Reinforcement Learning in Large State Spaces
Simulated Robotic Soccer as a Testbed

Karl Tuyls, Sam Maes, and Bernard Manderick

Computational Modeling Lab (COMO)
Department of Computer Science
Vrije Universiteit Brussel
Belgium
{ktuyls,sammaes}@vub.ac.be, bernard@arti.vub.ac.be
http://como.vub.ac.be

Abstract. Large state spaces and incomplete information are two problems that stand out in learning in multi-agent systems. In this paper we tackle them both by using a combination of decision trees and Bayesian networks (BNs) to model the environment and the Q-function. Simulated robotic soccer is used as a testbed, since there agents are faced with both large state spaces and incomplete information. The long-term goal of this research is to define generic techniques that allow agents to learn in large-scaled multi-agent systems.

1 Introduction

In this paper we address 2 important problems that occur when learning in multi-agent systems (MAS). The first is a problem of large state spaces. Existing formalisms such as the Markov game model[Hu99][Lit94] suffer from combinatorial explosion, since they learn values for combinations of actions. We suggest to use a combination of decision trees and Bayesian networks (BNs)[Rus94]to avoid this problem of tractability. We will discuss the problem of modeling the environment and other agents acting in the environment in the context of Markov Games. The Markov game model is defined by a set of states S, and a collection of action sets $A_1, ..., A_n$ (one set for every agent). The state transition function $S \times A_1 \times ... \times A_n \rightarrow P(S)$ maps a state and an action from every agent to a probability on S. Each agent has an associated reward function R_i :

$$S \times A_1 \times ... \times A_n \rightarrow \Re \tag{1}$$

where $S \times A_1 \times ... \times A_n$ constitutes a product space. The reward function for an agent A_i calculates a value which indicates how desired the state S and the actions of the agents $A_1, \ldots, A_n$ are for agent A_i.

In this model learning is done in a product space and when the number of agents increases this model becomes prohibitively Agent 1 observes the environment and all other n agents. A table represents the accumulated reward over time according to each possible situation. We represent the environment by a

G.A. Kaminka, P.U. Lima, and R. Rojas (Eds.): RoboCup 2002, LNAI 2752, pp. 319–326, 2003.
© Springer-Verlag Berlin Heidelberg 2003

set S of possible states, and each agent has a set A_i of possible actions. So we have $|S| \times |A_1| \times, ..., \times |A_n|$ possible situations, which need to be stored in a table with a matching reward value. Now if the number of agents increases and the environment becomes more complex this table becomes intractable. This is what we call the large state space problem.

Nowe and Verbeeck [Now00] avoid this problem by forcing an agent to model only the agents that are relevant to him. A similar approach is adopted in [Def01].

The second problem is one of incomplete information. Often an agent is not given all information about the environment and the other agents. For instance we don't always know what action each agent is taking, where the other agents and the ball are at every timestep. We suggest to use Bayesian networks to handle this problem of incompleteness. Bayesian networks are a compact representation of a joint probability distribution, which allows us to make estimates about certain variables, given values of other variables. We believe that BNs can contribute a great deal to this problem.

These two problems that emerge in the setting described above, will be dealt with in the context of simulated robotic soccer, and we will suggest solutions for them.

1.1 Simulated Robotic Soccer as a Testbed

We decided to use simulated robotic soccer as the test bed for our research, because it is particularly well suited for testing the specific problems we are interested in.

- Large state space, it is obvious that 23 objects on a field can be in a massive amount of states, especially if we take into account the speed, the acceleration, the direction of these objects next to their position on the field.
- Incomplete information, each agent has only a limited and noisy view of the environment.

Both the presence of these characteristics and the competitive aspect of RoboCup make it a challenging domain to work in.

In section 2 we will discuss our general solution to the problems stated in this introduction. Section 3 will discuss the methodology we will follow to test the presented ideas and section 4 will present some results. Finally we will end this paper with a conclusion.

2 Modeling the Environment and Other Agents

In this section we will present our solutions to the problems described in the introduction. We start with a small introduction on Q-learning, the form of learning under study.

2.1 Q-learning

We use a variation of reinforcement learning called single agent Q-learning. In this form of learning the Q-function maps state-action pairs to values. Let $Q^*(s,a)$ be the expected discounted reinforcement of taking action a in state s, then continuing by choosing the optimal actions. We can estimate this function by the following Q-learning rule

$$Q(s_t,a_t) = Q(s_t,a_t) + \alpha[r_{t+1} + \gamma max_a Q(s_{t+1},a) - Q(s_t,a_t)] \tag{2}$$

where γ is a discount factor and where α is the learning rate. r_{t+1} represents the immediate reward at time $t+1$. If each action is executed in each state an infinite number of times on an infinite run and α is decayed properly, the Q-values will converge with probability 1 to Q^*.

We will use this form of learning for learning individual skills in the simulated robotic soccer, like for instance learning to go to the ball, or learning to shoot at goal. When we come in a more challenging situation with other agents present (for example in a 2-2 situation), this Q-learning rule has to be converted to a multi-agent Q-learning rule, looking like

$$\begin{aligned} Q(s_t,a_t^1,...,a_t^n) = {} & Q(s_t,a_t^1,...,a_t^n) \\ & + \alpha[r_{t+1} + \gamma max_{a^1,...,a^n} Q(s_{t+1},a^1,...,a^n) - Q(s_t,a_t^1,...,a_t^n)] \end{aligned}$$

where $a_t^1,...,a_t^n$ present the actions of agent 1 to n at time t.

2.2 Bayesian Nets for Large State Spaces

We propose to use Bayesian nets (BNs) for modeling the other agents acting in the environment and the environment itself. A Bayesian net [Rus94] is a graphical knowledge representation of a joint probability distribution. A Bayesian net has one type of node, more precisely a random node which is associated with a random variable. This random variable represents the agent's possibly uncertain beliefs about the world. The links between the nodes summarize their dependence relationships. BNs can represent a certain domain in a compact manner and this representation is equivalent to the joint probability distribution.

Our idea is to use such Bayesian nets for each agent to model the other agents in the domain. The resulting model must describe how the different agents influence each other and how they influence the reward the agent receives.

Every node of a BN has an associated conditional probability table (CPT), which quantizes the direct influence of the parents on the child node. In figure 1 we give an example BN of a situation with 4 agents and 5 state variables representing the environment. This network represents the view of agent 1, and every other agent has an analogous network representing his view. Without independence relations in the domain, the network would be fully connected. As you can see, every node has a direct influence on the accumulated reward Q.

In figure 2 you can see the associated CPT for node Q. Again this table can become very large, depending on the number of links with Q. So we still have

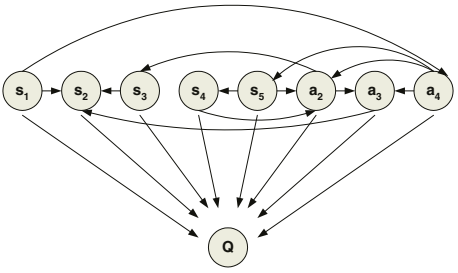

Fig. 1. Large state space solution: BN of a domain with 4 agents and 5 state variables.

s_1	s_2	s_3	s_4	a_2	a_3	a_4	Q for a_1
1	1	1	1	1	1	1	Q
1	1	1	1	1	1	2	Q
1	1	1	1	1	1	3	Q
⋮							
n_{s1}	n_{s2}	n_{s3}	n_{s4}	n_{a2}	n_{a3}	n_{a4}	Q

Fig. 2. The CPT of node Q from figure 1.

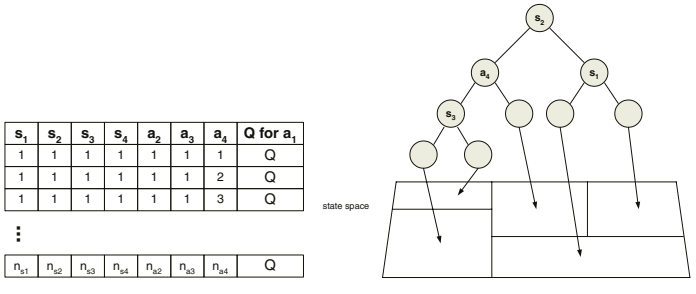

Fig. 3. The CPT of node Q converted to a decision tree.

a large state space problem. We suggest to use decision trees to overcome this problem. This is illustrated in figure 3 where we convert a CPT to a decision tree. On the left you see the classic table lookup with constant resolution and on the right you see a decision tree with varying levels of resolution.

Note that the tree is constructed online, and that it is continuously refined during the Q-learning process. In this way the attention of the agent can be shifted to other areas, by refining the resolution at some point in the state space. For more details on the approximating algorithm we refer to [Pye98].

2.3 Bayesian Nets for Incomplete Information

The second problem that we intend to solve in this paper is that of an agent being faced with incomplete information about the environment. In multi-agent

systems in general and especially in those where a large number of agents are involved, it is common that at any given moment a specific agent can only directly observe a part of the environment and a subset of the agents.

Our solution consists of learning a Bayesian network over the domain. As we have mentioned before a BN is a concise representation of the joint probability distribution of the domain. This means that, given a subset of variables, the BN can be queried to calculate beliefs for the unknown variables. A belief for a variable consists of a probability for each possible state of the variable. In other words, a BN can be used to calculate the most probable value for an unknown variable and that information can then be used to do Q-learning.

For the moment we concentrate our effort at investigating whether the use of a Bayesian network can help agents to have a more realistic and up-to-date view of the environment and of the other agents. Until now we assume that the Bayesian network of the domain is given beforehand, learning a BN online is part of the future work.

3 Methodology

In this section we clarify where we want to go with our research, how we want to get there and also where we stand today.

The ultimate goal of our research is to define generic techniques that allows agents to learn in large-scaled multi-agent systems. To achieve this, two problems stand out. Firstly, large state and action spaces, and secondly agents being faced with incomplete information about the domain and the other agents.

We want to reach this goal by the following distinct steps. Step 1: Using single agent reinforcement learning to learn an agent simple moves, such as controlling the ball, giving a pass, dribbling, etc. using only low-level actions. All this in a setting where the agents can cope with a large state and action space and with incomplete information. This has already been done by other people [Kos99,Sto00] but is indispensable for the rest of our approach.

Step 2: Using multi-agent RL on small groups of agents to learn them skills such as scoring a goal in a situation with 2 attackers vs. 1 defender and a goalkeeper, learning to defend in the same situation, etc.

An example of incomplete information at this point could be that an agent doesn't exactly know where his teammate is, because he isn't facing in that direction. Then the agent uses the information in his Bayesian network to calculate the most probable position of the agent, given the last known position of the agent and all the other agents he can see.

To reduce the complexity of this task, we want to avoid the action selection module of the agent to manipulate low-level actions directly. Instead we want it to make use of the basic skills learned in step 1.

Step 3: Using multi-agent RL on larger groups of agents (entire teams), using Bayesian networks to help the agents in modelling the domain. Additionally we want the BNs to make use of the local structure in the conditional probability distributions (CPDs) that quantify these BNs. To clarify: our approach must take

advantage of the following type of information: an attacker is independent of a defender if they are far from each other, but if they are together on the midfield they are clearly dependent. One way to do this is to insert local structure in the conditional probability distributions of the BNs [Bou96].

Again, to reduce complexity we want to use as much as possible the moves learned in step 2, instead of using basic low-level actions and moves learned during step 1.

At this point in time we are finishing step 1, and starting to tackle the problems associated with step 2. In the next section we will elaborate on what we have achieved so far.

4 Experiments

This section describes some of the experiments that we have conducted.

4.1 Learning to Run to the Ball

The first experiment conducted is an agent who learns to run to the ball. In the learning process he will explore his action set and try to exploit this knowledge to find the optimal policy, namely running to the ball via the shortest path. The experimental settings are as follows : the agent and the ball are put on the field in a random place. Every 1000 steps the ball will be randomly moved to different coordinates.

This simple example illustrates how complicated and large the state space can become. We used Q-learning to learn this skill and the state of the environment is represented by the distance to the ball and the angle of the body with the ball. We considered 2 possible actions for the agent in this situation, *turn* and *dash*. If you discretize the parameters of both actions in 10 intervals, you have alltogether 20 actions. With the default dimensions of the field[1] a player can be at most 125 meter from the ball. If we assume that the distance and the angle are discretized respectively to 1 meter and 1 degree, this makes a total of $125*360*20 = 900.000$ situations for which Q-values have to be learned. So learning with the classical table lookup method can demand quite some resources, even for simple player skills.

As explained in section 2.2 we take advantage of the fact that a lot of these situations are quite similar and that in some cases a Q-value can be associated with a set of situations instead of one situation. In figure 4 you can see a soccer field that is divided in planes with only one Q-value for each plane.

4.2 Learning to Dribble

This section describes an experiment where the goal was to learn the agent to go to the ball and to run with the ball.

[1] 68 * 105 meters

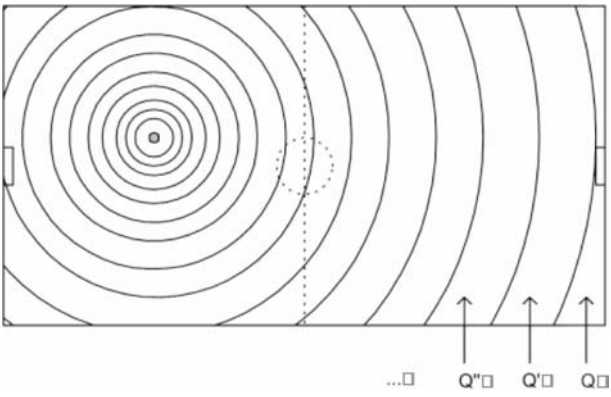

Fig. 4. An example of how a decision tree approach would split the field in regions with an equal Q-value.

We used the same variables to represent the state of the environment as in the previous experiment, but in this case an agent is capable of doing 3 actions: *turn*, *dash* and *kick*. We had to extend the reward function used in the previous experiment so that an agent is not only rewarded when he is close to the ball, but also when he performs a run with the ball.

5 Conclusion

In this paper we introduced a generic solution for learning in multi-agent systems that is able to cope with two important problems in MAS. Firstly, that of learning in large state and action spaces. Secondly, that of an agent being faced with incomplete information about the environment and other agents.

We propose to use Bayesian networks with the conditional probability distributions represented by decision trees instead of classical table lookup to solve the first problem. This reduces the size and complexity of the state and action space, because it causes Q-values to be associated with regions in the state space instead of having to learn a Q-value for every single point in the state space.

For the second problem, we propose to keep a model of the environment in a concise manner. Again we use a Bayesian network to do this, since it is a compact representation of the joint probability distribution over the environment. In this way estimates can be calculated for variables representing a part of the environment that hasn't been observed in recent timesteps.

In our experiments we prove that the learning approach is feasible for an agent running to the ball and dribbling the ball.

6 Future Work

– Allow pruning in the decision tree, so that an agent can also decrease his attention for a specific area of the state space.

– Use other techniques as decisiontrees that learn an adaptive-resolution model, such as the Parti-game algorithm [Moo95].
– Learning the Bayesian network that represents the environment and the other agents online and adaptively.

References

Bou96. Boutilier, C., Friedman, N., Goldszmidt, M., and Koller, D. Context-specific independence in Bayesian networks. In Proc. UAI, 1996.

Def01. Defaweux, A., Lenaerts, T., Maes, S., Tuyls, K., van Remortel, P., Verbeeck, K., Niching and Evolutionary Transitions in MAS. Submitted at ECOMAS-GECCO 2001.

Hu99. Hu, J., Wellman, M. P., Multiagent reinforcement learning in stochastic games. Submitted for publication, 1999.

Kos99. Kostiadis, K., Hu, H., Reinforcement Learning and Co-operation in a Simulated Multi-agent System. Proc. of IEEE/RJS IROS'99, Korea. 1999.

Lit94. Litmann M.L., Markov games as a framework for multi-agent reinforcement learning. Proceedings of the Eleventh International Conference on Machine Learning, pages 157–163, 1994.

Moo95. Moore, A. W., and Atkeson, C. The Parti-game Algorithm for Variable Resolution Reinforcement Learning in Multidimensional State Space. Machine Learning Journal, 21, 1995.

Nod98. Noda, I., Matsubara, H., Hiraki, K., and Frank, I., Soccer Server: A Tool for Research om Multiagent Systems. Applied Artificial Intelligence, 12:233-250, 1998.

Now00. Nowe, A., Verbeeck, K., Learning Automata and Pareto Optimality for Coordination in MAS. Technical report, COMO, Vrije Universiteit Brussel, 2000.

Pea88. Pearl, J., Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. Morgan Kaufmann, San Mateo, CA, 1988.

Pye98. Pyeatt, L.D., Howe, A.E., Decision Tree Function Approximation in Reinforcement Learning. Technical Report CS-98-112, Colorado State University, 1998.

Rus94. Russell, S., Norvig, P., Artificial Intelligence: a Modern Approach. Prentice Hall Series in Artificial Intelligence. Englewood Cliffs, New Jersey, 1995.

Sto00. Stone, P., Layered Learning in Multiagent Systems. A Winning Approach to Robotic Soccer. MIT Press, 2000.

Sut98. Sutton, R.S., Barto, A.G., Reinforcement Learning: An Introduction, Cambridge, MA: MIT Press, 1998.