Joint Learning in Stochastic Games: Playing Coordination Games Within Coalitions

Ana L. C. Bazzan

Instituto de Informática – UFRGS, Caixa Postal 15064, 91.501-970 Porto Alegre, RS, Brazil
bazzan@inf.ufrgs.br

ABSTRACT

Despite the progress in multiagent reinforcement learning via formalisms based on stochastic games, these have difficulties coping with a high number of agents due to the combinatorial explosion in the number of joint actions. One possible way to reduce the complexity of the problem is to let agents form groups of limited size so that the number of the joint actions is reduced. This paper investigates the task of multiagent reinforcement learning where individual agents learn within coalitions. The context of these learning tasks are coordination games, a class of games with multiple pure strategy Nash equilibria, which is of broad interest in social sciences such as choice of standards for new products. Experimental results show that the reward converges to values close to the optimum.

1. INTRODUCTION

The problems posed by many actors in a multi-agent reinforcement learning (MARL) scenario are inherently more complex than in single agent reinforcement learning (SARL). While one agent is trying to model the environment (other agents included), other agents are doing the same (or at least reacting). This produces an environment that is inherently non-stationary. Thus, the notions of convergence as previously known (e.g. Q-learning) cannot be guaranteed any longer. One popular formalism for MARL is the one based on stochastic games (SG), which is investigated by game theory and is an extension of the basic Markov decision processes (MDP). However, the aforementioned increase in complexity has many consequences arising from the use of this formalism. First, the approaches proposed for the case of general sum SG require that several assumptions be made regarding the game structure (agents' knowledge, self-play etc.). Also, it is rarely stated what agents must know in order to use a particular approach. Those assumptions restrain the convergence results to common payoff (team) games and other special cases such as zero-sum games. Moreover, the focus is normally put on two-agent games, and not infrequently, two-action stage games. Otherwise, an oracle is needed if one wants to deal with the problem of equilibrium selection when two or more equilibria exist. Second, despite recent results on formalizing multiagent reinforcement learning using SG, these cannot be used for systems of many agents, *if any flavor of joint-action is explicitly considered*, unless the obligation of visiting all pairs of state-action is relaxed, which has impacts on the convergence. The problem with using a high number of agents happens mainly due to the exponential increase in the space of *joint* actions.

Up to now, these issues have prevented the use of SG-based MARL in real-world problems, unless simplifications are made, such as letting each agent learn *individually* using single-agent based approaches. It is well-known that this approach is not effective since agents converge to sub-optimal states. In practice this means that, often, the problem can be solved neither in a centralized way nor in a completely distributed one. In the former case, computational complexity issues play a fundamental role, while in the latter, agents' actions cause non stationarity in the environment. Therefore, partitioning the problem in several, smaller multiagent systems may be a good compromise between complete distribution and complete centralization. Having this picture in mind, the goal of the present paper is to address a many-agent system in which these are grouped in a flat organization, based on coalition formation. The idea here is that each coalition must have a small size so that the size of the Q-value tables (or any equivalent to it) is computationally tractable. Therefore the approach we follow here is to take advantage of a natural clustering of a particular structure of the relationships among the agents. In our case we consider interactions in a grid. It has been shown that spatial interactions do matter in game playing. Moreover, our approach targets real-world systems problems with the following characteristics: they are comprised of a high number of agents; agents act and interact in environments that are dynamic due to the nature of agents' interactions.

2. MULTIAGENT LEARNING

Reinforcement learning (RL) problems can be modeled as Markov Decision Processes (MDPs). These are described by a set of states, $\mathcal{S}$, a set of actions, $\mathcal{A}$, a reward function $R(s, a) \rightarrow \Re$ and a probabilistic state transition function $T(s, a, s') \rightarrow [0, 1]$. An experience tuple $\langle s, a, s', r \rangle$ denotes the fact that the agent was in state s, performed action a and ended up in s' with reward r. Given an MDP, the goal is to calculate the optimal policy π^*, which is a mapping from

states to actions such that the discounted future reward is maximized.

Q-Learning works by estimating state–action values, the Q-values, which are numerical estimators of quality for a given pair of state and action. More precisely, a Q-value $Q(s, a)$ represents the maximum discounted sum of future rewards an agent can expect to receive if it starts in state s, chooses action a and then continues to follow an optimal policy. Q-Learning algorithm approximates $Q(s, a)$ as the agent acts in a given environment. The update rule for each experience tuple $\langle s, a, s', r \rangle$ is:

$$Q(s, a) \leftarrow Q(s, a) + \alpha \left(r + \gamma \, max_{a'} \, Q(s', a') - Q(s, a) \right) \ (1)$$

where α is the learning rate and γ is the discount for future rewards. If all pairs state-action are visited during the learning process, then Q-learning is guaranteed to converge to the correct Q-values with probability one.

Learning in systems with two or more players has a long history in game-theory. The connection between multiagent systems and game-theory in what regards learning has been explored as well at least from the 1990's. Thus, it seems natural to the reinforcement learning community to explore the existing formalisms behind stochastic (Markov) games (SG) as an extension for MDPs also called MMDP.

Coordination game is a type of matrix games, normally with a single state (stateless). In matrix games, agents aim at maximizing their payoffs or rewards, given the actions available to them. Actions can be selected according to a pure strategy (one that puts probability one in one single action), or according to a mixed strategy (there is a probability distribution over the available actions). A strategy for player i is denoted σ_i, while σ_{-i} denotes the joint strategies of all players but i. Similarly A_{-i} is the joint actions of all players excluding i.

In a SG the solution of the problem from the perspective of player i is to find the best response to σ_{-i}. All games have at least one equilibrium, possibly in mixed strategies. One issue is that some kinds of games have clearly more than one of these points so the selection of one of them – the coordination task – is not trivial. For instance, in pure coordination games, all agents have the same reward. Besides, in pure coordination games there will always exist a Pareto optimal equilibrium, but this need not be unique. Nowadays coordination games are of great interest to model for instance the establishment of standard for innovative technologies (e.g. media, music/video players, etc.). Another example of a coordination game is the following: two students are trying to decide which computer and operating system to use. Both will profit more if they decide to use the same system so that they can exchange software and skills. Considering the payoff matrix shown in Table 1 by playing $< a_0, b_0 >$ or $< a_1, b_1 >$ players have no reason to deviate. However when $\eta_0 > \eta_1$ the former is the Pareto-dominant one.

		Agent 1	
		a_0	a_1
	b_0	η_0 / η_0	0 / 0
Agent 2			
	b_1	0 / 0	η_1 / η_1

Table 1: Payoff-matrix for a coordination game with two equilibria ($\eta_0 > \eta_1 > 0$)

There are several approaches to achieve coordination in this kind of games (e.g. especially when there is one Pareto dominant equilibrium. However, a different situation arises if this is not the case. The game shown in Table 1 has two equally probable equilibria when $\eta_0 = \eta_1$. In [1] such a co-ordination game is used to investigate what the authors call individual learners (IL) and joint-action learns (JAL). Due to the stochastic nature of the selections, the convergence to a coordinated equilibrium (e.g. $< a_0, b_0 >$ or $< a_1, b_1 >$) is not guaranteed. Thus their approach tries to have agents explicitly modeling their opponents, assuming that these are playing according to a stationary policy. This is done via an estimation of the probability with which opponents will play a joint action, based on the past plays (thus a version of ficticious play). Agent i then plays its best response to this estimated distribution σ_{-i}.

Most of the research on SG-based MARL so far has been based on a static, two-agent stage game (i.e. a repeated game) with common payoff (payoff is the same for both agents), and with few actions available as in [1]. The zero-sum case was discussed in [3] and attempts of generalizations to general-sum SG appeared in [2], among many others.

The issue of coordination games with two equally probable equilibria has received some attention too. In [1] miscoordination is addressed by means of biasing the exploration. In the present paper, we depart from this explicit biased exploration by means of supervised learning and coalition formation, both based on groups. For example, within a coalition agents are committed with one of the two actions and this commitment is made involving neighbors that share similar Q-value patterns (policies).

3. COPING WITH JOINT LEARNING

As mentioned, this paper approaches multiagent learning via multiagent Markov decision process (MMDP) or stochastic games (SG), a generalization of a MDP for n agents. An n-agent SG is a tuple (N, S, A, R, T) where: $N = 1, ..., i, ...n$ is the set of agents; S is the discrete state space (set of n-agent stage games); $A = \times A^i$ is the discrete action space (set of joint actions); R^i is the reward function (R determines the payoff for agent i as $r^i : S \times A^1 \times \ldots \times A^k \to \Re$); and T is the transition probability map (set of probability distributions over the state space S).

In particular, here we follow the setting by [1] which addresses repeated games with $|S| = 1$. However, contrarily to previous works, we let agents play the game repeatedly with m other agents, all positioned in a grid. Using the approach proposed in [1] if agents all keep mappings of their joint actions, this would imply that each agent needs to maintain tables whose sizes are exponential in the number of agents: $|S^1| \times \ldots \times |S^n| \times |A^1| \times \ldots \times |A^n|$. This is hard even if, as said, $|S| = 1$. For example, assuming that agents playing the repeated game have only two actions, the size of the tables is $2^{|N|}$. Therefore one possible approach is to partition the agents to decrease $|N|$. Even doing this partition, it is necessary to redefine Bellman's equations for the multiagent environment.

In the individual learning, for policy update, the standard reinforcement learning algorithm is used (Equation 1): each agent keeps one single Q table where the rewards received by playing with the m "opponents" are collected. This avoids agents having to know what the opponents have played (as no joint actions are recorded). For action selection Boltz-

mann exploration is used with parameters as in [1], when possible. As these authors have noticed, because, for each action i_0 and i_1 of agent i there is a 50% probability that agent j selects j_0 or j_1, the Q-values for both actions i_0 and i_1 converge to the same value. Of course due to the stochastic nature of the selections and the decay in learning rate, one can expect the Q-values, once these converge, to be distinct for both actions. Thus two agents would prefer one action to the other. Unfortunately these preferences are not necessarily coordinated. Thus, in the case of games like that in Table 1, the performance of individual learners is poor.

The joint learning algorithm is adapted from [1]. Few modifications are introduced for the case where agents interact in a grid having m "opponents". Thus we only discuss the relevant issues here. In order to deal with the m opponents in a simple way, each agent keeps m Q tables. Since the agents are located in a non-toroidal grid, each must keep four tables: one for each close neighbor (with the exception of border agents that have less neighbors). Of course this assumes that each agent sees the actions of the others, an assumption that may pose questions on the communication demand.

As noted by Claus and Boutilier, joint-action learners do not necessarily performs better than individual learners. This happens because, despite the ability of agents to distinguish the Q-values for different joint actions, the use of this "information" is circumscribed by the action selection mechanism which uses Boltzmann exploration. This exploration does not allow the agents to fully exploit the Q-values of joint actions.

As mentioned before, one issue that has attracted many attention in multiagent systems is how to partition or organize a multiagent system in an effective way. Unfortunately, partitioning agents in coalitions that lead to an efficient utility is not a trivial problem. In the general case, the number of coalition structures ($O(|N|^{|N|})$) is so large that it cannot be enumerated for more than a few agents [4]. Considering all joint actions is not possible due to the combinatorial explosion of pairs state–action. On the other hand, it is more efficient than single-agent reinforcement learning because at least some joint actions are considered. We show here that for games with particular structures and where agents have particular characteristics (e.g. they form a network in which the neighborhood plays a role), coalitions among neighbors make sense and help agents to collect a much higher payoff. Because only coalitions among neighboring agents are initially formed, the number of coalition structures are smaller than $|N|^{|N|}$. This does not mean that coalitions are restricted to four or five agents. Rather, they may grow as agents in the initially formed coalitions may propose to their immediate neighbors to join and so forth.

Before coalitions can be formed, agents act as individual learners. After some time steps which are used to sense the environment and start filling the Q-table (line 4 in Algorithm 1), agents try to propose and/or join coalitions as described in the rest of that algorithm. Each coalition proposed is characterized by an action which is then performed by all members belonging to it.

4. EXPERIMENTS AND RESULTS

Experiments were performed using a coordination game where each agent has two possible actions; joint actions and their rewards are given by the matrix shown in Table 1.

Algorithm 1 Playing the Coordination Game Within a Coalition

1: **for all** $i \in N$ **do**
2:     initialize Q values, list of neighbors
3:     **while** not time out **do**
4:         **if** $t > t_c$ **then**
5:             **if** i not in coalition and one Q-value $>>$ other Q-value **then**
6:                 **if** some neighbor j already formed coalition to play action corresponding to the higher Q-value **then**
7:                     join this coalition
8:                 **else**
9:                     propose coalition to play action corresponding to the higher Q-value
10:                 **end if**
11:             **end if**
12:         **end if**
13:         **if** i not in coalition **then**
14:             select action a_i according to Boltzmann exploration (tailored as above whether agents play individually or jointly)
15:         **else**
16:             play action played by the coalition
17:         **end if**
18:         individual or joint play with each neighbor j as above
19:         update of the Q-values (tailored as above whether agents play individually or jointly)
20:     **end while**
21: **end for**

Agents are allowed to play this game repeatedly while rewards are recorded. Plots of average reward (over all agents) along time are shown. All experiments were repeated 20 times. To keep the figures more clear error bars are not shown; the standard deviation is 10% at most. Values for the main parameters used in the simulation are (when applicable, same as in [1]): N (number of agents) is 6×6 and 24×24; $\eta = 10$ (reward); $T = 100$ (temperature); T (temperature decay) is $0.95T$; $\alpha = 0.5$ (learning coefficient); $\gamma = 0$ (discount rate); $t_c = 10$ (time before forming coalitions).

The performance of the individual learning scheme can be seen in Figure 1 (dashed line) for a grid of size 6×6. In each simulation step, all agents play the coordination game with all neighbors. The reward of one agent is the average
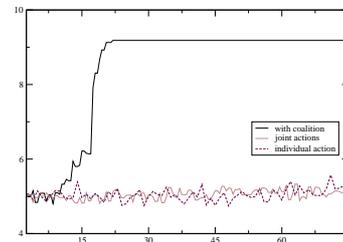

Figure 1: Comparison Reward along Time, Grid 6x6

of the rewards yielded from each play. Thus, ideally, the average reward is $\eta = 10$. As expected the performance of the individual learning is poor as agents get a reward of either $\eta = 10$ or $\eta = 0$ leading them to prefer one choice of action over the other. Because these preferences are not necessarily coordinated towards the same action, on average, the reward is only slightly superior to 5. In the case of the grid scenario each agent plays with more than one opponent. The more opponents, the less likely it is that they all play coordinated actions. Using joint learning yields a performance that is as poor as the individual learning as seen in Figures 1. It must be emphasized that joint action learning as proposed in [1] is not feasible if all 36 agents learn jointly as the size of the Q tables is 2^{36} each.

Simulations in which agents act in coalitions follow Algorithm 1. After t_c time steps which are used to explore the environment (learning individually) and start filling the Q-table (line 4 in Algorithm 1), agents try to propose and/or join coalitions. Each coalition proposed is characterized by an action which is then performed by all its members. Obviously, not necessarily all coalitions agree to play the same action. For instance, some groups may find action a_0 to be the best while other groups do opt for a_1. This causes agents in the borders between two neighboring coalitions to perform poor as they cannot be in both coalitions at the same time and therefore they cannot play coordinated actions with all neighbors. Hence, the performance of the whole set of agents is not always the best possible.

To test scalability, simulations with bigger grids were also performed. For a grid of size 24×24 the results are shown in Figure 2. The conclusions drawn before remain. Figure 3 depicts the configuration of the coalitions at the end of one simulation. Darker nodes are agents playing a_0. Notice that it is *not* the case that there are only (roughly) four coalitions. Their number is higher; it only happens that several coalitions playing the same action are clustered together, hence having the same color. Also, there is a tendency that the more agents, the higher the number of coalitions so that the loss in reward tends to increase when there are more agents. This can be seen by comparing the reward of coalitions in Figures 1 (grid of size 6) and 2 (grid size 24). When all agents are in coalitions and these all agree to select the same action, then obviously the best reward possible is achieved. However, due to the stochastic nature of the game, this seldom happens. Besides, it might be that an agent cannot decide which coalition to join as it has not converged to any pattern of its Q-values and hence cannot decide which group would be the best given its experience playing the game so far.

5. CONCLUDING REMARKS

Multi-agent reinforcement learning is inherently more complex than single agent reinforcement learning because while one agent is trying to model the environment, other agents are doing the same. For a large number of agents, alternative solutions are necessary. In this paper we propose an approach based on the partitioning of the problem in several, smaller multiagent systems as a compromise between complete distribution and complete centralization. In this approach coalitions are formed to facilitate the issue of jointly acting and learning in a coordination game played by agents located in a grid.

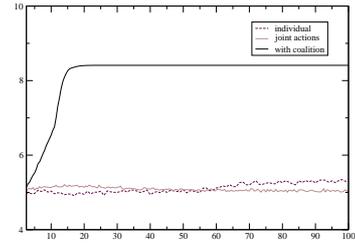One obvious extension is to tackle games with more than

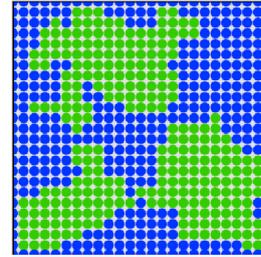**Figure 2: Comparison Reward along Time, Grid 24x24**

Figure 3: Coalitions in a Grid 24x24

one state (e.g. [5]) in which agents have distinct reward functions. If two neighbors are being paid according to different payoff matrices, then the pattern of convergence shown here will change, as well as the coalition structures.

6. REFERENCES

[1] Caroline Claus and Craig Boutilier. The dynamics of reinforcement learning in cooperative multiagent systems. In *Proceedings of the Fifteenth National Conference on Artificial Intelligence*, pages 746–752, 1998.

[2] Junling Hu and Michael P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Proc. 15th International Conf. on Machine Learning*, pages 242–250. Morgan Kaufmann, 1998.

[3] Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Proceedings of the 11th International Conference on Machine Learning, ML*, pages 157–163, New Brunswick, NJ, 1994. Morgan Kaufmann.

[4] Tuomas Sandholm, Kate Larson, Martin Andersson, Onn Shehory, and Fernando Tohmé. Coalition structure generation with worst case guarantees. *Artificial Intelligence*, 111(1–2):209–238, 1999.

[5] Peter Vrancx, Karl Tuyls, and Ronald L. Westra. Switching dynamics of multi-agent learning. In Lin Padgham, David Parkes, J. Müller, and Simon Parsons, editors, *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems*, volume 1, pages 307–313, Estoril, 2008.