Cooperative Foraging in Socially Sequential Dillemas Through Multi Agent Reinforcement Learning

Garvit Jairath
Purdue University
gjairath@purdue.edu

***Abstract*—Cooperation stands as a fundamental facet of intelligence, integral to solving diverse problems. Addressing the imperative need to understand and enhance artificial intelligence's cooperative capabilities, recent years have witnessed significant strides in the realm of multi- agent reinforcement learning (MARL).**

In this study, I delve into the intricacies of how agents grapple with challenges more aptly modeled by socially sequential dilemmas. Through sixteen or more novel experiments, the exploration spans a spectrum of grid sizes and food densities, shedding light on the nuanced responses of agents in these socially intricate scenarios.

The cumulative training duration surpassed thirty hours, utilizing the computational prowess of a T4 GPU. The code, complemented by some of the pretrained weights, is shared openly.

I. Introduction

This study is situated within the framework of Optimal Foraging Theory (OFT), which seeks to mathematically elucidate animal foraging behavior. According to OFT, animals are believed to evolve in a way that maximizes fitness, considering factors such as time and energy expenditure to optimize their foraging activities.

Several models within OFT, such as the patch exploitation model by Charnov and Orians (6), delve into determining the optimal durations for exploiting a foraging patch. Another notable model is the optimal diet theory proposed by Sih and Christensen (8), aiming to predict food choices for optimal foraging efficiency. Individual foraging tends to focus on low-energy foods, while cooperation can potentially yield higher-energy foods, contingent upon successful collaboration. This sets the stage for a Sequential Social Dilemma (8), wherein agents face the decision to forage with their team for increased energy gains but run the risk of gaining nothing without cooperation.

Traditional approaches to address this dilemma involve Stochastic Dynamic Programming (SDP), employing a divide-and-conquer strategy to solve smaller problems independently. However, SDP requires a comprehensive model of the environment, including its state transition probabilities and rewards.

To overcome this limitation, the study advocates for a model-free reinforcement learning method, eliminating the need for an explicit model of the environment. Reinforcement learning (RL) engages directly with the environment, learning a policy through a balanced approach of exploration and exploitation. Building on these insights, the foraging task is formulated as a multi-agent reinforcement learning (MARL) scenario.

In this paper, I present the following contributions:

1) **SSD Exploration:** Explored how MARL algorithms handle socially sequential dilemmas, revealing insights into agent navigation in modified environments.
2) **Varied Experiments:** Conducted 16 experiments across different conditions, providing a comprehensive understanding of MARL algorithm behavior with varied grid sizes and food densities.
3) **GPU Training Rigor:** Demonstrated computational rigor with over 30 hours of T4 GPU training, highlighting the resource-intensive nature of studying socially complex scenarios.
4) **Open Code Sharing:** Shared code and pretrained weights on Colab allowing for more easier calibration of different configurations and future experimentation.
5) **Behavior Insights:** Offering insights, learning curves into the behavior of MARL algorithms in socially sequential dilemmas.
6) **Future Study Foundation:** Identified limitations beyond simple time constraints and proposed more experiments alongside ideas for algorithmic changes MARL-like settings.

II. Related Work

In the ever growing domain of Multi-Agent Reinforcement Learning (MARL) and Sequential Social Dilemmas (SSDs) (8), a collection of notable contributions enrich our comprehension of cooperative behaviors and decision-making among agents.

Recently, a study in MARL by (9) undertakes the bench-marking of typical MARL algorithms in a standard Large Grid Foraging setting. This work distinguishes itself by focusing on bench-marking without introducing modifications to the underlying grid world structure, offering valuable insights into the performance of conventional algorithms in familiar environments.

A recent NeurIPS paper introduces SEAC (12), a novel algorithm emphasizing sharing that outperforms many existing algorithms. This work underscores the significance of collaborative approaches in MARL scenarios, showcasing superior performance through sharing mechanisms. This study also explores the inter algorithmic performances but in traditional LBF settings.

A recent study (14), more focused on the SSD aspect of MARL, proposed the idea of adherence, that is a form of reward engineering to give rewards to agents that adhere to other agents, a way to achieve more communication and cooperation.

Another study (15) attempts to clip, much like the IPPO algorithm, with the use of a surrogate function, the policy of agents to the "status quo" (15) in a similar setting.

While this work closely aligns with the idea of a Socially Sequential Dillema, it attempts to explore, and understand subtle inter-algorithmic performances in the MARL setting by modifying the original LBF environment. The study takes a distinct path by primarily investigating agents employing "commonly-used" MARL algorithms in a mixed motive SSD setting. In this setting, cooperation is not directly incentivised, and can result in interesting behaviors.

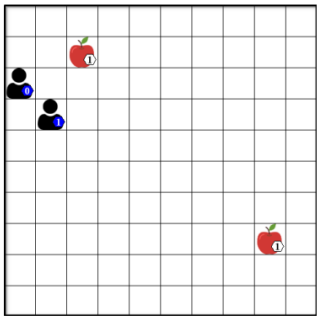


Fig. 1: Level-Based Foraging (LBF) with 2 agents and 2 food items each of level 1 in a 10x10 grid world

III. Background

A. Level Based Foraging

In Level-Based Foraging (LBF), (12) Agents begin in a grid-world setting where their action space includes movement in the four directions and can attempt to load or forage an item only if the sum of the player levels is greater than that of the food item they are adjacent to. This problem can be formulated as a partially Observable Stochastic Game (POSG), a generalisation of Stochastic (Markov) Games (11) to settings where agents are only able to observe parts of the environment's state. A POSG is defined by the ⟨N, S, A, T , R, O, Te⟩ tuple where N is the number of agents, S is the set of all possible states of the environment which in our case is a modified level based foraging. A represents the joint action space where $\mathcal{A} := A_1 \times A_2 \times \ldots \times A_N$ where Ai is the action space of the agent i. T maps the states and actions to create the transition probability matrix, R is defined in a similar fashion as the action space where Ri is the individual reward for any given agent $\mathcal{O} := O_1 \times O_2 \times \ldots \times O_N$ is the set of observations, with Oi representing the individual observation set for each agent i, and $T_e : S \times A \to \Delta(O)$ is the observation function,$T_e(o \mid a, s)$ represents the probability of observing $o \in \mathcal{O}$, given the joint action $a \in \mathcal{A}$ and a new state $s \in \mathcal{S}$ from the environment transition.

An agent's goal is to find a policy π_i which maximizes the expected discounted long-term reward:

$$V^{\pi_i} = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t r_{i,t} \,\middle|\, \pi\right] \quad (1)$$

As per (1), $\pi = (\pi_1, \ldots, \pi_n)$ represents the joint policy of the agents acting in the environment. The discount factor, γ, belongs to the interval $[0, 1]$. T denotes the number of time steps in an episode, and $r_{i,t} = R_i(s_t, a_t, s_{t+1})$ signifies the reward obtained by agent i at time step t. This occurs for the joint action $a_t \in A$, taking place at state $s_t \in S$ and transitioning to the subsequent state $s_{t+1} \in S$.

B. Difference in implementation

The deviation in approach in this study from the conventional open-source implementation lies in framing the problem within the framework of Optimal Foraging Theory (OFT). In the open-source software (OSS) implementation, food or items are randomly generated and fixed per episode, allowing agents to take actions as per their policy and action space. In contrast, this study adopts a model where food is regenerated at fixed intervals after consumption. This necessitates updating the OSS implementation to address concerns such as food collisions, potential spawn points, and the ability to re-spawn food regularly in these random positions. Consequently, a hyper-parameter, the re-spawn interval emerges enabling the modeling of environments with varying food densities leading to potentially intriguing interactions with agents. Note, this hyper-parameter leads to different problem formulations, and is not one that requires fine-tuning as it would yield to different interactions that could potentially answer interesting questions.

For example, in extensive state spaces with low food densities, it is expected that agents would prefer working independently. This could result in the development of characteristics such as "trauma" or greed since food is less abundant, and cooperation implies a team of agents sharing a sparse food source sparingly. Furthermore, in a large environment, the abundance of food may cause agents to cooperate, as not only the expected gains of the group would be larger than foraging alone, it would be supplemented by the fact that some foods require an effort of a cooperative group.

To streamline the training process within the constraints of time, the EPyMARL repository (9) is employed. This repository extends PyMARL (1). a widely used tool in Multi-Agent Reinforcement Learning (MARL) research. Given that the Learning-Based Foraging (LBF) scenario involves mixed

cooperative settings, disentangling rewards becomes a challenging task in MARL research. In this study, reward disentanglement is tackled by assigning a standardized reward, discounted by the energy penalty, to the agent in proximity to the food item.

C. Algorithms

IQL (Independant Q-Learning): as proposed by Tan in 1993 (2), operates on the principle of Independent Q-Learning. In this approach, every agent possesses an individualized state-action value function. This function relies solely on the local history of observations and actions specific to each agent. The process involves each agent receiving its localized history of observations. Subsequently, the agent updates the parameters of the Q-value network, following the principles outlined by Mnih et al. in 2015. (3). This form of training is IL or independent learning and does not fall in the CTDE (Centralized training, decentralized execution model). Much like Q-Learning in single agent RL, it is a value based learning algorithm that is off-policy in nature.

MAA2C (Multi-Agent A2C): Often referred to as Central-V, it computes a centralized V function. This algorithm falls in the CTDE category, and involves the use of a critic, that learns a joint state-value function where each agent adopts its own policy and acts as an actor helping refine the critics assessment of each states respective value. Since it extends the existing on policy A2C algorithm by applying centralized critics conditioned on the state of the environment instead of individual histories of observations, it is often referred to as Central-V.

VDN (Value Decomposition Network): Value decomposition is the process of decomposing a joint action state value function into individual state action value functions. VDN is a value based algorithm that takes advantage of the above rationale to find such a linear decomposition. Each agent has its own network to approximate its own state action values. VDN then decomposes this joint Q value into the sum of individual Q values, it is then trained with the standard DQN algorithm.

QMIX: an extension of VDN by Rashid et al (4) broadens applicability to diverse environments. It introduces a parameterized mixing network to compute joint Q-values based on individual state-action value functions. QMIX is trained to minimize the DQN loss.

D. Experimental Setup

I conducted more than 16 seperate experiments in total. Each algorithm was tested within different grid-world scenarios. To ensure a thorough evaluation, I ran each experiment for over two million timesteps on average for three to four hours each on a T4 GPU on Colab. The decision to take colab was primarily based on convenience. Alongside a modified grid world environment, I simulated a Sequential Social Dilemma (SSD) with spatial extensions. The experimental setup included variations in grid sizes (large, small) and food densities (large, less), introducing diversity. I then compared the performance of MARL algorithms in the SSD setting with their performance in a traditional Large Grid Foraging setting, as investigated in related works.

It must be noted, however, due to time constraints, several limitations may exist in this evaluation. For one, each experiment was not averaged over different seeds, as each experiment took upwards of 3 hours to train for a cumulative training time for over 35 hours. Although this isnt alot, the lack of a computing cluster and existing time constraints made it very significant in terms of computational costs. Furthermore, since competitive scenarios is still a relatively unexplored field in MARL research, this aspect was not a part of the problem formulation. The inclusion of these may or may not provide additional insights into the inter algorithmic behaviors of agents in MARL settings.

The algorithms were chosen in part by two driving factors, one due to the degree of differences that exist in their implementation and two with ones that are not covered by existing studies. For example, Central-V is often used as a baseline in MARL research, using another algorithm with a similar actor, critic network may not be as interesting as opting to use something like QMIX which attempts to create joint policies for agents. For the second factor, I reviewed over two dozen research-papers, which was slightly difficult since MARL is an emerging field of study and most of the works here are still relatively new and found that the experiements I have performed have not been done before, most related works that deal in SSDs attempt to propose novel algorithms or perform reward engineering.

Due to the re-spawning of food items, the metric used to evaluate or analyze behavior was average return per episode, evaluated once every hundred episodes. The training algorithms ran for 2 million time-steps, with each experiment or episode being constrained to 50. The number of agents used were two, with two state-spaces, one being 16x16 and the other being 8x8, they were tested in environments with high and low food densities respectively thus leading to a total of 16 experiments with over 30 hours cumulative of training time. In high food density regions, two items of level 1 and two items of level 2 were spawned in fixed intervals of 5 units, with 2 agents navigating the grid space. In low food density regions, only two food items of level 1 were spawned. Each agent was given a max possible player level capped by the maximum level of the food items spawned. To allow for more consistent evaluation of the underlying agent behavior. Several ideas like psuedo-induced scarcity in the form of famines (ie, half way through total timesteps, the high density becomes low density or vice-verca) and other settings like dynamic nature of food levels changing mid-episode or mid-training were avoided. This was done in part to build up on the experiments and gain a sense of baseline performance with some comparison to MARL benchmarks in the traditional LBF environment. Furthermore, this was also done in part due to time constraints, as each experimental setting would be subset to one of these environment and food density scenario which would result in exponential more training time, To grapple this fact, the

code was made to allow for such easy transformations of the environment and a migration to a computing cluster was somewhat under-way.

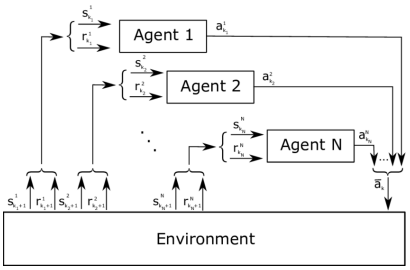


Fig. 2: MARL Architecture (5)

E. Methodology

The methodology for achieving the proposed objectives involves a systematic seven-step plan. Beginning with a thorough review of MARL literature, algorithm selection, and computational load testing, the initial steps aim to establish a baseline understanding of the nature of MARL algorithms given the fact I have never seen any of these things before. Subsequently, the experimental setup diverges into exploring socially sequential dilemmas by adjusting grid sizes, and food densities for several algorithms. This deviation provides valuable insights into the impact of trauma, scarcity, and socially sequential decision-making on MARL algorithms. As the study progresses, the environment is modified to handle mixed cooperative cooperative dynamics. Comparisons with existing MARL systems contribute additional benchmarks for a comprehensive evaluation. The final step involves meticulous data analysis, identifying strengths, weaknesses, and addressing the study's objectives. This methodology ensures a structured approach to investigating MARL algorithms in varying scenarios, laying the groundwork for informed insights and potential future studies. Deviations in this plan mostly revolved around time constraints, however the understanding of the OpenAI Gym API wrappers in conjunction with a thorough review of the LBF repository allowed for a fast recovery given some delays in hitting deadlines initially.

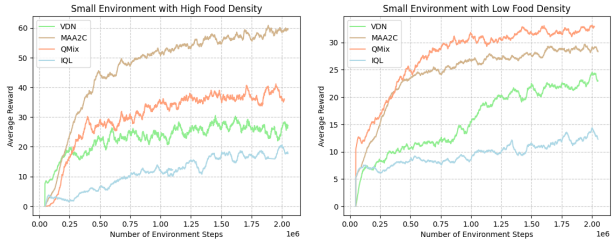
IV. Results

Timestep	Epsilon
50	1.0
26050	0.506
51050	0.05

TABLE I: Values of Timestep and Epsilon

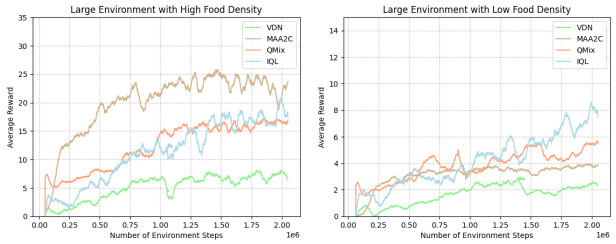
A. Independant Learning

Despite the simplicity of the IQL algorithm, it seems to perform reasonably well in all scenarios with the best being observed in a large environment with low food density, which is intuitively sound. In a large environment, where cooperation is not required, as food is scarce and perfectly attainable



(a) Small Environment, Large Food

(b) Small Environment, Small Food



(c) Large Environment, High Food

(d) Large Environment, Low Food

Fig. 3: Comparison of Different Environments with varying food densities

alone, the average return per episode is greatest in contrast to other algorithms. Although it still performs well on a smaller environment with higher food density, it's vastly outperformed by its CTDE counterparts. The rationale for this being that there is greater reward and incentives for cooperation, and agents get stuck in local minimums foraging for low energy foods. Based on the learning curves, I suspect that the agents begin with a higher exploration rate, which causes a fast rise in the average rewards earned per episode which exhibits a great deal of variance later down the line during the training phase. This could be because the agent struggles to develop an optimal policy to forage for food given it re-spawns in fixed intervals randomly. Not only this, but also because regions with higher food densities also require higher degree of cooperation or effort, it induces an almost comfort-like mechanism of being stuck in a local minimum. This could be remedied by an idea I had, what I call a "nudge" by engineering the exploration rate to increase a great deal when there is a decline in average returns earned. Intuitively, it could be like this: if you are struggling to pay the bills because of your job, start a company instead. This mechanism being in part mathematically inspired by the IPPO algorithm, which is a modification of the PPO (proximal policy optimization) with a surrogate objective function that caps the relative change in policy allowing for more update epochs using the same batch of trajectories.

B. CTDE Algorithms

MAA2C: As expected, MAA2C performed best in high food densities, since all agents share their observations and the critic was able to form a centralized policy that allowed agents to act in an optimal fashion cohesively. Since a smaller environment

allows for faster navigation, in the beginning phase of training, given a high enough epsilon, the centralized state action value function quickly learns to incentives cooperation as the agents remain in local neighborhoods of each other at all times (as seen through some renders on the model weights). Since the re-spawn timer is so frequent, while in the process of collecting foods together, agents also collect them in an independant fashion, whenever possible. However, MAA2C does not perform well in regions of low densities, this is because cooperation is not only required but detrimental. As the agents prefer to stick together, they get stuck in localized neighborhoods and often have to traverse long distances to forage for foods that may spawn far away from the position of their groups. Due to this, there is a reasonable increase in average rewards during the exploration phase and the exploitation phase gets stuck in a local minima. This makes sense as it is outperformed by its IL counterpart (IQL) in this scenario. It's interesting to note these trends also remain consistent in the small environment setting. If we were to somehow stumble across training data, it would be easy to realize the model or environment based on the interplay of these MARL-like algorithms. Since IL tends to perform more or less the same in small environments, we could deduce based on its comparison to its other algorithms, the nature of the world the agents found themselves in thereby highlighting the importance of state-agent dependencies.

VDN: In the context of small environments with low food densities, VDN exhibits a notable and abrupt increase in performance halfway through training. This observed spike suggests that, as the agents become more experienced in the environment, they progressively refine their coordination and decision-making strategies. The sudden improvement could indicate a phase where the agents successfully adapt to the dynamics of the sequential social dilemma, leading to more efficient collaboration and enhanced collective rewards.

Contrastingly, in larger environments, VDN faces substantial challenges, recording its lowest performance levels across both high and low food densities. This outcome indicates that the value-decomposition strategy encounters difficulties in effectively coordinating agents and capturing the intricate dynamics of larger, more complex spaces. The struggle in both food density scenarios emphasizes the scalability issues of VDN in expansive environments.

In small environments, VDN consistently outperforms its Independent Learning (IL) counterpart. However, when compared to more advanced MARL algorithms such as QMIX and MAA2C, VDN falls short in both high and low food density scenarios. This suggests that while VDN performs well in specific contexts, there are more sophisticated algorithms that surpass its capabilities in the considered sequential social dilemma settings.

QMIX: The sharp performance increases, notably in small environments with low food density, are intricately linked to QMIX's exploration strategy. The algorithm's efficient exploration, influenced by gradients and learning mechanisms, contributes significantly to rapid adaptation. This aligns with the anticipated benefits of QMIX's design, showing a capacity for swift learning and adjustment in environments where cooperative behaviors significantly impact overall performance.

QMIX's performance aligns with its design logic. The ability to learn joint Q decomposition methods is particularly effective in smaller environments where individual navigation is more straightforward. In these settings, the combination of individual foraging and collaborative efforts in high food density allows QMIX to outperform Independent Learning (IL). In certain test runs, it can be seen that there is a balance between individual foraging in certain neighborhoods whereby agents let go of their foraging to temporarily work together and return to their efforts. The coherence extends to small environments with higher food density, where cooperation becomes more crucial, explaining QMIX's superiority, especially over IL. This makes more sense when we reverse the argument, IQL struggles in these high density scenarios due to the "comfort-like" mechanism described. QMIX's behavior is like a ratio of comfort and expansion leading to higher than average return rewards.

However, in larger environments, where extensive coverage is required, and particularly in scarce food settings, cooperation becomes less practical. In such cases, QMIX trails IL. It surpasses IL in higher food densities as in these regions there are items that require cooperation, and IL is more concerned with individual foraging, but falls short in regions of lower densities, probably because of impractical cooperation that doesn't lead as much towards the average rewards. This alignment with expectations reinforces QMIX's versatility, especially in dynamically adjusting its strategy based on the environmental context.

V. Comparison to MARL-like Algorithms in Traditional LBF

Since LBF is an OSS environment that acts as a wrapper around Gym, the environments are indexed as such: "sxs-np-lf" where sxs is the state space, n is the number of agents and l is the number of food items. To keep consistency across the experimental setup, communication mechanisms were the same. The sight of each agent was limited to the same value of two and the decay in epsilon was also kept the same. This was done in part to study their relationships with each other and more importantly, the environment without giving one algorithm an advantage over the other (however it would be very interesting, as mentioned before, to model more complex communication mechanisms that can allow for more semantic capture of information or maybe even a more complex exploration function). Since traditional MARL-like algorithms have been run in a traditional LBF setting, it will be hard to compare their performance to the algorithms seen here. However, it would not be wrong to compare their performance with each other. For instance, I conducted a separate set of experiments with overall return value as an evaluation criteria in a large environment in traditional LBF with four food items and two players. MAA2C performed the best, VDN and IQL were almost tied for second place and QMIX came last.

However, in a modified setting, MAA2C comes first, IQL and QMIX are tied for second place and VDN slowly trails behind with average returns per episode as an evaluation criteria. Although we cannot compare their metrics, we can compare their ranks. In larger environments, VDN faces substantial challenges, recording its lowest performance levels across both high and low food densities. The value-decomposition strategy encounters difficulties in effectively coordinating agents and capturing the intricate dynamics of larger, more complex spaces. QMIX, on the other hand, exhibits sharp performance increases, especially in small environments with low food density. This is attributed to QMIX's efficient exploration strategy, influenced by gradients and learning mechanisms, facilitating rapid adaptation in environments where cooperative behaviors significantly impact overall performance. The scalability issues of VDN become evident in larger environments, where its value-decomposition approach faces challenges in coordinating agents effectively. In such complex settings, QMIX's exploration-focused strategy may lead to more adaptive and versatile behavior.

VI. Proposed Solutions

MAA2C demonstrates remarkable strengths in environments characterized by high food densities. Its ability to efficiently utilize a centralized policy for optimal agent cooperation shines in scenarios where collaboration is essential. The algorithm excels in the early phases of training in smaller environments quickly adapting to frequent re-spawn timers and encouraging cooperative foraging. However, its weaknesses become apparent in regions with low food densities, where cooperation is less crucial. In such situations, MAA2C tends to get stuck in localized neighborhoods, hindering efficient foraging and leading to challenges in traversing long distances for low-energy food. To address its decline in low food densities, proposed solutions include introducing more complex communication mechanisms and implementing mechanisms within the central policy to condition exploration probability based on environmental conditions. Furthermore, reward engineering could be more complex quadratic functions. However, as mentioned reward disentanglement in MARL is quite challenging and doing the former might require a strong sense or grasp over fundamental mathematics.

VDN exhibits notable strengths particularly in small environments with low food densities where it experiences a significant performance increase midway through training. This observed improvement suggests that as agents gain experience, they refine coordination and decision-making strategies, adapting to the dynamics of sequential social dilemmas. However, challenges arise in larger environments where VDN struggles to coordinate agents effectively leading to lower performance levels across both high and low food densities. The scalability issues become evident emphasizing the algorithm's limitations in expansive settings. Proposed solutions involve exploring strategies to enhance coordination in larger environments and considering adjustments to the value-decomposition approach for improved scalability.

QMIX stands out with sharp performance increases, especially in small environments with low food density. Its efficient exploration strategy, influenced by gradients and learning mechanisms, contributes significantly to rapid adaptation. QMIX's performance aligns with its design logic, as the joint Q decomposition method proves effective in smaller environments where individual navigation is more straightforward. The algorithm balances individual foraging and collaborative efforts in high food density, outperforming Independent Learning (IL). However, challenges emerge in larger environments where cooperation becomes less practical, causing QMIX to trail behind IL.

Proposed solutions include enhancing exploration strategies and investigating adjustments into the joint Q value function for better performance in larger environments.

IQL exhibits consistent performance in small environments, showcasing stability and providing valuable insights into the behavior of independent learning in sequential social dilemmas. However, its limitations become apparent in scenarios with heightened cooperation needs, where it may lack the adaptability seen in more advanced MARL algorithms. Serving as a stable reference point, IQL establishes a baseline for understanding the constraints of independent learning in sequential social dilemmas. Proposed solutions for IQL involve exploring mechanisms to enhance adaptability in various social dilemma scenarios and addressing challenges related to heightened cooperation requirements.

All of the above solutions require a certain kind of solution search that does not provide agents with knowledge about the model, this will produce very desirable short term results in gaining higher than expected average rewards or maybe even outperforming the state-of-the-art models in lieu of Moore's law. However, this will not contribute much to the idea of RL and the ability of agents to adapt to a wide array of scenarios. The dependancies of the environments configuration can yield a great degree of different results, and thus requires a compelete model-free approach.

VII. Limitations and Future Works

Given time constraints, some limitations have been noticed in the paper, with one of the prominent ones being the absence of the average of many trial runs across different seeds. However, since the main focus of the study was to develop a sense of inter-algorithmic performance in modified environments, the learning curves were smoothed with a moving window. In the future, randomizing over different seeds for five or more trials would provide more robust results. Also, since the study was more focused on inter-algorithmic performance across different settings, several parameters, communication topologies, and exploration strategies were forced to be kept the same to develop a consistent idea across the board.

MARL is a rapidly evolving field, and emerging areas such as HARL, involving heterogeneous agents, present intriguing opportunities for further exploration in problem formulations and algorithmic advancements. Integrating these concepts could enhance the study's depth and relevance. The inclusion

of an energy parameter could provide additional insights into algorithmic behaviors across diverse environments. While the discounting factor remained constant to establish baseline metrics, its potential variation, deferred due to prolonged training times and unexpected crashes, remains a prospect for future investigation. The experimental setup limitations stemmed, in part, from a learning curve with such problem formulations, a gap that has since been addressed.

Addressing competitive dynamics necessitates nuanced reward engineering, acknowledging the inherent challenge of the "moving target" problem in both single and Multi-Agent RL. The modified LBF design here aims to showcase mixed cooperative play, allocating rewards to adjacent players contributing to the acquisition of collectively unattainable items.

VIII. Conclusion

In conclusion, this paper presents an exploration of Multi-Agent Reinforcement Learning (MARL) algorithms in the context of socially sequential dilemmas (SSDs). Through a series of 16 experiments, the study unveils the intricate dynamics of agent navigation in modified environments, offering insights into the adaptability and performance of MARL algorithms. The comprehensive nature of the experiments, spanning varied grid sizes and food densities, contributes to a nuanced understanding of algorithmic behavior across diverse scenarios.

The computational rigor totalled over 35 hours of T4 GPU training, underscores the resource-intensive nature of delving into socially complex scenarios.

While the study provides valuable contributions, certain limitations are acknowledged. Time constraints prevented an exploration of average trial runs across different seeds. The study, focused on inter-algorithmic performance, necessitated the uniformity of parameters, communication topologies, and exploration strategies. Recognizing the evolving landscape of MARL, future works could extend into fields like Heterogeneous Agent Reinforcement Learning (HARL) for improved problem formulations and algorithmic enhancements.

The energy parameter's role in shaping algorithmic behavior in various environments and the exploration of different discounting factors remain avenues for future investigations. The challenge of competitive dynamics and reward engineering, compounded by the "moving target" problem inherent in Multi-Agent RL, poses a non-trivial task for further exploration.

References

[1] Mikayel Samvelyan, Tabish Rashid, Christian Schroeder de Witt, Gregory Farquhar, Nantas Nardelli, Tim GJ Rudner, Chia-Man Hung, Philip HS Torr, Jakob Foerster, and Shimon Whiteson. The StarCraft multi-agent challenge. International Conference on Autonomous Agents and Multi-Agent Systems, 2019.

[2] Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. International Conference on Machine Learning, 1993.

[3] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A. Rusu, Joel Veness, Marc G. Bellemare, Alex Graves, Martin Riedmiller, Andreas K. Fidjeland, Georg Ostrovski, et al. Human-level control through deep reinforcement learning. Nature, pages 529–533, 2015.

[4] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder De Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: monotonic value function factorisation for deep multi-agent reinforcement learning. International Conference on Machine Learning, 2018.

[5] Chincoli, Michele, Liotta, Antonio. (2018). Self-Learning Power Control in Wireless Sensor Networks. Sensors. 18. 375. 10.3390/s18020375.

[6] Charnov, E. and Orians, G. H. (2006). Optimal foraging: some theoretical explorations.

[7] Sih, A. and Christensen, B. (2001). Optimal diet theory: when does it work, and when and why does it fail? Animal behaviour, 61(2):379–390.

[8] Leibo, J. Z., Zambaldi, V., Lanctot, M., Marecki, J., and Graepel, T. (2017). Multi-agent reinforcement learning in sequential social dilemmas. arXiv preprint arXiv:1702.03037.

[9] M. Samvelyan, T. Rashid, C. Schroeder de Witt, G. Farquhar, N. Nardelli, T.G.J. Rudner, C.-M. Hung, P.H.S. Torr, J. Foerster, S. Whiteson. The StarCraft Multi-Agent Challenge, CoRR abs/1902.04043, 2019.

[10] Stefano V. Albrecht and Subramanian Ramamoorthy. A game-theoretic model and best-response learning method for ad hoc coordination in multiagent systems. International Conference on Autonomous Agents and Multi-Agent Systems, 2013.

[11] Littman, M. L. (1994). Markov games as a framework for multi-agent reinforcement learning. In Machine learning proceedings 1994, pages 157–163. Elsevier.

[12] Christianos, Filippos and Schafer, Lukas and Albrecht, Stefano V Shared Experience Actor-Critic for Multi-Agent Reinforcement Learning Advances in Neural Information Processing Systems 2020

[13] Gaurav Gupta (2020). Obedience-based Multi-Agent Cooperation for Sequential Social Dilemmas. UWSpace. http://hdl.handle.net/10012/15853

[14] Yuan, Y.; Guo, T.; Zhao, P.; Jiang, H. Adherence Improves Cooperation in Sequential Social Dilemmas. Appl. Sci. 2022, 12, 8004. https://doi.org/10.3390/app12168004

[15] Pinkesh Badjatiya, Mausoom Sarkar, Abhishek Sinha, Siddharth Singh, Nikaash Puri, Balaji Krishnamurthy. "Inducing Cooperation in Multi-Agent Games Through Status-Quo Loss," CoRR, vol. abs/2001.05458, 2020.

IX. Appendix

A. Configuration

Configuration	Value
Players	n_agents
Field Size	(n, n)
Food	$food$
discount	0.99
Sight	2
Max Episode Steps	50
Force Cooperation	False
Food Respawn Interval	5
Grid Observation	False
Food Respawn	True

TABLE II: Environment Configurations

B. MAA2C Hyperparameters

Parameter	Value
Action Selector	soft policies
Mask Before Softmax	True
Runner	parallel
Buffer Size	10
Batch Size Run	10
Batch Size	10
Target Update Interval or Tau	200
Agent Output Type	pi logits
Learner	actor critic learner
Learning Rate	0.0005

TABLE III: MAA2C Hyperparameters