Journal of Computational Science 9 (2015) 118–124

Contents lists available at ScienceDirect

Journal of Computational Science

journal homepage: www.elsevier.com/locate/jocs

Co-evolution in predator prey through reinforcement learning

Megan M. Olsen [a,*], Rachel Fraczkowski [b]

[a] Loyola University Maryland, Baltimore, MD, USA
[b] RDA Corporation, Baltimore, MD, USA

ARTICLE INFO

Article history:
Available online 18 April 2015

Keywords:
Reinforcement learning
Predator prey
Agent-based model
Q-learning

ABSTRACT

In general species such as mammals must learn from their environment to survive. Biologists theorize that species evolved over time by ancestors learning the best traits, which allowed them to propagate more than their less effective counterparts. In many instances learning occurs in a competitive environment, where a species evolves alongside its food source and/or its predator. We propose an agent-based model of predators and prey with co-evolution through linear value function Q-learning, to allow predators and prey to learn during their lifetime and pass that information to their offspring. Each agent learns the importance of world features via rewards they receive after each action. We are unaware of work that studies co-evolution of predator and prey through simulation such that each entity learns to survive within its world, and passes that information on to its progeny, without running multiple training runs. We show that this learning results in a more successful species for both predator and prey, and that variations on the reward function do not have a significant impact when both species are learning. However, in the case where only a single species is learning, the reward function may impact the results, although overall improvements to the system are still found. We believe that our approach will allow computational scientists to simulate these environments more accurately.

1. Introduction

Population dynamics study the development of either a single or multiple interacting species. In ecology, computational models are used to study the evolution within populations of plants and animals, such as which trees will survive in a forest over many hundreds of years, or what ratio of species is sustainable. A major topic of population dynamics is the cycling of predator and prey populations. Usually these systems are built to describe animal species, with at least one species as prey and one as predator, although more species can be included. The Lotka-Volterra [11] equations are based on the classic logistic equation, and commonly used to model this type of mutual interaction. However, it has been argued that these equations are not sufficient for truly modeling natural phenomena, as the expected fluctuations in species numbers are not sustained properly [10]. Also, the types of fluctuations found are likely only in simple situations, but cannot be maintained in complex interactions [5].

Agent-based models [13] of predator–prey dynamics are not new, but are the most recent approach to individual-based models [19]. Individual-based models have been used in population biology for many decades, and allow the inclusion of more detail than the traditional differential equations [6]. Using agent-based models, we can focus on interactions and behaviors, with fewer constraints on our modeling [2].

A common aspect to predator–prey interactions is their co-evolution through adaptation, learning, or both. However, we do not often model this aspect of population dynamics in simulation. When adaptation is included through machine learning, it is generally accomplished through Q-learning, a form of reinforcement learning [17]. Reinforcement learning is based on the psychological/biological learning approach of the same name, where an entity is rewarded for good behavior or punished for bad behavior. In computational reinforcement learning, an agent interacts with its world to learn the likely end value from a state change. The goal is usually to find the optimal state/action values for the agent following a specific policy, with the policy defining what action the agent should take based on the current state.

Standard Q-learning needs potentially hundreds of states for agents to learn. A state defines a single combination of aspects of the world that an agent may encounter, which may include locations of other agents, location of food, proximity to a certain location,

* Corresponding author.
E-mail addresses: mmolsen@loyola.edu (M.M. Olsen), fraczkowski@rdacorp.com (R. Fraczkowski).

http://dx.doi.org/10.1016/j.jocs.2015.04.011
1877-7503/

etc. In these instances, not only does the simulation run slowly, but learning requires many iterations to be successful; an agent does not learn in the same simulation in which it is studied, but instead must learn the value function first, and then have that function inserted into the final model.

We propose improving these problems by studying co-evolution through linear value function approximation Q-learning (LFA Q-learning). LFA Q-learning does not require a representation of states as defined by permutations of every agent and entity location within the agent's neighborhood, but instead focuses on the features of the environment that may affect a predator or prey's decision process. After an agent makes a decision within its world it is rewarded, and learns importance weights for each feature. The feature weights increase or decrease based on the state and reward, to learn which action is preferred. Since the number of features of the environment will always be smaller than listing every possible state (as each state would include most features), the agent has a smaller state space for learning. LFA Q-learning is known to converge faster than standard Q-learning, and can still converge to the optimal solution.

We show that LFA Q-learning can be utilized for prey and predator agents for real-time adaptation, and that both species learn successfully. We also show that the reward function affects learning success. We propose that LFA Q-learning is a more realistic approach to predator and prey learning as it allows learning in real time, allowing computational scientists to better study evolving competitive agent systems. In the next section we discuss agent-based models of predator prey systems and the role of learning in these simulations, then discuss our model and learning paradigm, followed by the results of our work and then conclusions.

2. Related work

Agent-based models provide a mechanism for observing complex systems in a controlled environment, such as predator–prey systems. In one case, hawk and dove agents are used in a predator–prey game theoretic model to study evolutionary biology [8]. Agent-based modeling is also used to study predator prey evolution, such as with fuzzy cognitive maps [7].

Most often when learning is applied to agent-based predator–prey models the goal is to study multi-agent cooperation and/or coordination; in this case, the problem is known as the "pursuit problem" [1]. In some studies of this problem variations on centralized control are used. Two agents may be joined together as joint-action learners, where they learn the state space around their combination of actions [4]. This combination may quickly become too time consuming to compute. Alternatively, a single agent may learn for the group, although this approach does not take advantage of all agents and would thus be less effective [3].

Independent learning agents are also studied in the pursuit problem. Genetic Programming, a technique for learning where various parameter sets are run and scored on how well they worked with new parameters evolved from the old ones, was used to co-evolve predators and prey. Although evolving a single species independently succeeded, co-evolution failed [9]. A modification to distributed Q-learning called hysteretic Q-learning was used to allow two predator agents to learn in a decentralized manner to chase a single prey [15]. In that case, coordination is limited to the predators, but is also forced by moving predators whenever they get too close to one another, so it is unclear if they can learn completely independently of the modeler [15].

We specifically study learning within the context of competition. In a true predator–prey system, we cannot assume that there is a centralized agent learning for all of its species, nor can we assume that predators and prey learn separately from each other. In these systems each entity is essentially an independent learner, although there may be sharing of knowledge within family units or packs [16]. For these more complicated situations, can both prey and predators learn the best possible strategy, and what is the best way to learn as one lives? Many studies have supported the idea of reinforcement learning in animal behavior [18], and that it may be the mechanism that helps them learn to find food and avoid enemies [14]. We propose that the best choice for these learners must be LFA Q-learning, because an animal most likely learns through experiencing a diverse set of situations, and detecting trends [16]. We are unaware of any prior work studying LFA Q-learning by independent learners in a predator–prey context.

We propose that LFA Q-learning provides a more realistic and computationally effective approach to modeling learning in predator prey dynamics. We show how LFA Q-learning can be incorporated into a predator-prey simulation, and that it results in agents learning fast enough to survive in their world without the need to iterate through many learning simulations as is traditionally done in reinforcement learning.

3. Methodology

We develop a stochastic agent-based model on a two-dimensional toroidal grid with three types of agents. Two represent the mobile population and focus of the model: predator and prey. The other non-mobile agent is the food source, and is considered an agent due to its features of reproduction and death. Multiple agents may inhabit the same location on the grid. We propose that predator and prey learn from their experiences, and investigate four research questions:

- How well can prey and/or predators use reinforcement learning to better survive in a basic model?
- What is the best reward function to encourage learning?
- How much better off is a species when it learns?
- If learning improves prey and/or predators' survival, does it only do so with co-evolution, or can each species evolve independently?

3.1. Non-mobile agent

In many models of predator and prey, food is ignored or placed as a constant non-interactive agent, since it is assumed that in stable environments food for prey lower on the food chain will likely be present. More recent models have added food agents that spread over time, and can disappear from over-grazing. Over-grazing and movement patterns based on food is a realistic issue, and therefore we include food as an agent that can be manipulated in the environment.

The food agent is structured similar to plant reproduction in reality, as it is unlikely that a plant will sprout up in a vegetation sparse environment. Thus, each food agent has a small chance of reproduction to create a new food agent in an adjacent location, as the only way for additional food to grow. Each food agent is technically a gradient of food supply, with values between 0 and 1. When a prey agent consumes food, it decreases the food agent's current level linearly. If the amount decreases to or below zero, it is considered to be an over-eaten area, and the food agent is removed. The model yields a gradual increase in the value over time for all living food agents, mimicking vegetation re-growth.

3.2. Mobile agents

The prey and predator agents in our model share many of the same attributes. A basic non-learning agent moves uniformly at

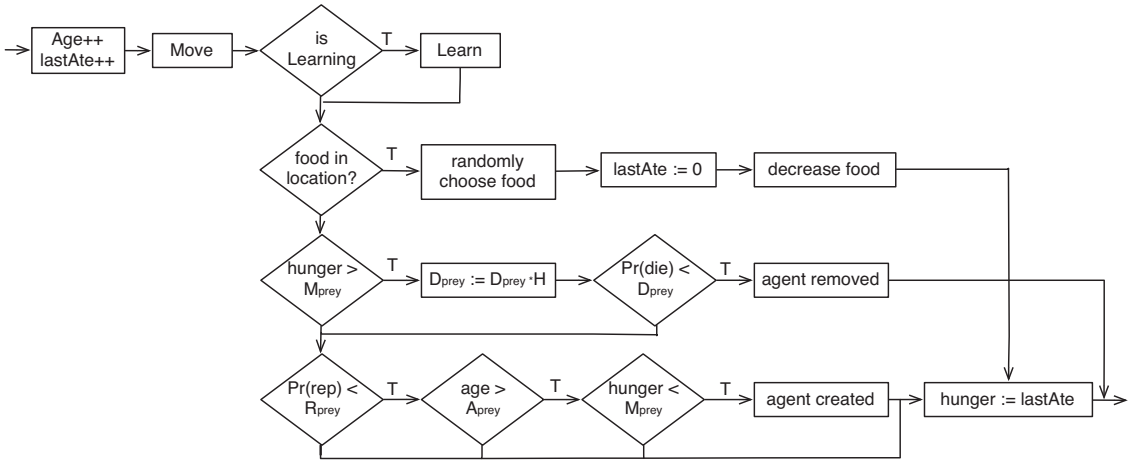

Fig. 1. Flowchart of a prey agent. Predator agent is identical except it eats prey instead of food agents, and all parameters correlate to predator. The details of the Move and Learn steps are described in Section 3.3.2.

random to a location in its Moore neighborhood, including its current location. There is no restriction on the number of agents that may inhabit a single location, so we do not force agents to move elsewhere when a grid location is occupied. Both predator and prey have a probability of reproducing at each time step R, creating a new agent in an adjacent location. Reproduction may only occur once the agent has reached reproduction age A, to limit agents from continuously reproducing without first interacting with the world.

To account for randomness and non-modeled events, each agent also has a small probability of death each step. This death probability can only be activated if an agent has a hunger level over the minimum level M. The death probability can also be increased the longer an agent has gone without food by a factor of H. A single step for a prey agent is demonstrated in Fig. 1.

Predator agents follow the same steps as prey, but using their own parameter values, and feed on prey instead of food agents. A predator feeds on a randomly chosen prey from those who are at the same location at the start of the predator's turn. Prey randomly choose a food agent at its location to eat.

When a new agent is created it inherits state from its parent, including the learned feature values. The feature values learned by agents only affect their movement choices, and thus movement for learning agents is discussed in the next subsection.

3.3. Reinforcement learning model

Predators and prey agents each use linear value function Q-learning. Linear value function Q-learning represents the world as a set of features instead of the classic set of states. Each feature is paired with a weight that defines its importance in the agent's decision making, and the agent learns the appropriate value for that weight over time based on rewards received for each action.

3.3.1. Features

Reinforcement learning is designed to learn the value of taking specific actions from a given state. In our model, as in most models, learning affects the agent's movement actions, as that is their primary action in the world. Learning more than one type of action at a time becomes too complex for most algorithms to be able to succeed in learning. A world state is represented by a set of 12 features:

f_0 ratio of all opponent agents in world that are in their neighborhood.

f_1 ratio of all similar agents in world that are in their neighborhood.

f_2 ratio of all food agents in world that are in their neighborhood.

f_3–f_{11} ratio of opponent agents in neighborhood that are in each of the nine neighborhood locations (includes current location).

These features were chosen as defining characteristics of the world. Although there are additional features that could be considered (location of similar agents or food agents in each neighborhood locations, for instance), 12 features is already a large number for LFA Q-learning to properly learn. We therefore refrain from adding additional features, which would hinder learning.

Each feature is initially assumed to have equal importance, and over time the agent will learn which are important or unimportant to determine their best action choice from a given situation. If a feature is deemed to be unimportant, its weight will move toward zero. Each feature is in the range [0, 1]. Due to the general difficulty in learning linear feature values without the use of a basis function, after calculating a feature it is modified via the radial basis function in Eq. (1) to normalize its value based on its similarity to the average case. This final feature value is the one used in learning, and is still in the range [0, 1]. The average case c_i is as follows: $c_0 = c_1 = c_2 = 9/(width * height)$; $c_i = 1/9$ for $i > 2$. In all cases, c_i represents an even distribution of the agents throughout the area being considered (number of neighborhoods in entire world, or in a single neighborhood).

$$f_i = \exp\left(-\frac{(f_i - c_i)^2}{2}\right) \tag{1}$$

3.3.2. Learning process

The quality of an action a from state s is calculated using a set of 12 weights (w_i), one corresponding to each feature (Eq. (2)). When an agent is deciding where to move, it calculates $Q(s, a)$ for every available location within their neighborhood, including its current location. We use the ϵ-greedy policy: the agent chooses the action from its current state with the highest calculated value with probability $1 - \epsilon$, and with probability ϵ it moves randomly to encourage exploration.

$$Q(s, a) = \sum_i w_i * f_i \tag{2}$$

Table 1
Table of non-learning simulation parameters: hunger minimum to allow random death (M), age of reproductive maturity (A), death modification due to hunger (H), initial agent count (food, prey, pred), death rate (D), and reproduction rate (R). Parameters that may differ between species denote the relevant species as a subscript.

Name	Value														
M_{prey}	12														
A_{prey}	12														
M_{pred}	15														
A_{pred}	15														
H	1												1.5		
Food	1500;2000								1500				2000	3000	4000
Prey	400			600					400				800	1600	1600
Pred	50			80					100	120			100	200	200
D_{prey}	0.001		0.01	0.001		0.01		0.01	0.01	0.01			0.01		
D_{pred}	0.001		0.01	0.001		0.01		0.01	0.01	0.01			0.01		
R_{prey}	0.1		0.1	0.1		0.1		0.2	0.1	0.1			0.1		
R_{pred}	0.05	0.01	0.05	0.05	0.01	0.05	0.1	0.2	0.05	0.05	0.1		0.05		

After movement, an agent is provided with a reward that approximates how good their new state is for them. We implement the following four reward functions:

$$reward_1 = opponent * type + 2 * same * type \quad (3)$$

$$reward_2 = opponent * type + 3 * same * type \quad (4)$$

$$reward_3 = opponent * type + 2 * same * type + food \quad (5)$$

$$reward_4 = opponent * type + 3 * same * type + food \quad (6)$$

where *opponent* is the number of the other species type within the agent's Moore neighborhood normalized by the number of that species in the world, *type* is 1 for predator and −1 for prey, *same* = {0, 1} for if the opponent is on the same location, and *food* is the number of food in the Moore neighborhood normalized by the amount of food in the world. The *type* variable allows the reward to be negative for a prey agent with surrounding predators, but positive for a predator agent with surrounding prey agents. We use the constants 2 and 3 to weight an opposing agent existing in an agent's exact location higher than in the neighborhood. The range of possible values for each reward function for predators are as follows: $reward_1 \in [0, 2]$, $reward_2 \in [0, 3]$, $reward_3 \in [0, 4]$, and $reward_4 \in [0, 5]$. For prey the ranges are $reward_1 \in [-2, 0]$, $reward_2 \in [-3, 0]$, $reward_3 \in [-4, 0]$, and $reward_4 \in [-5, 0]$. During each simulation only one reward method is active, and the same reward method is active for both agent types.

After receiving a reward, the agent learns how their choice of action a (movement direction) from state s (as defined by the features) performed when compared to their expected performance. The weights are adjusted based on the difference between what they earned and what they thought they would earn:

$$w_i = w_i + \alpha * (r + \gamma * \max_{a'} Q(s', a') - Q(s, a)) * f_i \quad (7)$$

where r is the reward given by the current reward function, γ is a discount factor for potential future values, $Q(s', a')$ is a calculated quality value of a possible future state by taking action a' from the new state s' (Eq. (2)), and $Q(s, a)$ is what the agent expected to accomplish by moving to the new location s' from state s through action a (Eq. (2)). The weight learning function will decrease the weight if it was too high, and increase if it was too low. The weight is a real number, and may be negative. Thus, the agent learns the magnitude of each feature's importance to its decision making.

4. Simulation

The simulation model is built using the MASON agent-based simulation framework for Java [12]. MASON includes the Mersenne Twister as its pseudorandom number generator, which is considered suitably random. At each time step agents are processed sequentially in random order following the process previously described, and then statistics are generated and output.

4.1. Simulation setup

Replications. The model is run with 15 random number generator seeds for each of the 22 parameter combinations in Table 1. These parameters are meant to act as a sensitivity analysis, testing various situations for prey and predator agents such as various food availability levels. If learning improves in all parameter sets, then it is more likely that learning improves in general in this type of system. Each parameter set is run for a maximum of 5000 steps in a 50-by-50 grid.

Learning parameters. LFA Q-learning uses an ϵ-greedy policy with three variables that affect learning: α, γ, and ϵ. The α dampens the level of learning; an $\alpha = 0$ means zero learning, whereas $\alpha = 1$ means the w_i is modified at the maximum level for its magnitude. Initial sensitivity analysis showed $\alpha = 0.05$ to be the highest α with good weight performance. In our simulations α begins at 0.05 and linearly decreases to 0 at timestep 500. The γ dampens the affect of new information versus old information in the w_i update. Initial tests showed that $\gamma = 1$ performed the best for this model, so all results use this value. The ϵ value determines the frequency at which a random movement is made instead of the maximum. We calculate $\epsilon = 1/t$ where t is the current time step starting from 1.

Evaluation statistics. We analyze all results in terms of the number of each agent type (*preyCount*, *predCount*), how often prey outrun predator (calculated as a decrease in neighboring predators after movement, *outran*), how often predator catches prey (predator eats, *caught*), how long the simulation runs before prey and predator decrease to zero (*ending*), how often prey eat (*preyEat*), and each species' hunger level (*preyHunger*, *predHunger*).

4.2. Validation

Stochastic agent-based models are notoriously difficult to validate, as the system generally lacks an oracle for determining if the answer is correct. After verifying that the code works as expected, we validate the non-learning model. We use animation to determine that the overall interactions and movement of the agents match what we would expect from a stochastic random movement agent-based predator–prey model. We validate this model via extreme condition tests by validating that extreme parameter values (high and low) affect the model as expected. We utilize sensitivity analysis through our simulation parameters, and do not find unexpected fluctuations due to parameter values. Validation of the learning model entails checking weights for the various features to determine if they are well-behaved. We performed these checks on multiple approaches to α, γ, and the policy for LFA Q-learning

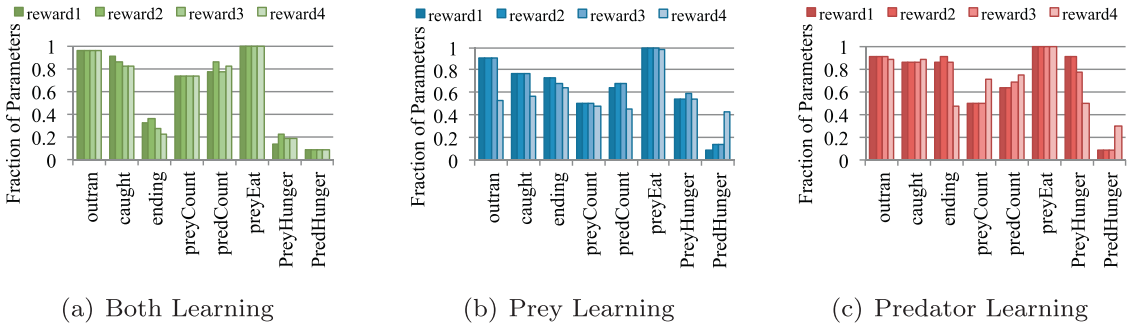

Fig. 2. Percentage of parameter sets in which the learning scenario performed better than the non-learning scenario in a given statistic, by reward function. Overall, both species learning and only predators learning provide the most frequent improvements.

before we found the approach presented in this paper, which is the most stable for this problem.

5. Results

We analyze the simulation results in terms of our four research questions: How well can prey and/or predators use reinforcement learning to better survive; What is the best reward function to encourage learning; How much better off is a species when it learns; If learning improves prey and/or predators' survival, does it only do so with co-evolution, or can each species evolve independently?

The last question will be addressed as each of the others are analyzed. We expect single species learning to be significantly more difficult than both species learning, as learning the correct movement when your opponent is moving randomly is much less likely to work than when your opponent is more strategic.

In addition to analyzing our research questions, we discuss the weights learned by prey and predators for each feature to determine their final policy.

5.1. Species survival improves across all parameters

How well can each species use learning to improve their survival? Across all parameter sets and reward functions, we see improvements on all statistics to some degree. When the two species are co-evolving by both learning, we see very few parameters sets that improve individual hunger levels (Fig. 2(a)). We see an improvement on prey eating in all parameter sets, and prey outrunning predators in 95.5% of parameter sets. The number of predators catching prey, prey counts, and predator counts are all improved in at least 72% of parameters. These statistics show that learning is successful in a variety of model scenarios. The ending point of the simulation is only improved in around 25% of parameters.

Single species learning improves the outcomes for both species. Similarly to [9], we see a benefit to each species when they are the only species learning, particularly in the case of only predators learning. Unlike in their work we also see an improvement when both species learns.

If only prey learn, all statistics except predator hunger improve for at least 50% of parameters for all but one reward function (Fig. 2(b)). Compared to the scenario of both species learning, the percentage is better for *ending* and *preyHunger*, and is still 100% for prey eating. It is logical that prey hunger and eating should improve when prey are learning, even if they do not learn to avoid the predator as well. All other statistics have a slight drop compared to the scenario of both species learning, as expected.

Both *ending* and *preyHunger* improve in the largest percentage of parameters when only predators learn (Fig. 2(c)). All other statistics are similar to the case of both species learning. There does not appear to be a significant detriment to the frequency of improvement over the non-learning case when only predators learn versus when both species learn. This result is surprising, as predator learning relies entirely on learning to find and capture prey, which are now moving randomly.

5.2. Reward function

What is the best reward function for each species? The reward functions differ primarily on the level of importance placed on the other agents in the neighborhood, and whether food is included. We can analyze each reward function based on the percentage of parameters sets that on average increase in each of our statistics from the non-learning case.

When both species are learning, there is very little difference among the reward functions. The best reward function is $reward_2$ as it gives a slightly better improvement across all parameters (63.6% versus 61.4% for $reward_1$ and 60.2% for the other two; Fig. 2). This reward function has the highest constant multiplier for nearby opponents, and does not include nearby food.

If only prey learn, there is slightly more fluctuation in the effect of reward functions. The most significant differences are seen in *outran*, *caught*, *predCount*, and *predHunger*. In all but the latter, $reward_4$ performs poorly whereas the other functions perform well. In the case of *predHunger*, $reward_4$ causes a 43% increase over the base case, whereas the other reward functions improve the statistic by 13% or less. As $reward_4$ has the highest constant multiplier for nearby opponents and includes nearby food, prey may be eating significantly more often than in other reward functions, but are sufficiently punished for moving toward predators even with food. This scenario matches the graph if the numbers are due to predator counts decreasing due to hunger. Comparing overall average improvement across all statistics, $reward_2$ and $reward_3$ have the highest at 65.9%, with $reward_1$ at 64.7% and $reward_4$ at 57.6%.

If only predators learn, the best reward function is again $reward_2$, with an overall improvement of 72.7% across all statistics and parameter sets. Whereas the first three reward functions have no meaningful differences, $reward_4$ is significantly different on four statistics: it improves *preyCount* and *predHunger*, but performs worst for *ending* and *preyHunger*. Rewarding predators for finding food when prey have not learned to stay near food may be detrimental to predators, as it alters their priorities inappropriately.

It is interesting that in all three learning scenarios $reward_2$ is the best performer. Whether prey and predators are competing with a learning species, they learn better when they are rewarded higher for having an opponent on their current location, but are not rewarded for being near food. The worst scenario to have a high value on the opponent, and value being near food $reward_4$, although the effect is extremely low when both species learn.

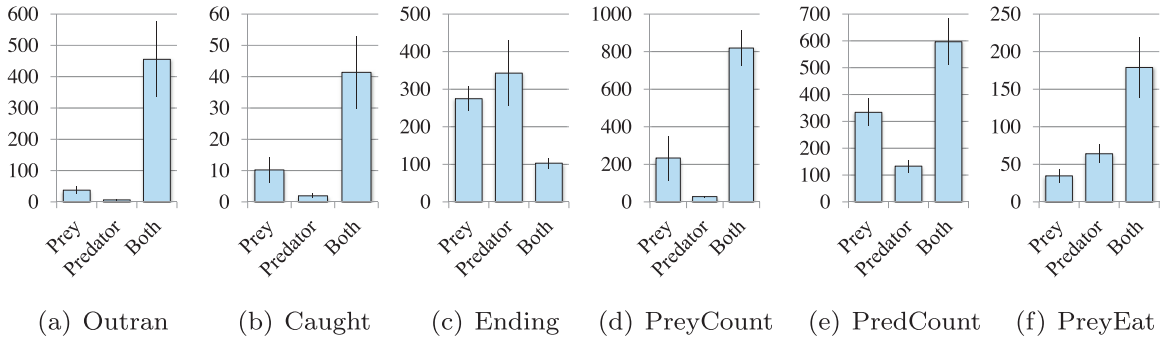

Fig. 3. The average increase for each statistic over the no learning base case for each learning scenario, with standard error. "Prey" refers to only prey learning, "Predator" refers to only predators learning, and "Both" refers to both species learning.

5.3. Learning improves each species

Although we can show that in general learning shows improvement for each species over non-learning scenarios, is the increase from learning significant? We compare the average values of six of the statistics for the best reward functions for each scenario ($reward_2$) to determine the level of benefit each species achieves through learning. The two hunger statistics are not shown as the number of improved parameters is too small.

As can be seen in Fig. 3, the most consistently significant improvement is when both species learn. This result supports the assumption that if only one species is learning it will dominate the other, which will lead to population crashes. The one statistic where prey only learning and predator only learning improve the situation better than when both species is learn is extending the life of the system (*ending*).

Surprisingly, both prey and predator counts increase more when only prey are learning than when only predators are learning. Prey counts are likely increasing due to their ability to survive longer for procreation. Predator counts are likely increasing due to predators eating less and thus not exhausting their food supply as quickly.

5.4. Learned weights

Both prey and predators learn until time step 500, with a decreasing α over time. Since each learning agent passes all weights to its offspring, every family is learning continuously from the start to end of the learning time steps. Instead of multiple iterations of the entire simulation, we can think of each family as an approximation of multiple iterations running concurrently. Of course, we have no guarantee of optimality in real-time learning without the usual replications, although the situation better approximates true co-evolution. Generally in reinforcement learning problems we want to be able to show that weights converge to optimality. Due to the nature of a stochastic agent-based model, we do not know the optimal solution.

The most interesting weights are those learned by prey in the case of both species learning. The features for prey and predators in the neighborhood have weights of 33 on average, with a standard deviation of 1.5. The weights for f_2, the count of food in the area, is universally 0 for each species in all learning scenarios. The weights learned for the individual Moore neighborhood locations $f_3 - f_1 1$ are all within $[-1, 1]$ on average, with standard deviations in that same range.

We find that when both species are learning, predators generally learn extremely large weights (3.3×10^{118}), but learn smaller weights when they learn on their own (6.4×10^{67}). When only a single species is learning, prey learn large weights universally (1.3×10^{140}). Most likely these large weights are caused by the random movement of predators interfering with the prey determining the actual value of an action. Surprisingly, despite this difficulty the results do still improve through single species learning, as already discussed.

6. Conclusions

In this paper we present an approach to co-evolution through learning in a predator–prey agent-based model. Predator prey models are similar to the competitive multi-agent system pursuit problem, which is difficult to solve in a distributed way. Unlike in the pursuit problem, we have a large number of agents, and the simulation does not end when a prey agent is caught by a predator. We propose using linear value function approximation Q-learning as an improvement over standard Q-learning for this type of system, so agents learn the importance of features of their world. In this variation of Q-learning the agents learn independently during their lifetime, pass their knowledge on to their children, and do not require a training period. This approach is more similar to how biologists theorize that animals learn and evolve. We are unaware of other work using this approach.

We test our learning approach on 22 simulation parameter sets to determine how often learning improves upon the non-learning base case, and to what extent it improves. We find that both predator and prey learn to improve their prospects in the majority of scenarios tested. We find that when both species are learning the improvement is more frequent and stronger than when one species learns alone. The exception to this rule is when studying the length of time the simulation runs before both populations die out, which improves best when only one species learns. Although prior work found individual species learning to be more successful than both species learning [9], we found a benefit in both single species and dual species learning.

The system is not sensitive to reward functions that return smaller values. However, if the reward function has a larger range of values the improvement deteriorates. In general, prey and predators should not be rewarded for being near food, but only for agents from the opposing species who are in their neighborhood. The system is most sensitive to reward function choice when only a single species is learning.

Our work demonstrates that co-learning can succeed in improving both competing species, and that LFA Q-learning can enable agents to learn in a real-time competitive predator–prey environment. These results can influence future approaches to simulation population dynamics and studying evolutionary learning. In the future we will study other reinforcement learning techniques to compare the success of each approach for a predator–prey system, as well as their time efficiency. We will also extend our

sensitivity analysis to other model parameter sets, in particular to study the impact on learning of hunger influencing death rate.

Acknowledgments

The authors thank Loyola University Maryland and the Clare Boothe Luce Program for their support of this research. They also thank George Konidaris for sharing his insights on linear value function approximation in Q-learning. All views expressed in this paper are views of the authors only. The authors also thank the reviewers for the helpful feedback.

References

[1] M. Benda, V. Jagannathan, R. Dodhiawalla, Technical Report. BCS-G2010-28, Boeing AI Center, Boeing Computer Services, 1985, August.
[2] F. Bousquet, C. Le, Multi-agent simulations and ecosystem management: a review, Ecol. Model. 176 (3) (2004) 313–332.
[3] C. Boutilier, Planning, Learning and Coordination in Multiagent Decision Processes, 1996, pp. 195–201.
[4] C. Claus, C. Boutilier, The Dynamics of Reinforcement Learning in Cooperative Multiagent Systems, 1998, pp. 746–752.
[5] J.L. Cloudsley-Thompson, Animal Conflict and Adaptation, Dufour Editions, 1965.
[6] D.L. DeAngelis, W.M. Mooij, Individual-based modeling of ecological and evolutionary processes., Annu. Rev. Ecol. Evol. Syst. 36 (2005) 147–168.
[7] R. Gras, D. Devaurs, A. Wozniak, A. Aspinall, An individual-based evolving predator–prey ecosystem simulation using a fuzzy cognitive map as the behavior model, Artif. Life (2009).
[8] H. Mirsad, C. Ted, C. Charles, Complex adaptive systems and game theory: an unlikely union, Complexity 1 (1) (2010).
[9] T. Haynes, S. Sen, Evolving Behavioral Strategies in Predators and Prey, Springer Verlag, 1996, pp. 113–126.
[10] C.L. Lehman, D. Tilman, Competition in Spatial Habitats, Princeton University Press, New Jersey, 1997, pp. 185–203.
[11] A.J. Lotka, Elements of Physical Biology, Williams and Wilkins, 1925.
[12] S. Luke, C. Cioffi-Revilla, L. Panait, K. Sullivan, G. Balan, Mason: a multi-agent simulation environment, Simul. Trans. Soc. Model. Simul. Int. 87 (7) (2005) 517–527.
[13] C.M. Macal, M.J. North, Tutorial on agent-based modeling and simulation, J. Simul. 4 (2010) 151–162.
[14] N.J. Mackintosh (Ed.), Animal Learning and Cognition. Handbook of Perception and Cognition, 2nd ed., Academic Press, 1994.
[15] L. Matignon, G.J. Laurent, N. Le Fort-Piat, Hysteretic Q-learning: An Algorithm for Decentralized Reinforcement Learning in Cooperative Multi-Agent Teams, 2007, October, pp. 64–69.
[16] J.E.R. Staddon, Adaptive Behavior and Learning, Cambridge University Press, 1983.
[17] R.S. Sutton, A.G. Barto, Reinforcement Learning: An Introduction, MIT Press, 1998.
[18] E.L. Thorndike, Animal Intelligence, Macmillan, 1911.
[19] D. Tilman, C.L. Lehman, P. Kareiva, Population Dynamics in Spatial Habitats, Princeton University Press, New Jersey, 1997, pp. 3–20.