Sparse Tabular Multiagent Q-learning*

Jelle R. Kok Nikos Vlassis
Informatics Institute, Faculty of Science, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
{*jellekok,vlassis*}*@science.uva.nl*

Abstract

Multiagent learning problems can in principle be solved by treating the joint actions of the agents as single actions and applying single-agent Q-learning. However, the number of joint actions is exponential in the number of agents, rendering this approach infeasible for most problems. In this paper we investigate a sparse representation of the Q-function by only considering the joint actions in those states in which coordination is actually required. In all other states single-agent Q-learning is applied. This offers a compact state-action value representation, without compromising much in terms of solution quality. We have performed experiments in the predator-prey domain and compared our method to other multiagent reinforcement learning methods with promising results.

1 Introduction

A multiagent system (MAS) consists of a group of agents that can interact with each other (Vlassis, 2003). When these agents have common interests, their actions become dependent and have to be coordinated. That is, the agents have to select the individual actions which benefit the group as a whole.

Reinforcement learning (RL) techniques (Sutton and Barto, 1998) have been applied successfully in many single-agent systems for learning the policy of the agent in uncertain environments. However, extending RL to the multiagent case is not that straightforward. One of the simplest approaches is to model the complete multiagent system as one 'big' single agent and thus treat the joint actions of the agents as single actions (joint action learners). Although this approach leads to the optimal solution if sufficient exploration is allowed, it is infeasible for problems with many agents since the joint action space explodes.

Another approach is to let the different agents learn their policy independent of the other agents (independent learners). Each agent treats the other agents as part of the environment and applies single-agent reinforcement learning. The main disadvantage of this approach is that the underlying environment transition model depends on the policy of the other learning agents, making it non-stationary. Although a stationary environment is one of the prerequisites for the convergence proof (Watkins and Dayan, 1992), this method has been applied successfully in multiple cases (Tan, 1993; Sen et al., 1994).

On the other hand, in many problems the agents only need to coordinate their actions in a few states, depending on the specific context (Guestrin et al., 2002b). Even if these 'coordinated' states are known to the designer in advance, it is not obvious how RL techniques can be applied to learn values for these states (or for state-action pairs).

In this paper we propose a multiagent RL method which combines the above ideas. Our method, called *sparse tabular Q-learning*, tries to learn joint action values on those states where the agents need to coordinate explicitly. In all other (uncoordinated) states, we apply the independent learning approach where the method learns individual action values. The main idea is that by specifying the coordinated states beforehand, the agents only have to learn to coordinate their actions in these specific situations. Since in practical problems the agents typically need to coordinate their actions only

* Appeared in the proceedings of Benelearn 2004: Proceedings of the Annual Machine Learning Conference of Belgium and The Netherlands.

in few states, the proposed framework allows for a sparse representation of the complete state-action value function, resulting in large computational savings.

The setup of the paper is as follows. In Section 2 we review the framework of Markov Decision Processes (MDP) and Q-learning. In Section 3 we discuss our approach in which we only learn the joint action values in predefined states. In Section 4 we describe the results of our experiments in the predator-prey domain and compare them to other multiagent Q-learning approaches. Finally, we end with a conclusion and discussion in Section 5.

2 Markov Decision Processes and Q-learning

In this section, we review the Markov Decision Process (MDP) framework. An observable MDP is a tuple $\langle S, A, R, P \rangle$ where S is a finite set of world states; A is a set of actions; $R : S \times A \rightarrow \mathbb{R}$ is a reward function that returns the reward $R(s, a)$ obtained after taking action a in state s and $P : S \times A \times S \rightarrow [0, 1]$ is the Markovian transition function that describes the probability $P(s'|s, a)$ of ending up in state s' when performing action a in state s. The Markov property implies that the state of the world at time t provides a complete description of the history before time t.

An agent's policy is denoted by $\pi : S \rightarrow A$ which maps a state $s \in S$ to the action $a \in A$ that the agent will take. The objective is to find an optimal policy π^* that maximizes the utility

$$U^*(s) = \max_{\pi} E\left[\sum_{t=0}^{\infty} \gamma^t R(s_t, \pi(s_t)) | s_0 = s\right] \quad (1)$$

for each state s. The expectation operator $E[\cdot]$ averages over reward and stochastic transitions and $\gamma \in [0, 1)$ is the discount factor. This value thus represents the expected future discounted reward for each state s and given policy π. We can also represent this using Q-values which explicitly store the expected discounted future reward for each state s and possible action a: $Q^*(s, a) = R(s, a) + \gamma \sum_{s'} P(s'|s, a) \max_{a'} Q^*(s', a')$. At any state s, the optimal policy is then obtained by choosing the action $\arg\max_a Q^*(s, a)$.

Reinforcement learning (RL) methods (Sutton and Barto, 1998) can be applied to estimate $Q^*(s, a)$ by interacting directly with the environment. Q-learning is a widely used learning method when the transition model is unavailable (Watkins, 1989). This method starts with an initial estimate $Q(s, a)$ for each state-action pair. During exploration it updates the Q-values based on the received reward $R(s, a)$ and the perceived state transition from s to s' using

$$\begin{aligned} Q(s, a) := (1 - \alpha) Q(s, a) + \\ \alpha \big[R(s, a) + \gamma \max_{a'} Q(s', a')\big] \end{aligned} \quad (2)$$

where $\alpha \in (0, 1)$ is the learning rate that specifies the incremental update. Tabular Q-learning converges to the optimal $Q^*(s, a)$ when all state-action pairs are visited infinitely often by means of an appropriate exploration strategy.

Extending the above methods to multiagent systems involves several issues, for instance whether the agents share the same reward, whether they observe the selected joint action, whether they model each other, etc. In our case, we assume that each agent receives an individual reward $R_i(s, a)$ based on the selected joint action a in state s. We assume that in states where the action of an agent is independent of the other agents, the reward is completely based on the action taken by this agent. For example, when two cleaning robots are working in two separate rooms and one performs an action resulting in a negative reward (e.g., bumping into a wall), only this agent will receive a negative reward since it makes no sense to incorporate this in the reward function of the agent working in the other room. On the other hand, if an agent is in a coordinated state, it receives the same reward as the agents it is coordinating with (e.g., both agents will receive the same negative reward when they are bumping into each other).

The most direct approach to extend MDP and Q-learning to a multiagent environment is to regard the complete system as one large single agent in which the joint action is regarded as a single action. We will call this approach, where each agent learns the value of joint actions as if they were single actions, MDP learners. This can be implemented by either assuming a central controller that communicates to each agent its individual action, or by making certain common knowledge assumptions about the agents. For instance, the problem of knowing whether the other agents are taking an exploration action can be solved by assuming that the agents are using the same random number generator and the same seed, and these facts are common knowledge among all agents (Vlassis, 2003). Although treating the multiagent environment as a single agent leads to the optimal solution, it is infeasible for problems with many agents since the joint action space becomes prohibitively large.

A second approach is to let each agent learn its strategy independently from the other agents and regard the other agents as part of the environment (Tan, 1993; Sen et al., 1994). We will refer to this method as independent learners (IL). The convergence proof for Q-learning does not hold in this case, since the underlying transition model becomes non-stationary because of its dependence on the policy of the other learning agents (Watkins and Dayan, 1992).

3 Sparse Tabular Q-learning

In many problems, agents only have to coordinate their actions in a specific context (Guestrin et al., 2002b). For example, two cleaning robots only have to take care that they do not obstruct each other when they are cleaning the same room. When they are working in two separate parts of the building, they can work independently. These types of context-specific coordination requirements are explicitly utilized in our method.

The main idea is to use a sparse representation of the joint state-action space by specifying in which states the agents do (and in which they do not) have to coordinate their actions. Roughly, in the uncoordinated states the agents apply the IL method, and in the coordinated states the agents apply the MDP learners approach.

Because of the distinction in action types for different states, we also have to distinguish between representations for the Q-values. Each agent i in our system maintains a single-action value table Q_i for the uncoordinated states, and one joint action value table for the coordinated states[1]. When the agents move from coordinated to uncoordinated states and vice versa, values from the different Q-tables have to be combined in order to update the Q-values during learning. There are four different situations that must be taken into account.

3.1 Uncoordinated to uncoordinated state

When moving from an uncoordinated state s to another uncoordinated state s' an agent acts independently with respect to the other agents and we can apply single-agent reinforcement learning. The table Q_i which only depends on the individual action of agent i is thus updated as in Eq. (2):

$$Q_i(s,a_i) := (1-\alpha)Q_i(s,a_i) + \alpha\big[R_i(s,a_i) + \gamma \max_{a'_i} Q_i(s',a'_i)\big]. \quad (3)$$

As mentioned earlier, the received reward R_i in an uncoordinated state is local. This ensures that the rewards of the other agents cannot influence this update rule.

3.2 Coordinated to coordinated state

When moving from a coordinated state s to another coordinated state s', the update rule is also equal to Eq. (2) since we again update the Q-value based on the future discounted reward from the same table:

$$Q(s,a) := (1-\alpha)Q(s,a) + \alpha\big[R(s,a) + \gamma \max_{a'} Q(s',a')\big]. \quad (4)$$

Now we update the shared Q-table based on the global reward R. In our implementation the global reward $R(s,a)$ for the joint action a is defined as $\sum_i R_i(s,a_i)$. The sum can be justified since we assume the different local rewards are assigned independently and we only consider common interest situations.

3.3 Coordinated to uncoordinated state

When moving from a coordinated state s to an uncoordinated state s' we have to back up the sum of expected future discounted reward from the different independent Q-tables to the shared

[1]Note that this shared Q-table is either stored centralized or updated in exactly the same way by all individual agents.

Q-table. We do that as follows:

$$Q(s,a) := (1-\alpha)Q(s,a) + \alpha\Big[R(s,a) + \gamma \sum_i \max_{a'_i} Q_i(s',a'_i)\Big] \quad (5)$$

The global Q-value in the uncoordinated state s' equals the sum of all individual Q-values.

3.4 Uncoordinated to coordinated state

When moving from an uncoordinated state s to a coordinated state s' we have to back up the expected future discounted reward from the shared Q-table back to the different independent tables. One approach is to divide this value equally among the independent tables. The update rule then looks as follows:

$$Q_i(s,a_i) := (1-\alpha)Q_i(s,a_i) + \alpha\big[R_i(s,a_i) + \gamma\frac{1}{n}\max_{a'} Q(s',a')\big] \quad (6)$$

where n equals the number of agents. When moving from an uncoordinated state, each agent is rewarded (either negative or positive) with the same fraction of the expected future discounted reward from the resulting coordinated state. Each agent thus contributes equally to the coordination. Other distribution schemes are also possible (see conclusions).

4 Experiments

We have applied our method to the well-known Predator-Prey (or Pursuit) Domain (Kok and Vlassis, 2003) in which the goal of the predators is to capture the prey as fast as possible in a discrete grid-like world. We have concentrated on a coordinated problem in which two predators in a fully observable 10×10 toroidal grid have to capture a single prey. An example grid with two predators and one prey is depicted in Figure 1. Each agent can move deterministically to one of the adjacent cells or stand still. In our case, the prey is captured when both predators are located in an adjacent cell to the prey and only one of the two predators moves to the location of the prey. In Figure 2 the prey can be caught using one of two action combinations. Either the predator above the prey remains on its current position and the other predator moves to the cell to its left or the

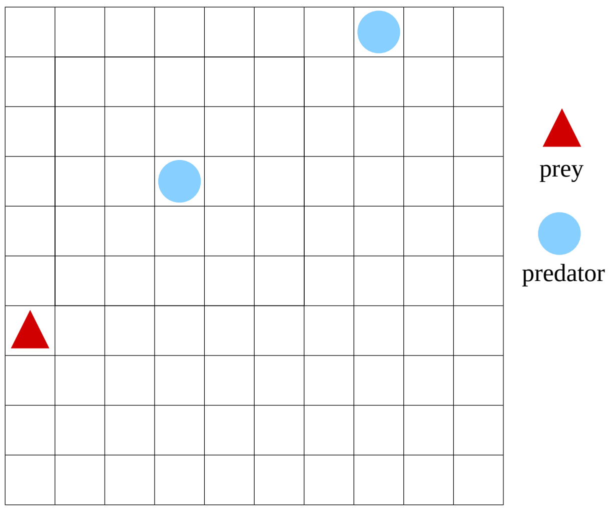

Figure 1: An example situation with two predators and one prey.

upper predator moves down to the prey position and the other predator remains on its current position. For all other combinations in which one of the predators moves to the prey position, the predator that moves to the prey is penalized and placed on a random position on the field. Predators are also penalized when they collide with each other by moving to the same cell (independent of the prey position). The policy of the prey is fixed. It remains on its current position with a probability of 0.2. In all other cases it moves to one of the free adjacent cells with uniform probability.

To apply our approach to this problem, we create three Q-tables. The first table relates to the shared Q-table which stores the Q-values for the joint actions of the agents when located in a coordinated state. A state is defined as the position of the two predators relative to the prey position. We assume that the predators only have to coordinate their actions when they are close to each other, so we only add states to this Q-table corresponding to the following situations:

- the (Manhattan) distance to the other predator is smaller or equal than two cells.
- both predators are within a distance of two cells to the prey.

The other two tables represent the policy for the agents when located in all other (un-

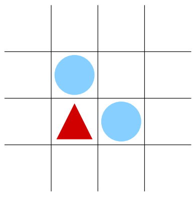

Figure 2: Possible capture position for the two predators.

coordinated) states. For this particular problem, this results in $1,248$ coordinated states and $8,454$ uncoordinated states. So in total, $115,740\,(= 1,248 \cdot 5^2 + 2 \cdot 8,454 \cdot 5)$ state-action pairs are updated.

During learning we use the four update rules from Section 3 to update the values in the three Q-tables. In all cases each predator receives a local reward of 37.5 when helping in capturing the prey and receives a negative local reward of -50.0 when colliding with another predator. This relates to a global reward of respectively 75.0 and -100.0 in the case of two agents. When moving to the prey without support that agent receives a reward of -5.0. In all other cases the reward is -1.0. Finally, we use one-step Q-learning with a Boltzmann distribution

$$\frac{e^{Q(s,a)/T}}{\sum_{a^i} e^{Q(s,a^i)/T}} \tag{7}$$

with a temperate T of 0.4 for action selection, a learning rate α of 0.3, and a discount factor γ of 0.9.

We compare our method to the two other Q-learning methods, IL and MDP learners, mentioned earlier. In the IL approach case, this corresponds to $48,510\,(= 99 \cdot 98 \cdot 5)$ different state-action pairs for each agent. For the second comparison, we model both agents as a complete MDP with the joint action represented as a single action for all possible states. In this case the number of state action-pairs equals $242,550\,(= 99 \cdot 98 \cdot 5^2)$. Finally, we also compared the three methods with a manually implemented policy in which a predator first minimizes its distance to the prey and then moves along with the prey until the other predator is also located next to the prey. Next, based on social conventions, one predator moves to the prey position and the other remains on its current position in order to capture the prey.

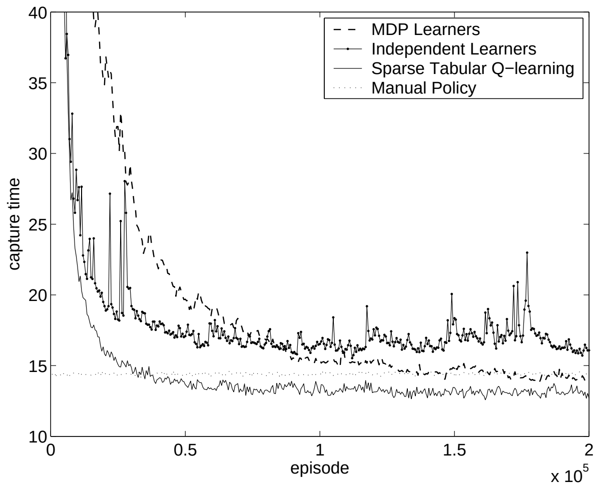

Figure 3: Capture times for the learned policy for the three different methods for the first 200,000 episodes. Results are averaged over 10 runs.

Figure 3 shows the capture times for the learned policy during the first 200,000 episodes for the different methods. The results are generated by running the current learned policy after each interval of 500 episodes five times on a fixed set of 100 starting configurations. During these 500 test episodes no exploration actions were performed. This complete procedure was repeated for 10 different runs. The 100 starting configurations were selected randomly beforehand and were used during all 10 runs.

Both the independent learners and our proposed method learns quickly in the beginning with respect to the MDP learners. This is caused by the fact that learning is based on fewer state-action pairs. However, the independent learners do not converge to a single policy but keep oscillating. This is caused by the fact that they do not take the action of the other agent into account. When both predators are located next to the prey and one predator moves to the prey position, this predator is not able to distinguish between the situation where the other predator remains on its current position or performs one of its other actions. In the first case a positive reward is returned, while in the second case a large negative reward is received. However, in both situations the same Q-value is updated.

These coordination dependencies are explicitly taken into account in the two other ap-

Method	**Avg. capt. time**
Independent learners	16.08
Manual policy	14.44
Sparse Tabular	12.70
MDP Learners	13.92

Table 1: Capture times of the different methods after 200,000 episodes. All results are averaged over 5000 episodes.

proaches. In the MDP learners approach, these dependencies are taken into account for every state which results in a slowly decreasing learning curve because of the large state-action space. The sparse tabular approach has a quicker decreasing learning curve because only joint actions are considered for these states in which the agents have to coordinate their actions.

Table 4 shows the average capture times for the different approaches. After 200,000 episodes our method has the best average capture time, while the MDP learners approach still hasn't converged completely. Experiments show that only after approximately 450,000 episodes, the MDP learners approach is able to improve upon the policy learned by the sparse tabular method (which already converges around approximately 60,000 episodes) resulting in an average capture time of 12.34. This is better than the sparse tabular approach which is caused by the fact that not all necessary coordination requirements are added to the shared Q-table. We explicitly assumed that the predators did not have to coordinate when they were far away from each other, although already coordinating in these situations might have an additional positive influence on the final result. Of course, it is possible to extend the shared Q-table with more states, but that would decrease the learning speed because of the increased state-action space.

5 Conclusions and discussion

In this paper, we discussed a sparse tabular multiagent Q-learning approach. The proposed method utilizes the fact that many states in a multiagent problem do not require any coordination between the agents. Beforehand, we specify the states in which the agents have to coordinate their actions and then apply joint action Q-learning to learn the policy of the agents in those states. The independent learners approach is used for the states in which the agents do not have to coordinate their actions. During learning rewards received in the specified coordinated states are propagated back to the uncoordinated states which causes agents to learn to move to coordinated states. The proposed method offers large savings in terms of the state-action representation, without significantly affecting the solution quality.

Results in the predator-prey domain showed that this method improves the learning time considerably and the final results are comparable to the optimal policy. Furthermore, our method clearly outperforms the IL approach that was not able to converge for the given problem.

The crucial point in our approach is the way we back up the expected future discounted reward when moving from a uncoordinated state to a coordinated state. Our choice of dividing the joint Q-value equally among the agents seems reasonable, but other options include dividing the value proportionally to the local immediate reward or to the individual Q-values. We are currently performing more experiments with these different settings.

Our approach bears resemblance to the coordinated reinforcement learning approach of (Guestrin et al., 2002a) where the global Q-function is explicitly represented as the sum of local Q-functions and the learning rule is defined accordingly. The main difference is that in that work the agents are always updating their local Q-functions based on the best joint action, while in our case the updates may also involve the best individual actions of the agents (e.g., in Eq. (5)).

In this paper, we only investigated tabular environments. As future work, we like to apply our method also to continuous environments where the state is represented using function approximation techniques as CMAC (Watkins, 1989) (also known as tile coding (Sutton and Barto, 1998)).

Another interesting direction is to see whether the coordination dependencies which are now set beforehand, can be learned automatically.

Acknowledgements

This research is supported by PROGRESS, the embedded systems research program of the Dutch organization for Scientific Research NWO, the Dutch Ministry of Economic Affairs and the Technology Foundation STW, project AES 5414.

References

C. Guestrin, M. Lagoudakis, and R. Parr. 2002a. Coordinated reinforcement learning. In *Proc. 19th Int. Conf. on Machine Learning*, Sydney, Australia, July.

C. Guestrin, S. Venkataraman, and D. Koller. 2002b. Context-specific multiagent coordination and planning with factored MDPs. In *Proc. 8th Nation. Conf. on Artificial Intelligence*, pages 253–259, Edmonton, Canada, July.

J. R. Kok and N. Vlassis. 2003. The pursuit domain package. Technical Report IAS-UVA-03-03, Informatics Institute, University of Amsterdam, The Netherlands, August.

S. Sen, M. Sekaran, and J. Hale. 1994. Learning to coordinate without sharing information. In *Proc. 12th Nation. Conf. on Artificial Intelligence*, Seattle, WA.

R. S. Sutton and A. G. Barto. 1998. *Reinforcement Learning: An Introduction.* MIT Press, Cambridge, MA.

M. Tan. 1993. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Proc. 10th Int. Conf. on Machine Learning*, Amherst, MA.

N. Vlassis. 2003. A concise introduction to multiagent systems and distributed AI. Informatics Institute, University of Amsterdam, September. http://www.science.uva.nl/~vlassis/cimasdai.

C. Watkins and P. Dayan. 1992. Technical note: Q-learning. *Machine Learning*, 8(3-4):279–292.

C. J. C. H. Watkins. 1989. *Learning from delayed rewards.* PhD thesis, Cambridge University.