An Experimental Study of Different Approaches to Reinforcement Learning in Common Interest Stochastic Games

Avi Bab and Ronen Brafman

Ben-Gurion University, Beer-Sheva 84105, Israel

Abstract. Stochastic (a.k.a. Markov) Games pose many unsolved problems in Game Theory. One class of stochastic games that is better understood is that of Common Interest Stochastic Games (CISG). CISGs form an interesting class of multi-agent settings where the distributed nature of the systems, rather than adverserial behavior, is the main challenge to efficient learning. In this paper we examine three different approaches to RL in CISGs, embedded in the FriendQ, OAL, and Rmax algorithms. We show the performance of the above algorithms on some non-trivial games that illustrate the advantages and disadvantages of the different approaches.

1 Introduction

Learning in Common Interest Stochastic Games (CISGs) provides an interesting intermediate ground between single-agent reinforcement learning (RL) [1] and general multi-agent RL (MARL). CISGs pose all the standard challenges of single-agent RL, in particular the need to balance exploration and exploitation [2] and to exhaust information (i.e propagate new experience) [1,3]. In addition, they challenge the agents to coordinate behavior, without confronting the more difficult task of optimizing behavior against an adversary [4].

CISGs require inter-agent coordination at two levels: (i) selecting whether to explore or exploit in unison; (ii) coordinating the exploration and exploitation moves. This requirement stems from the dependence of the team's next state on the actions of *all* its members. Hence, it is impossible for the team to explore (or exploit) unless all agents explore/exploit together. Moreover, even when the model is known, multiple Nash equilibria are likely to exist, and the agents still face the task of reaching consensus on which specific Nash equilibrium to play.

We compare three algorithms for learning in CISGs: OAL [5], FriendQ [6], and Rmax [7]. They were selected because each embodies a different approach to these learning tasks, while guaranteeing convergence to optimal behavior in CISGs. We examine diverse variants of these algorithms with the aim of gaining better understanding of their performance w.r.t. their approach to exploration-exploitation, information exhaustion, and coordination tasks.

The main phenomenon demonstrated by our experiments is the high sensitivity of FriendQ and OAL to the topology and dynamics of the environment vs. Rmax's stability. Even moderate-sized physical domains lead to numerous

J.-F. Boulicaut et al. (Eds.): ECML 2004, LNAI 3201, pp. 75–86, 2004.

Nash-equilibria, where Rmax's exploration bias, efficient implicit model-based coordination and information exhaustion lead to superior performance.

The paper is structured as follows: Section 2 provides the necessary background on multi-agent reinforcement learning. Section 3 describes the algorithms involved. Section 4 describes our experiments, and Section 5 concludes the paper.

2 Multi-agent Reinforcement Learning

In MARL, the environment is modeled as a Stochastic Game(SG). An SG is a tuple $\langle P, S, A, R, T\rangle$ where: P is a set of n agents; S, a state space; $A = \times A_{i=1\ldots n}$, a set of joint-actions, where A_i is the set of private-actions available to agent i; $R = \{R_1, R_2, \ldots, R_n\}$ is a set of reward functions for the agents where $R_i : S \times A \to \mathbb{R}$ is the reward function for agent i and $R_i(s, a)$ is the expected reward for agent i when joint action $a \in A$ is performed in state $s \in S$; and $T : S \times A \times S \to [0, 1]$ is a stochastic transition function where $T(s, a, s')$ is the probability of reaching state s' when joint action a is played in state s. In CISGs, all agents have identical interests, i.e., $R_i = R_j\ \forall i, j \in P$.

Each agent in an SG attempts to maximize its expected value, i.e., its expected sum of discounted rewards: $E[\sum_{t=0}^{\infty} \gamma^t (r_{i,t})]$ where $r_{i,t}$ is agent i's reward at time t, and $\gamma \in (0, 1)$ is a discount factor. In CISGs, this means that agents attempt to maximize their common expected value. Thus, CISGs can model RL by a distributed team of agents.

A deterministic *joint policy* $\pi = \{\pi_1 \ldots \pi_n\}$ is a mapping from states to joint actions where $\pi_i(s)$ is agent i's part of the joint action $\pi(s) = (\pi_1(s), \ldots, \pi_n(s))$. Every CISG has at least one deterministic policy π^* that achieves optimal value. Any such policy π^* satisfies $\pi^*(s) = argmax_a Q^*(s, a)$. $Q^*(s, a)$ is called the Q value of s and a. It is the expected sum of rewards received for taking action a in state s and behaving optimally thereafter, i.e.,: $Q^*(s, a) = R(s, a) + \gamma \sum_{s' \in S} T(s, a, s') \max_{a' \in A} Q^*(s', a')$.

A joint policy $\{\pi_1 ... \pi_n\}$ is a *Nash equilibrium* if each individual policy π_i is a best response to the others. That is $V(s, \{\pi_1 ... \pi_i ... \pi_n\}) \geq V(s, \{\pi_1 ... \pi'_i ... \pi_n\})$ for all $i \in P$, $s \in S$, and individual policy $\pi'_i \neq \pi_i$, and where $V(s, \pi)$ is the expected return when joint policy π is played starting from state s. An *optimal Nash equilibrium* is a Nash equilibrium that maximizes the expected return.

We shall concentrate on CISG learning under *perfect monitoring*, where each agent sees the actions of all agents. OAL and FriendQ rely on this assumption. Rmax can handle scenarios in which each agent sees its own private action only.

3 The Learning Algorithms

Next we describe the three different reinforcement learning algorithms we studied.

3.1 FriendQ

FriendQ [6] extends basic Q-learning into CISGs. It attempts to learn Q-values without maintaining a model of the environment. After taking a joint action

$a = (a_1, ..., a_n)$ in state s at time t and reaching state s' with reward r_{imm} the agents update the Q-value of $\langle s, a \rangle$ as follows:

$$Q_t(s, a_1, \ldots, a_n) \leftarrow (1 - \alpha_t) Q_{t-1}(s, a_1, \ldots, a_n) + \\ + \alpha_t (r_{imm} + \gamma \max_{b_1, \ldots, b_n} Q(s', b_1, \ldots, b_n))$$

where $\alpha_t \in (0, 1]$ is a learning rate parameter and γ the discount factor. Given that $\sum_{t=0}^{\infty} \alpha_t = \infty$, $\sum_{t=0}^{\infty} \alpha_t^2 < \infty$ and that every joint action is performed infinitely often in every state, the Q-values are guaranteed to converge asymptotically to Q^* [8]. In online learning, convergence to optimal behavior is considered. This is achieved using "Greedy in the Limit with Infinite Exploration"(GLIE) learning policies. In GLIE every state-action pair is visited infinitely often, and in the limit, the action selection is greedy w.r.t. the Q-values w.p.1. Two common GLIE policies are Boltzman distributed action selection and ϵ-greedy action selection. FriendQ lacks a measure for private actions, required for Boltzman exploration, so we use it with ϵ-greedy exploration only. In ϵ-greedy exploration, each agent randomly picks an exploratory private-action with probability ϵ, and with probability $1-\epsilon$ takes it's part of an optimal (greedy) joint-action w.r.t. the current Q-value. ϵ is asymptotically decreased to zero over time.

All agents make the same observations, so they maintain identical Q-values. But two problems arise: (i) Because randomization is used to select exploration *or* exploitation, the agents cannot coordinate their choice of when and what to explore. (ii) In case of multiple optimal policies, i.e., several joint actions with maximal Q values in a certain state, the agents must agree on one such action.The original FriendQ algorithm has no explicit mechanism for handling these issues. Here, we examined some enhanced versions of FriendQ: We begin with Uncoordinated FriendQ (UFQ), the simple version described above. We then examine the effect of adding coordination of greedy joint actions by utilizing techniques introduced in [9], basically, a shared order over joint actions is used for selecting among equivalent Nash equilibria. If such an order is not built into the agents, it is established during a preliminary phase (see [9] for more details). This version is referred to as Coordinated FriendQ (CFQ). For comparison we also combine this equilibrium selection technique with the model based Q-learning algorithm used by OAL. We call this combination ModelQ (MQ). We continue with Deterministic FriendQ (DFQ) in which the agents explore and exploit in unison, always exploring the least tried joint action. An exploratory action is taken each $\lfloor 1/\epsilon \rfloor' th$ move. Finally we add Eligibility Traces [1] to DFQ (ETDFQ).

3.2 OAL

OAL [5] combines classic model-based Q-learning with a new fictitious play algorithm for action and equilibrium selection named BAP (Biased Adaptive Play). The Q-value update rule is:

$$Q_{t+1}(s, a) = R_t(s, a) + \gamma \sum_{s'} T_t(s, a, s') \max_{a'} Q_t(s', a')$$

where R_t, the approximated mean reward, and T_t, the approximated transition probability are estimated using the statistics gathered up to time t. Using GLIE policy, OAL converges in the limit to optimal behavior.

An OAL agent records the last m joint-actions taken at each state. Each round, the agent randomly samples k joint-actions from the last m joint-actions ($k < m$) taken at the current state. Greedy private-action selection is done using BAP. BAP treats the stage-games in the CISG (the single-state games induced by the Q-values at each state). At each round BAP builds and plays a Virtual Game (VG) for the current stage-game. The VG has reward 1 for any optimal equilibrium and reward 0 for all other joint actions. Unless equilibrium-selection conditions are met, BAP chooses the best response private-action according to the current model and w.r.t. the current history sample, i.e., private-action that maximizes expected payoff assuming the history sample represents the other agents strategies[1]. If equilibrium selection conditions are met – i.e., if in all k current samples the other agents chose the same part of some optimal equilibrium (w.r.t. the current VG) and if, in at least one of the k samples, a complete optimal joint action was played – the agent plays its part of the last such action in the sample. (Recall that different agents may sample different past plays).

OAL can use Boltzman distributed exploration. In Boltzman exploration a private action $a_i \in A_i$ is taken with probability $\frac{e^{ER(a_i)/\tau}}{\sum_{a' \in A_i} e^{ER(a')/\tau}}$ where $ER(a_i)$ is the expected payoff of private-action a_i. τ, the temperature, is a positive parameter decreased over time asymptotically to zero. High temperatures cause the actions to be all nearly equi-probable and when $\tau \to 0$ action selection becomes completely greedy. We also examine OAL with an addition of Prioritized Sweeping [3] to the underlying Q-learning algorithm (PSOAL).

3.3 Rmax

Rmax [7] is a model-based algorithm designed to handle learning in MDPs and in zero-sum stochastic games. The agent maintains a model of the environment, initialized in a particular optimistic manner. It always behaves optimally w.r.t. its current model, while updating this model (and hence its behavior) when new observations are made. Because the initialization and model-update steps are deterministic, so is the whole algorithm. Recall that a CISG can be viewed as an MDP controlled by a distributed team. [9] observes that such a team can coordinate its behavior given a deterministic algorithm such as Rmax. At each point in time, all agents have an identical model of the environment and know what joint-action needs to be executed next[2]. Thus, each agent plays its part of this action. When a number of actions are optimal w.r.t. the current state, the agents utilize a shared order over joint-actions to select among these actions. [9] shows how such an order can be set-up, how even weaker coordination devices that do not require such an order can be used, and how these ideas can be employed even under imperfect monitoring.

The model M' used by Rmax consists of $n+1$ states $S' = \{s_0, ..., s_n\}$ where $s_1, ..., s_n$ correspond to the real states and s_0 is a fictitious state[3]. The transition

[1] This is known as fictitious play [10].

[2] Knowledge of fellow agents (private) action sets can be acquired online.

[3] The model may be constructed online as states are discovered.

probabilities in M' are initialized to $T_{M'}(s,a,s_0) = 1\ \forall \langle s,a \rangle \in S' \times A$. The reward function is initialized to $R_{M'}(s,a) = R_{max}\ \forall \langle s,a \rangle \in S' \times A$, where R_{max} is an upper bound on $max_{s \in S, a \in A} R(s,a)$. Each state/joint-action pair in M' is classified either as *known* or as *unknown*. Initially, all entries are unknown.

Rmax computes an optimal policy with respect to M' and follows this policy until some entry becomes known. It keeps the following records: (i) number of times each action was taken at each state and the resulting state. (ii) the actual rewards r_{ac} received at each entry. An entry (s,a) becomes known after it has been sampled $K1$ times, such that with high probability $T_M(s,a,s') - \rho \leq PE(s,a,s'|K1) \leq T_M(s,a,s') + \rho$ where T_M is the transition function in M, $PE(s,a,\cdot\,|K1)$ the empirical transition probability according to the $K1$ samples, and ρ the accuracy required from M'. When (s,a) becomes known the following updates are made: $T_{M'}(s,a,\cdot) \leftarrow PE(s,a,\cdot\,|K1)$ and $R_{M'}(s,a) \leftarrow r_{ac}(s,a)$. Once a new entry becomes known, Rmax computes the new deterministic optimal policy w.r.t. the updated model M' and follows it.

The known (worst-case) polynomial bounds on $K1$ are impractical. In the experiments we violated these bounds. This enables to eliminate knowledge of the state space size. We also do not assume R_{max} is known, instead we initialize R_{max} to some positive value and update it online to be twice the highest reward encountered so far.

3.4 Discussion of Algorithms

Efficiency of GLIE learning policies depends on the topology and dynamics of the environment. If the probability to explore falls low before "profitable" parts of the environment are sufficiently sampled, the increasing bias to exploit may keep the agents in sub-optimal states. This means that GLIE policies can exhibit significant differences depending on the particular schedule of exploration. GLIE policies also suffer from their inability to completely stop exploration at some point. Thus, even when greedy behavior is optimal, the agent is unable to attain optimal return. The exploration method of Rmax is less susceptive to the structure of the environment. As long as Rmax cannot achieve actual reward ϵ-close to optimal it will have a strong bias for exploration since unknown entries seem very attractive. This strategy is profitable when the model can be learned in a short time. However, the theoretical worst-case bounds for convergence in Rmax [7] are impractical. In practice, much lower values of $K1$ suffice. Bayesian exploration [11,12] and locality considerations might help to obtain better adaptive bounds, but we did not pursue these approaches here.

GLIE makes learning "slower" as the agents get "older". To accelerate learning one must use new experience in an exhaustive manner (i.e, use current experience to improve behavior in previously visited states). Eligibility traces are used to propagate information in model-free algorithms. In model-based algorithms an exhaustive computation per new experience is too expensive (in cpu time) and Prioritized Sweeping [3] is a well known solution, yet not exhaustive. Rmax makes one exhaustive computation each time a new entry becomes known (and does no further computation).

Exploration in FriendQ and OAL algorithms is not coordinated. Each of the agents independently chooses an exploratory action with some diminishing probability. Thus, joint-actions that have no element (private-action) of some optimal joint-action have a lower chance of being explored. Hence, some popular techniques for decreasing exploration in the single agent case lead to finite exploration in the multi agent case. For example taking $\epsilon = 1/time$ for ϵ-greedy policies will make the chance of exploring such joint actions $1/time^n$, where n is the number of agents.

Equilibrium selection in Rmax and CFQ comes with no cost. In OAL it is essentially a random protocol for achieving consensus. This protocol may take long to reach consensus w.r.t. the current Q-values, but provides for another exploration mechanism at early stages, when Q-values are frequently updated.

Parameter tuning is task specific and based more on intuition and trial and error than on theoretical results. FriendQ has a range of parameters for decaying the learning rate, the exploration probability and the eligibility traces which also pose inter-parameter dependencies. For decreasing the learning rate parameter we used the results presented in [13]. OAL takes parameters for history sample size and for exploration. In this respect, we found Rmax superior. It has a single and very intuitive parameter - number of visits to declare an entry *known*.

4 Experimental Results and Analysis

We exhibit results on two 2-agent grid games. The games were designed to evaluate the effects of exploration, coordination and information exhaustion methods on performance in different environments. The game environments are grids in which the agents can move using the actions *up, down, left, right, stand*. Agent position pairs constitute the state space of the underlying stochastic game.

The games were played under deterministic and stochastic transition probabilities. In stochastic mode, each action (excluding *stand*) succeeds with probability 0.6. With probability 0.1 the agent is moved to an adjacent cell or stays in place. *stand* succeeds w.p.1.

We tested the algorithms on the first set of experimental conditions (see below Game1 with deterministic setup) with a range of different parameters. The parameters that achieved best results were then used throughout:

FriendQ

Exploration: ϵ-greedy with – (i) $\epsilon_t \leftarrow 1/count_t^{0.5000001}$ where $count_t$ is the number of exploratory steps taken by time t. (ii) $\epsilon_t \leftarrow 0.99998^{count_t}$. (unless specified otherwise, (i) is used)[4]. **Learning rate:** $\alpha_{s,a} \leftarrow 1/n(s,a)^{0.5000001}$ where $n(s,a)$ is the number of times action a was taken in state s.

OAL

Exploration: For ϵ-greedy, $\epsilon_t \leftarrow 1/count_t^{0.5000001}$, as in FriendQ. For Boltzman exploration the temperature parameter was decreased by $\tau \leftarrow 100/count^{0.7}$ **History:** Random history sample size = 5. History memory size $m = 20$ (m must

[4] Exponential decay of ϵ violates the "infinite exploration" condition for convergence.

satisfy $m \geq k \times (number\ of\ agents + 2)$.) We refer to OAL with ϵ-greedy exploration as ϵ-OAL, and to OAL with Boltzman exploration as B-OAL.

Rmax

Sampling: values of 50, 100, 200 for K1 (visits to mark an entry known) were tested. **Accuracy of policy iteration:** Offline policy iterations was halted when the difference between two successive approximations was less than 0.001

Each set of experimental conditions, other than those related to Rmax, was subjected to 100 repeated trials. For Rmax, 20 trials were done using K1=50, 40 with K1=100 and 40 with K1=200. The discount factor was 0.98 in all trials.

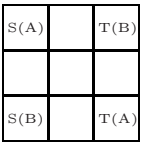


Fig. 1. Game 1 - Initial and Goal states.

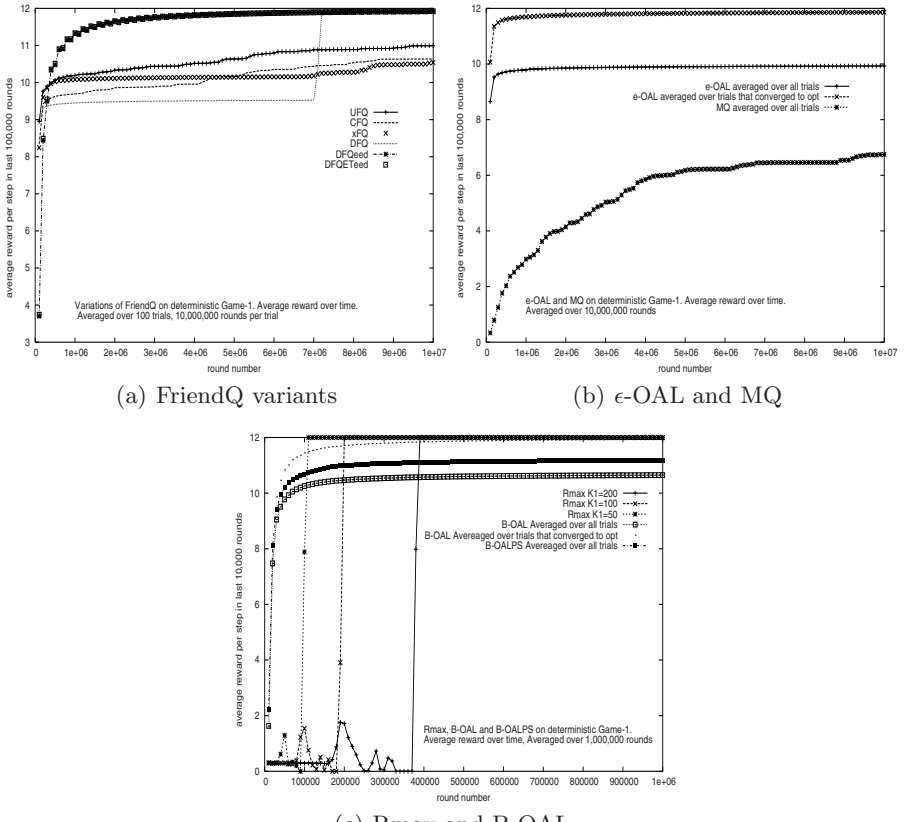
4.1 Game 1

This game was devised to emphasize the effects of equilibrium selection methods. It has a single goal state (the only reward yielding state) and several optimal ways of reaching it. The game is depicted in Figure 1. $S(X)$ and $T(X)$ are the respective initial and goal positions of agent X. The underlying SG has 71 states. The goal state T (both agents in goal positions) generates a reward of 48. Upon reaching the goal the agents are reset to their initial position. The optimal behavior is to reach T in four steps. Under deterministic setup, this yields an average reward per step of 12. There are 11 optimal different equilibria. The optimal policies are the same under the stochastic setting, with $\sim$6.14 average reward per step. Algorithms were executed for 10^7 rounds on both settings. For the deterministic setup Table 1 classifies the number of trials (of 100 in total) according to the algorithms and learned policies. Here xFQ is a variant of FriendQ in which the agents explore in unison but do not coordinate exploratory actions. The suffix "ϵed" of DFQϵed denotes exponential decay of ϵ. In the present context, the agents' learning of an optimal policy means that their greedy choice of actions is optimal. Because of continued exploration, this does not

Table 1. Game1 – Number of Trials Per Learned Policies under Deterministic Setup.

steps to goal	UFQ	CFQ	xFQ	DFQ	DFQϵed	ϵ-OAL	B-OAL	B-OALPS	MQ	Rmax
4	62	49	47	100	100	26	49	41/60	1	100
5	38	49	46			62	51	19/60	49	
6+		2	7			12			29	
∞									21	

necessarily yield optimal behavior. Figure 2 presents the average reward obtained by the agents over time under deterministic setup.



(a) FriendQ variants

(b) ϵ-OAL and MQ

(c) Rmax and B-OAL

Fig. 2. Game 1 – Avg. Reward under Deterministic Setup. Graphs for Rmax and B-OAL are over 10^6 Rounds Only.

FriendQ converges quickly to second-best behavior (Fig. 2a). From that point on, the average learning curve of UFQ, CFQ and xFQ increases stepwise rather than continuously. This results from a sudden switch of the FriendQ agents from sub-optimal to optimal behavior upon restructuring of the Q-values series. We would expect ϵ-OAL to present a similar trend but when OAL converges to the second-best behavior in the first 5×10^5 rounds, it fails to order the Q-values properly even after 10^7 rounds. In DFQ since exploration is deterministic this switch is always after 7×10^6 rounds(Fig. 2a).

Surprisingly, UFQ fares better than CFQ (Fig. 2a). At an early learning stage dis-coordination leads to exploration. We discovered that, later on, the estimated Q-values of optimal actions are rarely equal, and thus, coordinating exploitation

does not pose a problem (at the examined time interval). Exponential decay of ϵ supplies more exploration at an early period then polynomial decay leading to faster convergence of DFQϵed(Fig. 2a). The eligibility traces do not contribute much in this example. We found the parameters of eligibility traces very hard to tune and very sensitive to change in other parameters or environment dynamics.

OAL agents converge relatively quickly to optimal or second-best behavior, and from that time onwards stick to their behavior (Figs 2b,c). B-OAL converges faster and more often to optimal than ϵ-OAL. This stems from more exploration supplied by the Boltzman method than by the ϵ-greedy method in early period of learning. Later on, ϵ-greedy maintains a low exploration probability that decays very slowly while Boltzman exploration drops faster to zero. Thus, when ϵ-OAL learns optimal behavior it keeps achieving only near-optimal average-reward. As expected Prioritized Sweeping improves the performance of B-OALPS.

The performance of ModelQ is inferior to that of OAL (Fig. 2b) presumably because ModelQ does not explore as much as OAL: at early stages of learning fictitious play provides OAL with other means of exploration. When the agents make many stochastic action choices in early stages of learning, fictitious play amplifies the random behavior. However, at later stages of learning, deviation from constant action choice is rare and will probably not affect fictitious play.

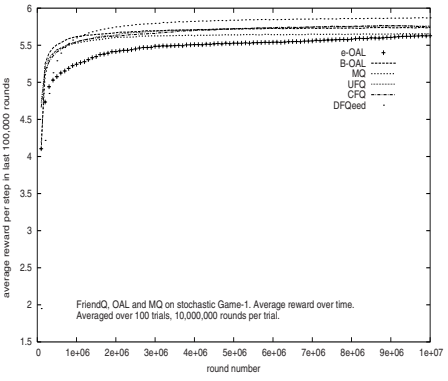
The results on the stochastic setup are presented in Figure 3. By contrast to the deterministic case, MQ performs better than ϵ-OAL. This improvement is attributable to additional exploration stemming from the stochastic nature of the environment. For the same reason CFQ performs the same as UFQ. The slower climb of DFQϵed in relation to U/CFQ at first 10^6 rounds is because the additional early exploration supplied by ϵed is redundant in the stochastic case. The slightly higher return gained by DFQϵed later on is due to the faster decay of ϵ. Rmax behaves similarly in the stochastic and deterministic setups. While the other algorithms achieve only near-optimal return Rmax spends the same amount of time to discover all entries and then attains optimal return.

Rmax's strong exploration bias results in low return until model entries are known (Fig. 3c). From that point on, Rmax attains an optimal return. Very low K1 values, which mean rough transition probability estimates, are enough for computing optimal behavior.

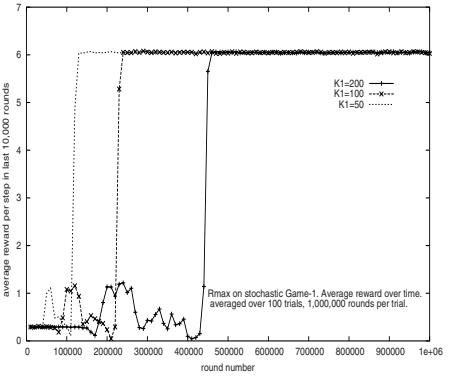
The phenomena observed in Game 1 repeated in Game 2. Therefore, in the following we discuss only phenomena not observed in Game 1.

4.2 Game 2

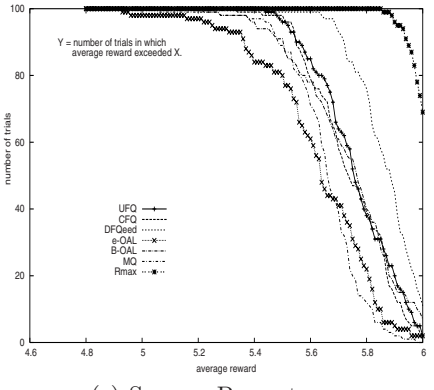
This game was designed to minimize the effects of equilibrium selection and to show how GLIE policies may keep agents exploiting suboptimal possibilities and the importance of coordinated exploration. The game has four goal states and one optimal equilibrium. The game is depicted in Figure 4(a). It consists of an additional element, an object that can be moved by the agents. Agents can push the object by standing to its right(left) and moving left(right) and pull the object by standing to its right(left) and moving right(left). However, the object is too



(a) FriendQ; OAL; MQ



(b) Rmax



(c) Success Percentage

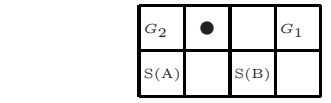
Fig. 3. Game1 – Avg. Reward and Success Percentage under Stochastic Setup.

heavy for one agent and requires cooperation of the two agents to be moved. The manner by which the object is moved is depicted in figure 4(b).

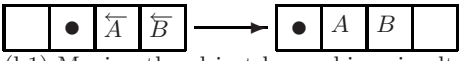
The agents' goal is to move the object into one of the upper corners of the grid, at which point the game is reset to its initial state. Moving the object to the upper right (G_1) or left (G_2) corner yields a reward of 80 and 27, respectively. The optimal behavior is to move the object to G_1 in 8 steps. The average reward per step of an optimal strategy under deterministic setup is 10, and the discounted return is ~ 465. The "second best" strategy is moving the object to G_2 in 3 steps, with an average reward per step of 9 and discounted return of ~ 440. The optimal and second best strategies are the same under stochastic setup. The average reward per step of the optimal policy is ~ 4.8. The underlying CISG contains 164 states. Algorithms were executed for 3×10^7 rounds.

Table 2 classifies the number of trials (of 100 in total) according to the algorithms and learned policies under the deterministic setup.

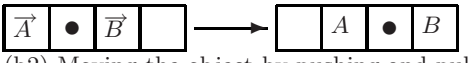
Figure 5 shows the average reward over time obtained by the different algorithms under the deterministic setup. The results are averaged over all trials.



(a) Initial state and Goal states.



(b1) Moving the object by pushing simultaniosly. (Agents' order does not matter).

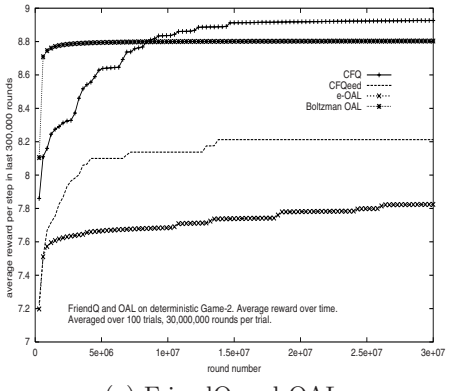


(b2) Moving the object by pushing and pulling simultaniosly. (Agents' order does not matter).

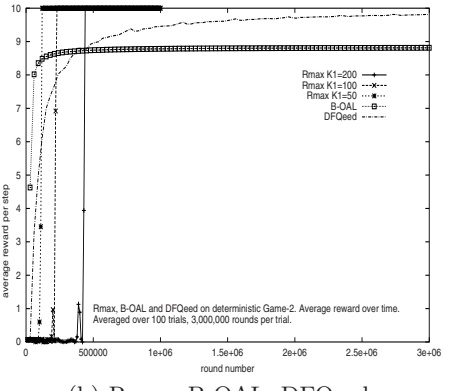
Fig. 4. Game 2.

Table 2. Game2 – Number of Trials Per Learned Policies under Deterministic Setup.

Goal steps to goal	CFQ	CFQϵed	DFQϵed	ϵ-OAL	B-OAL	Rmax
G_1 8			100		1	100
G_2 3	99	39/60		54	91	
G_2 4	1	21/60		46	8	



(a) FriendQ and OAL



(b) Rmax; B-OAL; DFQeed

Fig. 5. Game 2 – Avg. Reward under Deterministic Setup. Rmax and Boltzman OAL are presented on a scale of 3×10^6 rounds.

The main reasons for the poor performance of OAL and CFQ in this game are (i) random exploration has a greater chance of reaching G_2 than G_1. Discovering G_2 before G_1 further reduces the chance of visiting G_1 because of the increasing bias towards exploitation. (ii) exploration of the CFQ and OAL agents is not coordinated. If reaching G_2 is the current greedy policy then G_1 will not be visited unless both agents explore simultaneously. Fig. 5a shows that CFQ does

better with polynomial decay of ϵ then with exponential decay. This stems from finite exploration supplied by exponential decay. However, this finite amount of exploration is sufficient given coordinated exploration as shown by Fig. 5b.

5 Summary

We presented an experimental study of three fundamentally different algorithms for learning in CISGs. Our results illustrate the strength and weaknesses of different aspects of these algorithms in different settings, highlighting the accentuated importance of effective exploration in this class of games, the importance of coordinated exploration, the confidence gained by deterministic behavior and the benefits of exhaustion of information. For lack of space we did not exhibit or discuss situations in which it is impractical or inefficient to explore the whole state space. Such situations may occur when optimum can be gained in any local subset of states or when the state space is large in relation to agents lifetime.

A longer version of this paper containing additional results and analysis is available from the authors.

References

1. Sutton, R.S., Barto, A.G.: Reinforcement Learning: An Introduction. MIT Press (1998)
2. Kaelbling, L.P., Littman, M.L., Moore, A.W.: Reinforcement learning: A survey. JAIR **4** (1996) 237–285
3. Moore, A.W., Atkeson, C.G.: Prioritized sweeping: Reinforcement learning with less data and less time. Machine Learning **13** (1993) 103–130
4. Claus, C., Boutilier, C.: The dynamics of reinforcement learning in cooperative multi-agent systems. In: Proc. Workshop on Multi-Agent Learning. (1997) 602–608
5. Wang, X., Sandholm, T.: Reinforcement learning to play an optimal nash equilibrium in team markov games. In: NIPS'02. (2002)
6. Littman, M.L.: Friend-or-foe q-learning in general-sum games. In: Proc. ICML'01. (2001)
7. Brafman, R.I., Tennenholtz, M.: R-max – a general polynomial time algorithm for near-optimal reinforcement learning. JMLR **3** (2002) 213–231
8. Bertsekas, D., Tsitsiklis, J.: Neuri-Dynamic Programming. Athena Scientific (1996)
9. Brafman, R.I., Tennenholtz, M.: Learning to coordinate efficiently: A model based approach. JAIR **19** (2003) 11–23
10. Fudenberg, D., Levine, D.: The theory of learning in games. MIT Press (1998)
11. Dearden, R., Friedman, N., Andre, D.: Model based bayesian exploration. In: UAI'99. (1999)
12. Chalkiadakis, G., Boutilier, C.: Coordination in multiagent reinforcement learning: A bayesian approach. In: AAMAS'03. (2003)
13. Even-Dar, E., Mansour, Y.: Learning rates for Q-learning. In: COLT'01. (2001)