Anticipatory Learning for Focusing Search in Reinforcement Learning Agents

George Konidaris and Gillian Hayes

Institute of Perception, Action and Behaviour
School of Informatics, University of Edinburgh
James Clerk Maxwell Building, King's Buildings
Mayfield Road, Edinburgh EH9 3JZ
Scotland, UK

Abstract. We introduce the use of an anticipatory learning element that ranks novel state-action pairs in reinforcement learning agents, replacing the use of uniformly optimistic initial values for exploration. We argue that reinforcement learning functions as a search process for situated agents, and that anticipatory learning can be viewed in this context as the process of learning a heuristic that guides search behavior. The discussion is grounded with some gridworld experiments that demonstrate that a simple learning element can learn to rank actions sufficiently accurately to rapidly improve search behavior.

1 Introduction

Although the concept of anticipatory learning is intuitively appealing, its role and the potential behavioral benefits it brings to situated agents are not yet well understood [2]. In this paper, we consider one particular way in which an anticipatory learning element can provide useful behavioral benefits – that of learning to rank novel state-action pairs in reinforcement learning agents.

Reinforcement learning provides an intuitively appealing agent learning model because it is well suited to the problems that situated agents face [10], and because it provides a principled way to build agents motivated by a set of internal drives. Unfortunately, the use of such drives inevitably results in situations where reward is delayed because it results from a long sequence of unrewarded actions. In such cases, the agent often spends a significant amount of time initially searching for any solution at all, since the impoverished nature of the reward signal is not sufficient to quickly guide it to a solution.

In this paper, we show that a reinforcement learning agent with the additional capacity of being able to learn to distinguish good actions from bad, and thereby desirable novel states from undesirable ones, acquires a behavioural advantage that can be characterised as analogous to guided rather than blind search, with the additional advantage of being able to learn the heuristic online.

Previously [5], we have shown that the use of associative learning to model such a heuristic can improve the performance of a realistically simulated robot that uses reinforcement learning to learn to find a puck in a maze, and return home. The approach presented here differs in that state-action pair rankings are learnt explicitly (rather than implicitly, using a state similarity metric), and in that we explicitly evaluate such behavior as anticipatory. In addition, the computational experiments presented here utilise a simple, abstract model of a situated agent, and thus allow for a more thorough evaluation of the model and its behavioral characteristics than would be possible with a real or realistically simulated robot.

2 Background and Preliminaries

In this section we briefly discuss what we mean by anticipatory learning, outline the particular model of situated reinforcement learning that we assume, and introduce the gridworld environment that we employ as an experimental domain throughout the remainder of this paper.

2.1 Anticipatory Learning

Anticipatory behavior is a broad term – virtually any form of learning can be considered anticipatory because it acquires knowledge about the world and how to act in it. In this paper, however, we are concerned specifically with learning mechanisms that, given an agent's past experiences, predict something useful about situations that it has never encountered before. Such mechanisms could be classed as *payoff anticipatory* or *state anticipatory* mechanisms, depending on what it is that they predict [3]. The mechanism introduced here falls into the payoff anticipation category, since it ranks novel state-action pairs with the aim of maximising future total payoff. Note that although reinforcement learning mechanisms by themselves are often considered anticipatory [3], such anticipation is usually either based on an agent's direct prior experience of each state, or an explicit model given in advance. For the purpose of this paper we do not consider either anticipatory, as the first deals with states the agent has experienced before, and the second deals with states that it has been given explicit prior knowledge of.

Following Butz, Sigaud and Gérard [3], we are interested in establishing how such mechanisms could be used to generate useful behavior, and under what circumstances they are feasible. The model we introduce here is one example of an anticipatory mechanism that produces useful behavior, in the form of guided search, and the remainder of this paper is concerned with demonstrating the advantages it provides and discussing the conditions under which it is able to provide them.

2.2 Situated Reinforcement Learning

Reinforcement learning refers to a family of methods that aim to solve the problem of learning to maximise a numerical reward signal over time in a given environment (Sutton and Barto [11] give a thorough introduction to the field).

In control tasks, reinforcement learning is usually used to learn the values of *state-action pairs*, so that the agent can select the action that should be taken at each state in order to maximise total reward over time. It is attractive because it provides a principled way to build agents whose actions are guided by a set of internal drives, has a sound theoretical basis, can allow for the principled integration of a priori knowledge, handles stochastic environments and rewards that take multiple steps to obtain, and is intuitively appealing.

In this paper, we assume that we are dealing with a situated agent using the reinforcement learning model described in Konidaris and Hayes [4]. This model differs from standard temporal difference reinforcement learning models in three ways. First, reinforcement learning is layered on top of a topological map, so that it takes place at the level of the problem space (in this case, at the level of navigation), rather than at the level of the robot's sensors. Second, learning is distributed, so that updates may occur in parallel if the agent has parallel hardware. Finally, learning is asynchronous, in that each of the states performs updates constantly, whatever state the agent happens to be in, and whatever else it is doing. The first difference means that learning occurs in a small state space, and that there is a separate learning element that is able to learn to distinguish between aliased states and form an internal map of the resulting state space. The last two differences mean that thousands of updates may occur while the agent completes an action, so that we can assume (as has been shown to occur on a real [4] and a simulated [5] robot) that the learning process *converges between decisions*, so that the agent makes the best possible decisions given its experiences.

Exploration in reinforcement learning is usually performed using ϵ-greedy methods, where the agent makes a random move some fraction (ϵ) of the time, which guarantee asymptotic coverage of the state space. In addition, the agents typically assign optimistic initial values to new states to encourage exploration, so that agent is more likely to explore unknown regions in the state space because they have higher values [11]. These values are gradually replaced as the agent learns the actual value of each state. However, we claim [4] that ϵ-greedy methods (along with all other methods that emphasise asymptotic coverage) are not appropriate for situated agents that must act as best they can given the knowledge, time, and resources they have available. Rather than occasionally making random decisions, an agent should learn how to satisfy its drives as soon as it can, and explore when it has time. We therefore separate exploration in situated reinforcement learning into two separate concerns: first, how to find any solution path at all, and second, how to thoroughly explore the state space in order to improve that solution or find new ones. In this paper we use anticipatory learning to improve the first, since mechanisms already exist to handle the second (e.g., using an exploration drive [4], or modelling curiosity [9]).

2.3 The Gridworld

We use a simple gridworld domain throughout this paper to illustrate our arguments and compare standard reinforcement learning to a model with an anticipatory learning element. A randomly generated example gridworld is shown in Figure 1.

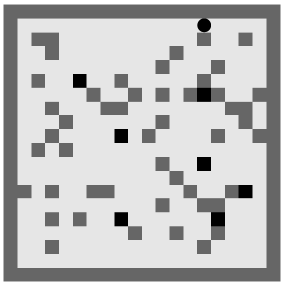

Fig. 1. An example gridworld

The agent starts in the lower left corner, and at each open square is able to take one of four actions (move up, down, left or right), unless that would result in it hitting one of the walls (dark gray squares, and here scattered over 15% of the grid and forming its outer perimeter). The agent's task is to reach the puck, (the black circle), whereupon it receives a reward of 1000. It should also avoid falling into a hole (black squares, here scattered over 2% of the grid), which would cause it to lose 10 points. Every other action costs the agent a single point. We assume that the agent has sensors sufficient to be able to determine when it cannot execute an action due to the presence of a wall, and because we assume the agent has a topological map, that it can unambigously determine its state and store data corresponding to it. In this very simple domain, this means that we can use a lookup table with an element for each grid square as the state space. Reinforcement learning is accomplished using an asynchronous temporal difference equation [4] similar to TD(0) [11].

In addition, we assume that the agent has a hole sensor for each direction which can sense when there is a hole directly adjacent to the agent in a given direction, and four "smell" sensors that provide an intensity reading (inversely proportional the distance to the puck) for each direction. Note that although the agent possesses these sensors, it does not a priori know how to use them.

Although this is obviously a highly simplified environment, and agents in it can hardly be considered situated, it serves as a clean and clear experimental domain where the points we discuss here can be easily illustrated.

3 Reinforcement Learning as Systematic Search

We can view the process of trying to find an initial solution as a search process where the agent itself is physically, rather than mentally, searching through the state space. This is similar to the classical idea of searching as problem solving

[8], except that since the search is actually being carried out (rather than being simulated in memory) only one node in the search tree can be considered "open" at a time. Korf [6] considered this problem within the framework of the classical deterministic search problem where an exact model of the state space is available, and Barto, Bradtke and Singh [1] showed that dynamic programming is a generalisation of this idea for stochastic problems. Although both papers included work on learning heuristics for the search space, these were simply more accurate values for states that had already been encountered, and neither paper considered the use of other sensors to predict the value of each state.

When searching a new state space, the simplest useful solution to the search problem is to have the agent select moves at random. This obviously does not perform very well because the agent spends much of its time in territory that it has already covered, until it finally stumbles across the puck.

An immediate improvement is to make the agent *explore greedily with respect to novelty*, where it chooses an action that has not been taken from its current state if one is available, and moves randomly otherwise. Although we would expect this agent to perform better than a purely random agent (because we would expect it to explore more efficiently) it nevertheless will often waste time randomly wandering through fully explored regions of the space until it chances across a state which has an unexplored action, because it is a greedy explorer. Note that this strategy requires a structure capable of recording which actions have been taken at previously visited states – a requirement easily fulfilled if a topological map is present but which could create difficulties in systems using function approximation techniques to compress the state space.

Reinforcement learning using (uniformly) optimistic initial values [11] for exploration is an improvement over this search strategy, because it propagates the optimistic exploration values of unvisited states through the state space, and thus causes an agent that has not found a solution to take the shortest route to an unexplored action from its current state. It can thus be considered a *systematic exploration strategy* because it thoroughly explores the state space, and does not waste time wandering around already explored regions of it. In addition, in more general situations where movement penalties are not uniform or where negative reward states are present, it is able to find paths to unexplored states without incurring unnecessary penalties. This is illustrated by the sample gridworld and coverage maps shown in Figure 2. Figure 2a shows a randomly generated sample gridworld, with Figures 2b, 2c and 2d showing the relative density (darker squares having been visited more) of visits over 100 runs of random, greedy exploration, and reinforcement learning agents, respectively, with each run starting in the lower left corner with no prior knowledge of the gridworld, and ending when each agent first finds the puck. The reinforcement learning agents clearly explore systematically and more thoroughly than the other agents, and in addition they are able to learn to avoid holes, as shown by the lighter colouring of the hole squares. The other searches cover the entire search space roughly uniformly, although they both appear to get stuck in the top left corner and very lightly cover the bottom right corner, probably due to the trap-like configuration of the

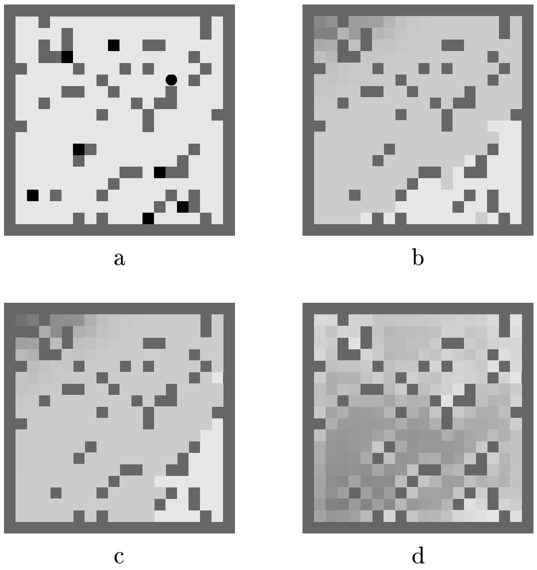

Fig. 2. A sample gridworld (a) and coverage maps for it using the random (b), greedy (c) and reinforcement learning (d) exploration strategies.

walls in each. This systematicity means that the agents using it find the goal in an average of 586 moves, whereas the greedy-exploration and random agents require an average of $13,595$ and $87,916$ moves, respectively.

The reinforcement learning agents still explore (on average) in a breadth-first manner – they cannot do any better because they have no a priori way to evaluate potential actions, and thus must consider all potential actions equal. Such search strategies can be considered *unguided* [8] because they do not use any form of heuristic to inform the search process. Without such a heuristic, we can expect to do no better – but if one is present, we can use it to perform a guided search of the state space.

4 Anticipatory Learning: Providing a Heuristic

When reward is necessarily delayed, we can expect reinforcement learning by itself to do no better than systematic search, because it simply does not have the information available to do any better. Taking the search analogy further, if we would like our agents to exhibit behavior similar to A* search [8], then we must provide them with a heuristic that can be used to rank novel state-action pairs in order of interest, and thereby guide search.

Unfortunately, building such a heuristic directly into the agent requires a high level of a priori knowledge of the task, entails significant design effort, and results in agents with fixed, rather than adaptive, search patterns. Using the agent's sensors directly as the state space is also difficult because it removes the advantages of using a topological map – the Markov property will be lost, it will be difficult to link high level actions to states, and most of the agent's sensors will not be relevant to the task, resulting in an infeasibly large state space. The natural solution to this is to employ a learning element that attempts to rank all of the possible actions at a newly discovered state. Such a solution would ideally be sufficiently rapid to learn quickly, but sufficiently flexible to be able to adjust to new environments. To implement this, we connected four output sigmoid units [7] to the agent's four hole sensors and four smell sensors. Unless the smell sensors were all zero or all equal, they were pre-processed so that:

$$s_i = \frac{s_i - s_{min}}{s_{max} - s_{min}} \tag{1}$$

where s_i was the ith sensor, s_{max} the one with the highest reading and s_{min} the one with the lowest. The sensors were therefore scaled relative to each other, with the one with the highest reading returning 1, and the one with a lowest reading returning 0. The output units were then used to rank potential novel actions, with each untried action given an action value of:

$$v_i = kn_{novel} \frac{o_i}{o_{total}} \tag{2}$$

where k was the default optimistic initial value (set to 10), o_i was the ith sigmoid unit's output, n_{novel} was the total number of untried actions at the current state, and o_{total} the sum of their sigmoid outputs, resulting in values scaled against the total output for all novel actions. The action with the highest action value was taken, although this was not always a novel action since a previously taken action from the same state may lead to a higher reward.

In order to evaluate the benefits of this element, we compared the average total reward obtained by 100 agents using the standard reinforcement learning search method in the gridworld shown in Figure 2a with that obtained by 100 agents with the extra learning element as they were exposed to more and more training examples. Each agent with the anticipatory learning element was given 70 trials, consisting of a training stage and an evaluation stage.

Each training stage took place on a randomly generated gridworld, using the following procedure. First, the agent was left to find the puck using reinforcement learning and using its existing sigmoid units to estimate optimistic values. The ranking element weights were frozen during thise phase so that no learning took place. If the puck was found in fewer than 1000 steps, the agent made the remainder randomly, without learning, in order to further explore the gridworld. Finally, the agent was replaced in its initial position, and allowed to follow the path it had found to the puck. For each action that it took, the sigmoid units were updated towards a value of 0.95 for the chosen action, and 0.05 for the remainder, using gradient descent [7]. Following each training trial, the agents were evaluated in the gridworld shown in Figure 2a, with their ranking element weights frozen. The results obtained thus indicated performance in an environment in which the agents had never been allowed to learn in.

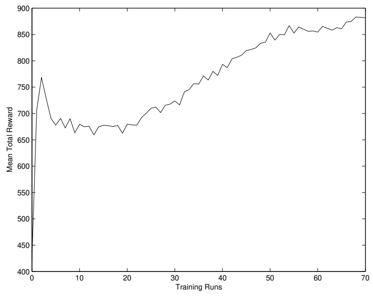

Fig. 3. Mean Reward over Time

Figure 3 shows the resulting improvements in reward over time, where it is clear that the anticipatory learning element is able to improve the agent's behavior and allow it to obtain a higher mean total reward as time progresses. The agent learns a relatively effective strategy fairly quickly, but is also able to improve steadily over time. The mean reward for the standard reinforcement learning agent over the same task was slightly less than 400, which matches with the level at which the learning agents start out. Figure 4 shows the drop in mean ranking error over time for the learning element, showing consistent but diminishing improvement.

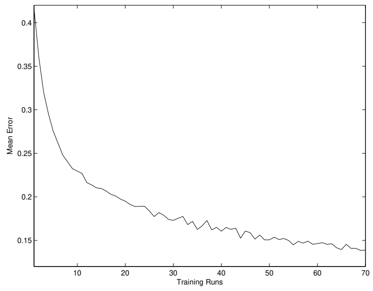

Fig. 4. Mean Ranking Error over Time

Finally, Figure 5 shows the resulting average search pattern after 0, 5 and 25 training experiences, which is a good indicator of the agent's expected search behavior. Initially, the agent spends most of its time in a breadth-first outward expansion, similar to the coverage map generated by an agent using reinforcement learning with uniformly optimistic initial values, as shown in Figure 2d. This is to be expected, since the untrained ranking network can be expected to rank novel actions randomly, which is equivalent to ranking them uniformly and then breaking ties randomly. Later, the agents learn to move towards the

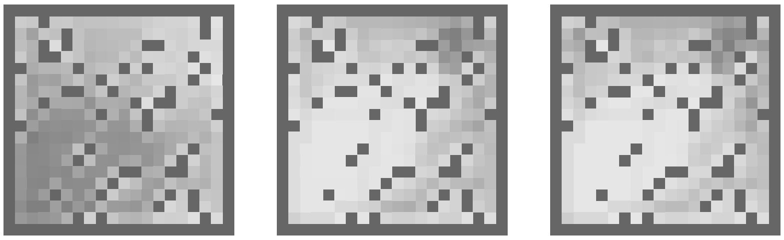

Fig. 5. Frequency Maps after 0, 5, and 25 Training Runs

smell while avoiding obstacles, and spend most of their time in the area where their smell sensors have very high readings (in the upper right corner near to the puck). It is clear from these patterns that the learning element is successfully able to focus the agents' searching behavior, even though it is never allowed to learn in this particular gridworld.

The anticipatory learning element is thus able to rapidly learn to significantly decrease the time taken by a reinforcement learning agent to find a solution, and to avoid holes. The resulting search behavior can be characterised as a guided search, with the anticipatory learning mechanism providing a form of heuristic.

5 Discussion

The mechanism demonstrated in this paper for anticipatory learning was very simple, but it supported relatively interesting, goal-directed behavior and provided some obvious behavioral benefits. Despite its simplicity, we feel that the work presented here highlights two interesting aspects of anticipatory learning systems. First, the system demonstrated here works because of the existence of a behavioral regime that provides a context in which anticipatory learning is useful. The ability to forsee the consequences of action is not useful in and of itself – rather, it is useful because it provides a means by which good actions may be chosen over bad. In this paper, the form of the behavioral strategy and learning mechanism used both afforded the opportunity for anticipatory learning and provided a framework in which it could provide behavioral benefits.

Second, although the sensors and learning element used in this paper were deliberately kept simple in order to demonstrate clear results, we would expect a kind of synergy between sensing and learning: that sensors and learning systems would be co-evolved to make learning and the generation of adaptive behavior flexible and computationally easy. In this case, anticipatory learning does not have to be exact – that would probably require a much more complex learning element – it just needs to rank novel actions in roughly the right order. Explicitly learning a predicted value for each state is likely to require significantly more computational effort, and may be difficult to accomplish within a reasonable portion of the task lifetime.

6 Conclusion

In this paper, we have introduced the use of an anticipatory learning element for novel state-action pair ranking in reinforcement learning agents. We have shown that anticipatory learning is effective as a heuristic that is capable of focusing search for such agents, and thus provided a concrete example of the behavioural benefits of adding anticipatory learning to an existing control system.

Acknowledgements

We would like to thank the Institute of Perception, Action and Behaviour at the University of Edinburgh for the use of its resources, the three reviewers for their useful comments, and the organisers of ABiALS 2004 for some good ideas. George Konidaris is employed under the HYDRA project (EU grant IST 2001 33060).

References

1. A.G. Barto, S.J. Bradtke, and S.P. Singh. Learning to act using real-time dynamic programming. *Artificial Intelligence*, 72, 1995.
2. M. V. Butz, O. Sigaud, and P. Gérard. Anticipatory behavior: Exploiting knowledge about the future to improve current behavior. In M. V. Butz, O. Sigaud, and P. Gérard, editors, *LNCS 2684 : Anticipatory Behavior in Adaptive Learning Systems*. Springer-Verlag, 2003.
3. M. V. Butz, O. Sigaud, and P. Gérard. Internal models and anticipations in adaptive learning systems. In M. V. Butz, O. Sigaud, and P. Gérard, editors, *LNCS 2684 : Anticipatory Behavior in Adaptive Learning Systems*. Springer-Verlag, 2003.
4. G.D. Konidaris and G.M. Hayes. An Architecture for Behavior-Based Reinforcement Learning. To appear, *Adaptive Behavior*, 2004.
5. G.D. Konidaris and G.M. Hayes. Estimating future reward in reinforcement learning animats using associative learning. In *From Animals to Animats 8: Proceedings of the 8th International Conference on the Simulation of Adaptive Behavior*, July 2004.
6. R.E. Korf. Real-time heuristic search. *Artificial Intelligence*, 42, 1990.
7. TM Mitchell. *Machine Learning*. McGraw-Hill, 1997.
8. S.J. Russell and P. Norvig. *Artificial Intelligence: A Modern Approach*. Prentice Hall, Eaglewood Cliffs, New Jersey, 1995.
9. J. Schmidhuber. A possibility for implementing curiosity and boredom in model-building neural controllers. In J. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of the International Conference on Simulation of Adaptive Behavior*. MIT Press, 1990.
10. R.S. Sutton. Reinforcement learning architectures for animats. In J. Meyer and S.W. Wilson, editors, *From Animals to Animats: Proceedings of the International Conference on Simulation of Adaptive Behavior*. MIT Press, 1990.
11. R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, 1998.