.3145375

[8] Marlos C. Machado, Marc G. Bellemare, and Michael Bowling. 2017. A Laplacian Framework for Option Discovery in Reinforcement Learning. https://doi.org/10.48550/ARXIV.1703.00956

[9] Laetitia Matignon, Guillaume J. Laurent, and Nadine Le Fort-Piat. 2012. Independent reinforcement learners in cooperative Markov games: a survey regarding coordination problems. *The Knowledge Engineering Review* 27, 1 (2012), 1–31. https://doi.org/10.1017/S0269888912000057

[10] Georgios Papoudakis, Filippos Christianos, Lukas Schäfer, and Stefano V. Albrecht. 2020. Benchmarking Multi-Agent Deep Reinforcement Learning Algorithms in Cooperative Tasks. https://doi.org/10.48550/ARXIV.2006.07869

[11] Alexander Politowicz and Bing Liu. 2021. Learning to Dynamically Select Between Reward Shaping Signals. https://openreview.net/forum?id=NrN8XarA2Iz

[12] Rahul Ramesh, Manan Tomar, and Balaraman Ravindran. 2019. Successor Options: An Option Discovery Framework for Reinforcement Learning. https://doi.org/10.48550/ARXIV.1905.05731

[13] Enna Sachdeva, Shauharda Khadka, Somdeb Majumdar, and Kagan Tumer. 2021. MAEDyS: Multiagent Evolution via Dynamic Skill Selection. In *Proceedings of the Genetic and Evolutionary Computation Conference* (Lille, France) *(GECCO '21)*. Association for Computing Machinery, New York, NY, USA, 163–171. https://doi.org/10.1145/3449639.3459387

[14] Alec Solway, Carlos Diuk, Natalia Córdova, Debbie Yee, Andrew G. Barto, Yael Niv, and Matthew M. Botvinick. 2014. Optimal Behavioral Hierarchy. *PLOS Computational Biology* 10, 8 (08 2014), 1–10. https://doi.org/10.1371/journal.pcbi.1003779

[15] Martin Stolle and Doina Precup. 2002. Learning Options in Reinforcement Learning. In *Abstraction, Reformulation, and Approximation*, Sven Koenig and Robert C. Holte (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 212–223.

[16] Richard Sutton, Doina Precup, and Satinder Singh. 1998. Intra-Option Learning about Temporally Abstract Actions. 556–564.

[17] Richard S. Sutton, Doina Precup, and Satinder Singh. 1999. Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence* 112, 1 (1999), 181–211. https://doi.org/10.1016/S0004-3702(99)00052-1

[18] Ming Tan. 1997. Multi-Agent Reinforcement Learning: Independent versus Cooperative Agents. In *International Conference on Machine Learning*.

[19] Vivek Veeriah, Tom Zahavy, Matteo Hessel, Zhongwen Xu, Junhyuk Oh, Iurii Kemaev, H. V. Hasselt, David Silver, and Satinder Singh. 2021. Discovery of Options via Meta-Learned Subgoals. In *Neural Information Processing Systems*.

[20] John Winder, Stephanie Milani, Matthew Landen, Erebus Oh, Shane Parr, Shawn Squire, Marie desJardins, and Cynthia Matuszek. 2020. Planning with Abstract Learned Models While Learning Transferable Subtasks. *Proceedings of the AAAI Conference on Artificial Intelligence* 34 (04 2020), 9992–10000. https://doi.org/10.1609/aaai.v34i06.6555

[21] Connor Yates, Reid Christopher, and Kagan Tumer. 2020. Multi-Fitness Learning for Behavior-Driven Cooperation *(GECCO '20)*. Association for Computing Machinery, New York, NY, USA, 453–461. https://doi.org/10.1145/3377930.3390220

[22] Xuejing Zheng, Chao Yu, Chen Chen, Jianye Hao, and Hankz Hankui Zhuo. 2021. Lifelong Reinforcement Learning with Temporal Logic Formulas and Reward Machines. https://doi.org/10.48550/ARXIV.2111.09475

[23] Xiangbin Zhu, Chongjie Zhang, and Victor Lesser. 2013. Combining Dynamic Reward Shaping and Action Shaping for Coordinating Multi-agent Learning. In *2013 IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, Vol. 2. 321–328. https://doi.org/10.1109/WI-IAT.2013.127