Evaluating concurrent reinforcement learners

Manisha Mundhe & Sandip Sen
Department of Mathematical & Computer Sciences
University of Tulsa
600 South College Avenue, Tulsa,OK 74104
phone:918-631-2985, FAX:918-631-3077, e-mail:sandip-sen@utulsa.edu

Abstract

Assumptions underlying the convergence proofs of Reinforcement learning (RL) algorithms like Q-learning are violated when multiple interacting agents adapt their strategies on-line as a result of learning. Empirical investigations in several domains, however, have produced encouraging results. We systematically evaluate the convergence behavior of concurrent reinforcement learning agents using game matrices of varying complexity as studied by Claus and Boutilier Claus & Boutilier [?]. Variants of simple RL algorithms are evaluated for convergence under relative prominence of global optima, feedback noise, scale up of game matrix size, and game matrix characteristics. Our results show surprising departures from that observed by Claus and Boutilier, particular for larger problem sizes. We present an analysis and explanation of the experimental results that provides insight into the nature of convergence of these concurrent learners. We identify the variants of the algorithms studied as recursive modeling agents. This allows us to suggest more effective learning agents with deeper levels of nesting of beliefs about other agents. We also discuss the effect of greedy and non-greedy strategies on the modeling agents. Our results show that the greedy strategy turns out to be better for higher level modeling agents.

1 Introduction

Whereas previous research on developing agent coordination mechanisms focused on off-line design of agent organizations, behavioral rules, negotiation protocols, etc., it was recognized that agents operating in open, dynamic environments must be able to flexibly adapt to changing demands and opportunities [?]. In particular, individual agents are forced to engage with other agents which have varying goals, abilities, composition, and lifespan. To effectively utilize opportunities presented and avoid pitfalls, agents need to learn about other agents and adapt local behavior based on group composition and dynamics. For machine learning researchers, multiagent learning problems are challenging, because they violate the stationary environment assumptions used by most machine learning systems. As multiple agents learn simultaneously in a tightly coupled system, the feedback received by the same agent for the same action varies considerably. The stationary environment assumption used by most current machine learning systems do not hold for such rapidly changing environments.

In this paper, we evaluate the effectiveness of standard reinforcement learning techniques for learning coordination strategies from feedback received due to repeated interaction with other agents in the environment. The attractiveness of RL algorithms stem from the fact that little prior knowledge of the domain or of the decision mechanisms adopted by the other agent are required. For example, assumptions regarding whether other agents are friendly or hostile are not necessary for the use of RL techniques. As suggested above, no formal guarantees exist for the convergence of concurrent reinforcement learners to mutually coherent policies. Researchers, however, have demonstrated partially successful applications of standard RL methods for a number of synthetic multiagent environments [?; ?; ?; ?].

Claus and Boutilier [?] have used repeated two-agent single stage games as the environment to demonstrate the relative efficiency of two concurrent RL agents. This paper is largely motivated by this latter body of work. In particular, our work begins by carefully evaluating RL algorithms used by Claus and Boutilier. The type of games that we study, however, is more varied than those studied by them and helps to better identify the strengths and weaknesses of these learning schemes. We are also able to identify critical problems with these algorithms, which in turn, allows us to develop more appropriate concurrent learning schemes with significantly improved performance over a wide variety of game types.

2 Reinforcement Learning

In reinforcement learning problems [?] reactive and adaptive agents are given a description of the current state and have to choose the next action from a set of possible actions so as to maximize a scalar *reinforcement* or *feedback* received after each action. The learner's environment can be modeled by a discrete time, finite state, Markov decision process that can be represented by a 4-tuple $\langle S, A, P, r\rangle$ where $P : S \times S \times A \mapsto [0,1]$ gives the probability of moving from state s_1 to s_2 on performing action a, and $r : S \times A \mapsto \Re$ is a scalar reward function. Various reinforcement learning strategies have been proposed using which agents can develop a policy to maximize rewards accumulated over time. For evaluating the classifier system paradigm for multiagent reinforcement learning, we compare it with the Q-learning algorithm, which is designed to find a policy π^* that maximizes reward for all states $s \in S$. The decision policy is represented by a function, $Q : S \times A \mapsto \Re$. If an action a in state s produces a *reinforcement* of R and a transition to state s', then the corresponding Q value is modified as follows:

$$Q(s,a) \leftarrow (1-\beta)\ Q(s,a) + \beta\ (R + \gamma \max_{a' \in A} Q(s',a')).$$

In this paper, we present experiments with stateless domains and immediate feedback. So, Q-values are learned only for actions, and the update rule is simpler (actually the learning methods studied here are much more akin to estimator algorithms [?] but we are following Claus & Boutilier's framework for ease of comparison). While stateless RL is indeed much simpler than state-based RL, it allows for a much more systematic and controlled variation of the environment to evaluate the effectiveness of concurrent reinforcement learners. We will see that even such "simple" domains can be used to obtain interesting insight into the working of concurrent RLs. Our work, as well as that of Claus and Boutilier [?] should, however, be recognized as only the first steps towards building a much needed understanding of concurrent learning agents in more general scenarios.

2.1 Concurrent reinforcement learners

We present four types of concurrent RL algorithm that have been used in our experimentation. The first two of these are based on Claus and Boutilier's experiments [?], while the third one is a modification of the second algorithm: **Individual reinforcement learners (Is):** These agents simply use the following update rule to update their Q-values: $Q(a) \rightarrow (1-\alpha) * Q(a) + \alpha * r$, where r is the reward obtained from the game matrix, and α is the learning rate. The exploration scheme used is a biased Boltzmann method, with action a being chosen with the probability

$$\frac{exp(\frac{Q(a)}{T})}{\sum_{i \in \mathcal{A}} exp(\frac{Q(i)}{T})},$$

where $\mathcal{A}$ is the set of actions available to agent A. Vidal and Durfee [?] describes such an agent as a 0-level agent, as it does not maintain any explicit model of other agencies in its environment.

One-level Expected Utility based Probabilistic reinforcement learner (1EUPs): Q-values are calculated as in the case of Is. Expected value of action a is calculated as $E(a) = \sum_{b \in B} Q(a,b)Pr(b)$, where B is the set of actions available to the other agent, and $Pr(b)$ is the probability that the other agent will choose action b. The probabilities are calculated from the observed frequencies with which the other agent has chosen from its set of actions. The expected values for actions, and not the corresponding Q-values, are used to calculate the Boltzmann exploration probabilities[1]. These agents model other agents as 0-level agents. Vidal and Durfee refer to such agents as 1-level agents.

Two-level Expected Utility Maximizing reinforcement learner (2EUMs): A 2EUM models the other agent as a 1EUP. In order to calculate the action probability distribution of this other agent, the 2EUM assumes that the other agent has the same policy matrix as itself. Using its own past history of actions, the 2EUM first calculates the probability of other agent's taking a particular action. Using this probability distribution, it then calculates its own expected utility of taking any action and chooses the action that has the maximum expected utility. These agents model other agents as 1-level agents. Vidal and Durfee refer to such agents as 2-level agents.

For ease of reference, in the rest of the paper, we will refer to the Is, 1EUMs, and 2EUMs as 0, 1, and 2-level learners respectively.

3 Results

We ran a series of experiments on deterministic game matrices of the form shown in Table 1. The game matrix represents the feedback given to both agents when they select any particular action combination. We have also used stochastic games where the matrix elements contain both a mean and a standard deviation, and Gaussian random numbers generated with those means and standard deviations are provided as feedbacks for given action choices.

The goal of our experiments is to evaluate the relative capabilities of the described independent learning strategies to quickly converge to globally optimal action combinations, where globally optimal refers to feedback maximization. If two strategies converge to the same average payoffs, we prefer the strategy which allows agents to reach the corresponding policies with smaller number of interactions. For our experiments we have chosen $\alpha = 0.1$ and have initialized Q-values to 0. Initial value of the Boltzmann exploration temperature parameter, T, was set to 16 and it was discounted in successive iterations by a temperature discounting factor of 0.99.

Our first set of experiments were run on a 2x2 game matrix with $x = n = 1$ and $m = y = 0.75$ 1. The main surprise from these set of experiments was that the 0-level learners converged faster and more successfully than non-myopic 1-level learners. This is in contrast to the results reported in Claus and Boutilier's [?] work.

[1] Claus and Boutilier [?] call these agents "Joint action learners" or JAL agents.

	b0	b1
a0	x	y
a1	m	n

Table 1: A 2x2 game matrix with deterministic payoffs.

	b0	b1
a0	$v+\delta$	v
a1	v	$v+\delta$

Table 2: Intermediate learned Q-values in a 2x2 game.

They did not report the α values used or the exact methods of calculating the probabilities. We, however, used the temperature values they used (initial $T = 16$ and discount factor of 0.99) and got better results than they reported with our choice of $\alpha = 0.1$ and frequency-based probability calculation. The performance order difference was produced by the fact that the 0-level learners in our implementation improved more than the 1-level learners as compared to the results reported by Claus & Boutilier [?].

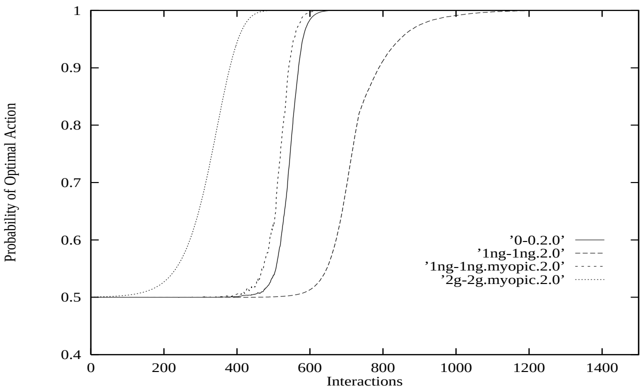


Figure 1: Relative performance of individual I (0-0), 1EUP (1ng-1ng), 1MEUP (1ng-1ng.myopic) and 2EUM (2g-2g) learner pairs where each agent can choose one of two available actions.

The above performance order observation is counter-intuitive as one would think that with more information about action choices of the other agent, agents would learn to coordinate in fewer interactions. But, if the probabilities of other agent's actions are inaccurate, this misleading information can delay convergence. Another reason may be that the 1-level learner assumes that the other agent is a 0-level agent. Since this assumption is wrong, performance suffers. A key reason for the 1-level agent's inefficiency is that it uses all the previous interactions to calculate the probabilities of the other agent taking different actions while deciding its own action. Thus it also takes into account the initial random moves that other agent makes. This itself may significantly delay convergence.

This observation led us to devise a variant of the 1-level agent that we describe below:

One-level Myopic Expected Utility based Probabilistic reinforcement learner (1MEUPs): We believe that as agents learn, their probability distribution of action choices can change quickly. Thus using the entire history of action choices to construct the probability distribution may not produce useful models of opponent behavior. In this form of learner, only the last k (we have used $k = 10$) interactions are used to generate the probability distribution. This, in effect, captures the recent trend in the behavior of the other agent which results in a more up-to-date model. We should also clarify that the 2-level agents in our experiments model the other agent as a myopic 1-level agent.

When we ran experiments with the myopic 1-level agents we found that they not only out-performed the non-myopic 1-level learners, but also converged faster than 0-level learners. Thus a myopic evaluation of other agents' behavior better enables the learning process to keep pace with the adaptation of the other agent. Unless explicitly mentioned otherwise, all 1-level agents used here and the rest of the paper are myopic in nature. Finally, the 2-level learners out-perform all the other strategies we have seen earlier in this section.

3.1 Effects of scale-up of policy space

We varied the number of actions available to each agent from 2 to 5 to 10 to 20. The payoff matrices used had only two possible values: all the diagonal elements had the same optimal value of 1, and the off-diagonal elements had a payoff of 0.75. The results from these experiments are presented in Figure 2.

In general, for the smaller problem sizes, performance of the different agent types are in the following order: 2-2, 1-1, 0-0 (see Figure 1). But as we increase the action space, i.e., allowing more actions per agent, the 1-1 scenarios converge at a suboptimal value, even though, the rate of convergence is still better than the 0-0 agents (see Figure 2). The 0-0 and 2-2 scenarios still converge optimally.

A plausible explanation of the quick convergence in the case of 1-1 scenarios is that there is insufficient exploration of the state space, resulting in premature convergence. Let us consider a sample Q-value matrix for 2 agents with 2 actions per agent. Suppose at some intermediate state, the Q-values learnt are as shown in Table 2. Let p and $1-p$ be the observed probabilities of agent B taking actions $b0$ and $b1$. The expected utilities for agent A's two actions can then be calculated as

$$E(a0) = p(v+\delta) + (1-p)v = p\delta + v,$$

$$E(a1) = pv + (1-p)(v+\delta) = v + (1-p)\delta.$$

Let us suppose $p > 0.5$, i.e., $p > 1-p$. Then, from the above we have $E(a0) > E(a1)$. The probabilities of different action choices for a 1-level learner varies directly as the expected utility of those actions. The higher expected utility for action $a0$, increases the probability of

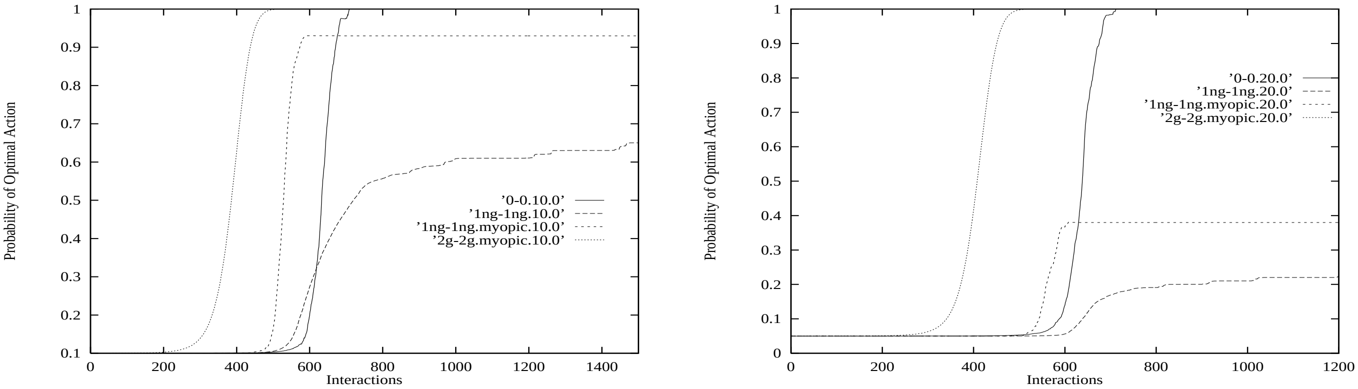


Figure 2: Relative performance of individual I (0-0), 1EUP (1ng-1ng), 1MEUP (1ng-1ng.myopic) and 2EUM (2g-2g) learner pairs with scale up of problem size. The numbers between the periods refer to the number of actions available to an agent.

agent A taking action $a0$, and this probability increases with the frequency with which A have seen B taking action $b0$. Therefore, with a larger value of p, the expected utility of action $a0$ for agent A increases, resulting in $a0$ being chosen more frequently. This makes the 1-level agent more exploitative in nature. A similar analysis applies for agent $B$and results in its choice of action $b0$ if it sees A choosing $a0$ more often. This means that there is mutual reinforcement for exploiting small differences in learned policy values. For a 0-level agent the utilities of each action are difficult to calculate for the same scenario as the Q-values will evolve differently. But in the absence of the additional multiplicative probability factors, the two actions are likely to be closer in estimated utility compared to the 1-level agents. This means that the 0-level agent would continue to explore longer than the 1-level agents.

The mutual bias in the case of 1-1 agents allow for their relatively quick convergence to the global optimum when initial samplings produce higher Q-values for optimal action combinations. The downside of this bias is the convergence to suboptimal choices when these policy elements have higher estimates from initial samplings.

As problem size increases, each action gets fewer evaluations during the initial explorative phase. As such, there is a greater chance that some suboptimal action combination will be estimated to have a higher payoff than the optimal combinations. A plausible fix to this problem would be to extend the initial exploration phase should be extended by choosing an appropriate temperature schedule as problem sizes increases, e.g., by increasing the initial temperature or the discounting factor in the Boltzmann exploration mechanism. This fix, unfortunately, will also cause the learning curve of 1-1 scenarios to rise slowly and it is not clear a priori if the 1-1 scenarios will then converge any faster than the 0-0 scenarios.

The premature convergence of 1-1 scenarios can be avoided if the Q-values represent the actual pay-off structure before the agents start exploiting. This can also be achieved by increasing the learning rate (the α value in the Q-value update rule). The higher learning rate enables actual pay-off values to be estimated with less number of interactions. This is particularly true in the absence of noisy payoff. In such cases, a shorter exploratory phase may be sufficient to learn accurate estimates of action combinations and as such will produce convergence to optimal action combinations. For a matrix size of 10 and $\alpha = 0.25$ the agents converge optimally. But this α value is not sufficient for optimal convergence for much larger problems. As α values have to be continually increased for larger problem sizes, such schemes will be ineffective in noisy domains, as larger values of α will produce wide fluctuation in policy values over time and can actually slow down convergence. A preferred approach to attacking the premature convergence problem might be to combine the above two approaches, i.e., to use higher α values and longer exploratory phase as problem size increases.

We also considered the two agents using different strategies, e.g., one of the agents being 1-level whereas the other being a 0-level agent (we will refer to this situation as a 1-0 scenario), and so on. We conducted experiments with $m = 0.75$ in 2-1, 1-1, 1-0 scenarios and observed that the 2-1 scenarios worked better than the 1-0 scenarios which, in turn, worked better than the 0-0 scenarios. More importantly, the 1-0 scenarios worked better than the 1-1 scenarios. Actually, the 1-0 scenarios always converge optimally for problem sizes up to 20, the largest problems we have experimented with. In the 1-0 scenarios, the exploitative behavior of the 1-level learner is compensated by the explorative behavior of the 0-level learner.

Probabilistic versus deterministic action choices in higher level agents

Instead of using a probabilistic (non-greedy) measure for selecting action, we used a greedy utility maximization strategy for the 2-level agents so far, i.e., the action chosen by 2EUM is given as $\arg\max_{a \in \mathcal{A}} EU(a)$, where $\mathcal{A}$

is the action set for the agent (this is a deterministic choice). We implemented a non-greedy version of the 2-level agent which uses a Boltzmann exploration scheme to choose its action. In this scheme, the probability of choosing action a is given by $\frac{exp(\frac{EU(a)}{T})}{\sum_{i\in\mathcal{A}} exp(\frac{EU(i)}{T})}$. Since this agent uses the expected utilities to define a probability distribution over its actions, we call it a Expected Utility based Probabilistic agent, or 2EUP agent. On the flip side, the 1-level agent we have used before, i.e., the myopic 1EUP learner, chooses action probabilistically. We also implemented a greedy 1-level learner, an 1EUM, which chooses the action that maximizes its expected utility.

We performed experiments to compare the greedy strategies with the non-greedy strategies. If both learners are greedy 1-level agents, they converge immediately to the first action combination they choose (as the corresponding payoff is then more all other actions) and hence these agents were not used together in any further experiments. The 2-level agents with non-greedy strategy converge prematurely and take more interactions to converge than 2-level agents with greedy strategies (see Figure 3). The greedy 2-level agent assumes that the other agent is an 1EUP learner and the corresponding probability calculations to generate that agent's action probabilities temper the exploitative behavior of the greedy 2-level agents and avoids premature convergence. The higher degree of exploitation in greedy 2-level agents while selecting its own action after forming the action probability distribution about the other agent causes faster convergence compared to the non-greedy 2-level agents.

We also performed experiments with the 1(greedy)-0 and 2(non-greedy)-1(non-greedy) scenarios. Though the convergence speed of this 1-0 and 2-1 scenarios are similar, only the 2-1 scenarios show premature convergence. The overall message from all this seems to be that the greedy versions of the 2-level learners, 2EUMs, are the best strategies to be used in homogeneous cooperative groups. The best mixed group in our experiments is the 2(greedy)-1(non-greedy pair).

3.2 Effects of prominence of global optima

In this series of experiments, we we varied h, the prominence of global optima compared to other solutions, by varying the off-diagonal elements in the payoff matrix. We set the diagonal elements to be 1 and created several game matrices with off-diagonal values of 0, 0.25, 0.5, 0.75. The corresponding h values are 1, 0.75, 0.5, and 0.25 respectively. All the four strategies took more time to converge as h is decreased. Of greater concern is the fact that convergence to sub-optimal solutions can increase with decreasing h if initial exploration is limited. This happens in our experiments if either the initial temperature or the temperature discounting factor in the Boltzmann exploration scheme is relatively small. Any constant value for the discount factor will cause the premature convergence problem for sufficiently large problem sizes. A solution to this problem is to use larger factors for larger problem sizes.

3.3 Effects of noisy feedback

In the experiments reported above, the feedback was deterministic. In this set of experiments, we used payoff matrices with off-diagonal element values of 0.75 and varied the amount of noise associated with the feedback (see Figure 4). Gaussian noise with zero-mean and varying standard deviation values were used for every action combination. The standard deviation was varied from 0 (deterministic payoff) to 0.5. As noise is increased, the agents take longer to converge to the global optima.

3.4 Varying matrix characteristics

We provided two 0-level agents with two distinct payoff matrices for the same action choices. Such "games" are not necessarily cooperative in nature. In this paper, however, we have deliberately chosen payoff matrices such that the game is not competitive in the sense of the corresponding game being a zero-sum one. Two particular variations of matrix types, and corresponding problems, are described below (the variables in the following correspond to Table 2):

Case analysis: Agent A's clear choice is to choose action $a0$, i.e., $x_A > m_A$, and $y_A > n_A$. For agent B, though there were no obvious choices. However, $y_B > x_B$, and hence, once agent A commits to action $a0$, agent B learns to choose action $b1$. Results are shown in figure 5.

Best plan: In these games, x_A is better than all other matrix values for agent A, and x_B is better than all other matrix values for agent B. Over time, the agents learn to choose $a0$ and $b0$ respectively 5.

4 Related work

Previous work on using reinforcement learning for coordinating multiple agents [?; ?] have relied on explicit information sharing. We, however, concentrate on systems where agents share no problem-solving knowledge. Agents that learn independently and autonomously without communication are not affected by communication delays (due to other agents being busy) or failure of a key agent (who controls information exchange or who has more information), and do not have to be worry about the reliability of the information received (Do I believe the information received? Is the communicating agent an accomplice or an adversary?).

There has been some recent theoretical work on the convergence to global optima by concurrent reinforcement learners [?]. These results does not, however, explain the convergence of individual reinforcement learners when multiple global optima exists as in the case of our experiments. Another recent work by Vidal & Durfee [?] uses the approach of a moving target function to model concurrent learners. Their model requires the use of change, learning, retention rates and volatility metrics to predict the convergence properties of learners. They claim to have been able to reproduce Claus &

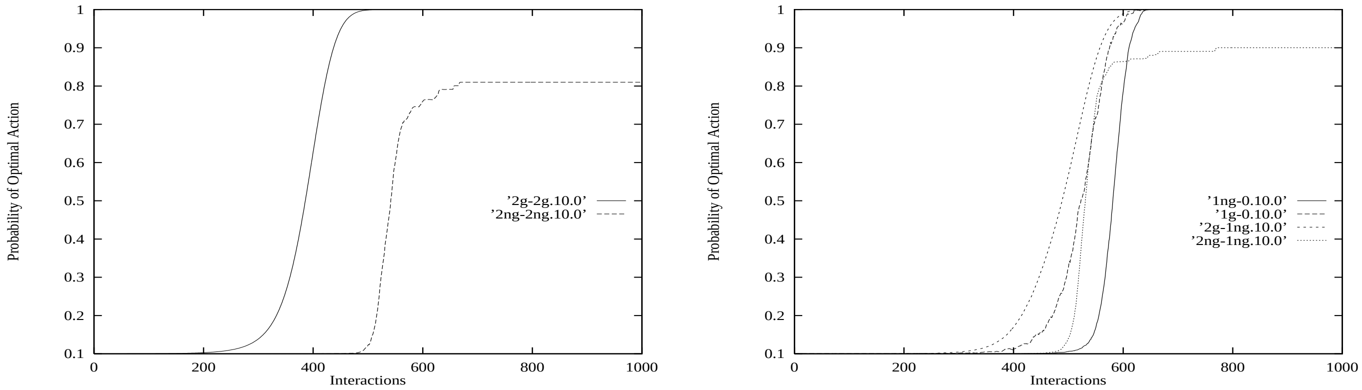


Figure 3: Relative performance of mixed and homogeneous pairs of 0, 1 and 2-level agents. Greedy and non-greedy agents are suffixed with 'g' and 'ng' respectively.

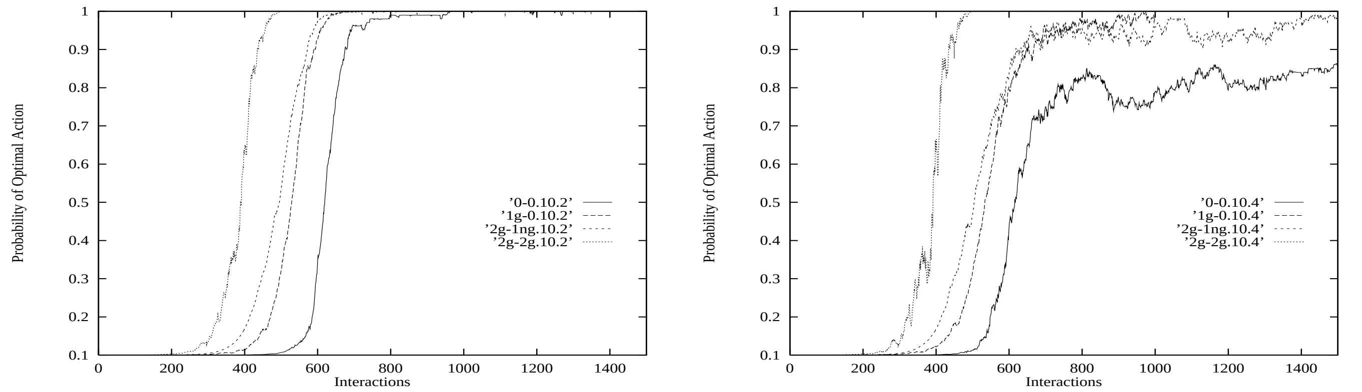


Figure 4: Relative performance of mixed and homogeneous pairs of 0, 1 and 2-level agents with varying noise in feedback. In the figure the trailing numbers represent the standard deviation of Gaussian noise in feedback.

Boutilier's results, but do not provide corresponding details. Though our results differ from Claus & Boutilier's results in some fundamental ways, and we have explained those difference in this paper, we plan to analyze Our results using Vidal & Durfee's framework.

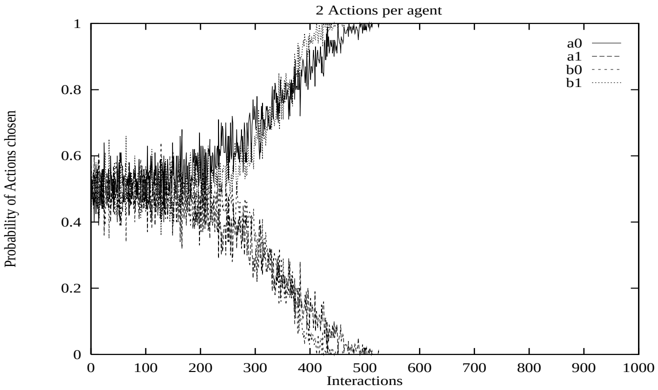
5 Conclusions

We have evaluated concurrent reinforcement agents for stylized multiagent environments as studied by Claus & Boutilier [?]. Some of our results contradict the relative performance presented in that paper. A recency-dominated myopic approach cures some but not all problems for joint learners. Our analysis suggests that a mutually reinforced bias leads the joint learners to quick convergence even if the solution reached is a suboptimal one in some cases.

We categorize the learners in a hierarchy depending on their use of models about other agents. This allows us to develop a more effective 2-level learner. We also identify a greedy trait in our 2-level learner that allow it to converge to the optimum faster. It appears that expected utility maximization is preferred to expected utility based Boltzmann exploration for the 2-level agent. Such a greedy attitude is useful also for the 1-level agent in mixed groups. As long as one agent is sufficiently explorative, the other agent can be exploitative.

We plan to dynamically adjust the learning rate to balance exploration-exploitation to avoid premature convergence. The learning rate adjustments can be based on the variance of payoffs for an action. In this paper, our experiments use games with immediate feedbacks. We plan to experiments where the feedback is available only after a fixed number of interactions. We also plan to study the scalability of such systems with the number of concurrent learners. As the number of concurrent learners increase, more uncertainty will be associated with the feedbacks received by an agent for any of its chosen action. This is likely to aggravate the convergence problem of concurrent learners.

References



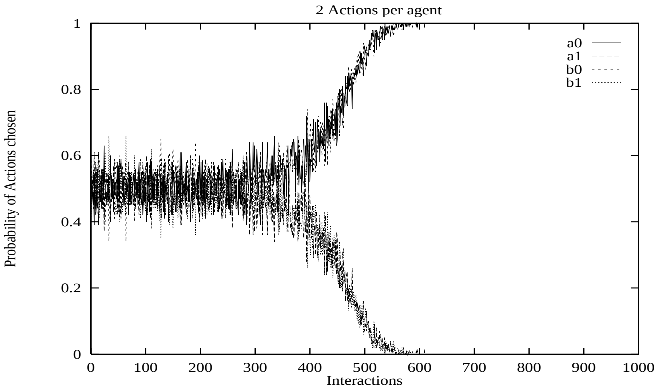


Figure 5: Individual learners on case analysis (left) and best plan (right) problems.