Rationality Assumptions and Optimality of Co-learning

Ron Sun and Dehu Qi

CECS Dept, University of Missouri
Columbia, MO 65211, USA
http://www.cecs.missouri.edu/~rsun

Abstract. This paper investigates the effect of different rationality assumptions on the performance of co-learning by multiple agents in extensive games. Extensive games involve sequences of steps and close interactions between agents, and are thus more difficult than more commonly investigated (one-step) strategic games. Rationality assumptions may thus have more complicated influences on learning, e.g., improving performance sometimes while hurting performance some other times. In testing different levels of rationality assumptions, a "double estimation" method for reinforcement learning suitable for extensive games is developed, whereby an agent learns not only its own value function but also those of other agents. Experiments based on such a reinforcement learning method are carried out using several typical examples of games. Our results indeed showed a complex pattern of effects resulting from (different levels of) rationality assumptions.

1 Introduction and Background

Assumptions that an agent makes about other agents may have significant impact on co-learning by multiple autonomous agents. This issue has not been investigated sufficiently in the past. In this paper, we will investigate the impact of such assumptions on co-learning performance in extensive games. Extensive games involve sequences of steps and close interactions between agents, and are thus more difficult than more commonly investigated (one-step) strategic games. This type of assumption may thus have more complicated influences on co-learning (e.g., improving performance sometimes while hurting performance some other times).

In order to test the effect of such assumptions, we will develop a "double estimation" method for reinforcement learning suitable for extensive games, whereby an agent learns not only its own value function but also those of other agents. We will then carry out experiments, using the method, on several typical examples of extensive games.

1.1 Extensive Games

Game theory studies decision making under the condition of interaction of multiple agents (Osborne and Rubinstein 1994). A *strategic* game is the one in which all agents choose their actions simultaneously and once for all. In contrast, in *an extensive* game, agents take turn in performing actions. Formally, an extensive game is a 4-tuple: (N, H, P, U), where N is a set of agents, H is a set of history (see Osborne and Rubinstein 1994 for further specifications), P is the player function such that $P(h)$ specifies the player after history h, U is the payoff function that maps each terminal history to a real value. Such a game is the one with perfect information – all the previous steps are observable. On the other hand, an extensive form without perfect information can be defined as a 5-tuple: (N, H, P, U, I), where N, H, P, U are defined as before, I is a set of partitions where for each agent $i \in N$, I_i is a partition of $h \in H : P(h) = i$ and $I_i^h \in I_i$ is an information set of agent i (i.e., what it can observe).

A *pure strategy* of an agent in an extensive game is defined as follows: it is a function of an agent that assigns an action (among all the possible actions) to each of its information sets. A *mixed strategy* is a probability measure over the set of an agent's pure strategies, which is used to randomly select a pure strategy to use in game playing. A *behavior strategy* is a set of probability measures, each of which is over the set of all the possible actions for an agent given an information set and is used to select stochastically actions when facing the information set.

1.2 Reinforcement Learning

Reinforcement learning (RL) (Watkins 1989, Bertsekas and Tsitsiklis 1996) is a general learning framework suitable for learning extensive games. There is a discrete-time system, the state transitions of which depend on actions performed by an agent. A Markovian process is in the working in determining a new state (from state transition) after an action is performed: $prob(s_{t+1}|s_t, a_t, s_{t-1}, a_{t-1}, ...) = prob(s_{t+1}|s_t, a_t) = p_{s_t,s_{t+1}}(a_t)$, where a_t is determined based on a policy L (i.e., $a_t = L(s_t)$). In this process, costs (or rewards) can occur for certain states and/or actions. Normally the costs/rewards accumulate additively, with or without a discount factor $\gamma \in (0, 1]$.

The cumulative cost/reward estimate J that results from following an optimal policy of actions satisfies the Bellman optimality equation (Bellman 1957):

$$J(s_t) = \max_{a \in A} \sum_{s_{t+1} \in S} p_{s_t,s_{t+1}}(a)(r(s_t) + \gamma J(s_{t+1}))$$

where r denotes cost/reward, s_t is any state and s_{t+1} is the new state resulting from action a. Or, using the notation $Q(s_t, a_t)$ (where $\max_{a \in A} Q(s, a) = J(s)$), we have

$$Q(s_t, a_t) = \sum_{s_{t+1} \in S} p_{s_t,s_{t+1}}(a_t)(r(s_t) + \gamma \max_{a \in A} Q(s_{t+1}, a))$$

Based on the Bellman optimality equation, there are a number of on-line algorithms for learning Q or J functions. One is the Q-learning algorithm of Watkins (1989). The updating is done completely on-line, without explicitly using probability estimates:

$$Q(s_t, a_t) :=$$

$$(1-\alpha)Q(s_t,a_t)+\alpha(r(s_t)+\gamma \max_{a_{t+1}\in A}(Q(s_{t+1},a_{t+1})))$$

That is,

$$\Delta Q(s_t,a_t)=\alpha(r(s_t)+\gamma \max_{a_{t+1}\in A} Q(s_{t+1},a_{t+1})-Q(s_t,a_t))$$

where a_t is determined by an action policy: e.g., $prob(a_t) = \frac{e^{Q(s_t,a_t)/\tau}}{\sum_{a\in A} e^{Q(s_t,a)/\tau}}$, where τ is the temperature that determines the degree of randomness in action selection. The updating is done based on actual state transition instead of transition probabilities; that is, on-line "simulation" is performed. Such learning allows completely autonomous learning from scratch, without a priori domain knowledge. Singh et al (1994) showed that if transition probability and reward estimates satisfy some ergodic conditions, the on-line Q-learning will converge to Q values that satisfy the Bellman optimality equation involving true means of transition probabilities and rewards.

2 Levels of Rationality Assumptions

In using RL in game theoretical settings, there is an extra twist that we must consider. When a small set of agents (for example, two) is involved, it is advantageous to learn an (relatively) accurate estimate of the action propensities of other agents. Without such knowledge (and being blind to the choices made by other agents), it may not be possible to reach jointly optimal outcomes (in some sense), as demonstrated by early work in learning games, such as Shoham and Tennenholtz (1994) and Sen and Sekaran (1998). Shoham and Tennenholtz (1994) showed that simple learning rules could be very inefficient in cooperative games. Sen and Sekaran (1998) also attempted to learn to coordinate without explicit regard to other agents. They achieved limited success in small domains. Sen and Sekaran (1998), however, pointed out that such learning is problematic "when agent actions are strongly coupled, feedback is delayed, and there is one or a few optimal behavior combination". Claus and Boutilier (1998) dealt partially with such problems in reinforcement learning of strategic, cooperative games, and tried to encourage reinforcement learners to reach the optimal outcomes by incorporating the values of "optimal strategies" of other agents in updating value functions. [1]

The issue can be formulated as the levels of rationality assumptions: whether we assume that an agent makes the rationality assumption about its opponent (that is, whether it believes that its opponent will always choose the best action with respect to its own reward function), [2] and whether we assume that an agent assumes that its opponent makes the rationality assumption about it, and so on (cf. Hu and Wellman 1998 b). If an agent assumes that its opponent is rational (i.e., tends to select the best actions in various circumstances), we call

[1] See also Littman (1994) and Hu and Wellman (1998 a).

[2] Here the word "opponent" is simply used to denote the other agent in a game.

it the level-1 rationality assumption. If an agent assumes that its opponent in turn makes the same rationality assumption about it, this is a case of the level-2 rationality assumption. In the same vein, we term it level-0 rationality if an agent tends to choose best actions in given circumstances with no regard to its opponents.

We want to investigate how different levels of rationality assumptions affect learning of multiple interacting agents. Will deeper levels of rationality assumptions help to improve co-learning? Arguments can be made that the more information one gathers, the better one learns. Or will deeper levels of rationality assumptions hurt co-learning? Arguments can also be made that inaccurate, partial, or irrelevant information may prove to be harmful. Some empirical work showed that this may indeed be the case (e.g., in strategic games; Hu and Wellman 1998 b). We thus would like to better understand this issue by performing more extensive experiments.

3 "Double-Estimation" Reinforcement Learning

Our RL algorithms that incorporate rationality assumptions are based on "double estimation": An agent not only learns its own value function (which is dependent on other agents' value functions), but also learns others' value functions (and uses them in learning its own value function).

First is the case where only the level-0 rationality assumption is made. Assuming two agents are involved, we will deal with individual learning of these two agents, not social learning (of randomly paired agents). We assume that the two agents take action alternately, and the sequence is $a_1^0, a_2^1, a_1^1, a_2^2, a_1^2$......., where generally a_1 denotes actions of agent 1, a_2 denotes actions of agent 2, and superscripts denote time steps. The corresponding observed states at the time when each of these actions is taken are, respectively, s_2^0 (when a_1^0 is taken), s_1^1 (when a_2^1 is taken), s_2^1 (when a_1^1 is taken), s_1^2 (when a_2^2 is taken), s_2^2 (when a_1^2 is taken), ... For the first part of the double estimation, self-estimation (i.e., updating the value function of agent 1) is as follows:

$$\Delta Q_1(s_1^1, a_2^1, a_1^1) = \alpha(r_1(s_1^2) + \gamma \max_{a_1^2 \in A_1} Q_1(s_1^2, a_2^2, a_1^2) - Q_1(s_1^1, a_2^1, a_1^1))$$

where Q_1 is the value function of agent 1, r_1 is the reward function for agent 1, [3] A_1 is the action set of agent 1, Note that the updating is done after action a_2^2 is completed (by agent 2 in state s_1^2). [4] This is appropriate because agent 2 may be using a stochastic policy (i.e., mixed or behavioral strategy in a game theoretic sense) and agent 1 has no a priori knowledge to determine whether this is the case or not. On the other hand, in referring to its own future action, agent 1 may reasonably assume that a greedy policy will be used and hence the MAX

[3] In game learning, reward can only occur at the end when a terminal state is reached. Thus $r(s)$ is zero except in terminal states.

[4] To get agent 2's actual action, agent 1 has to wait until agent 2 takes an action before it can learn.

operation (using the level-0 rationality assumption). (Alternatively, agent 1 may adopt a stochastic policy (mixed strategy), in which case the MAX operation should be replaced by the statistical average resulting from different possible actions.)

For the second part of the double estimation, updating the (estimated) value function of agent 2 by agent 1 (i.e., opponent-estimation) is as follows: [5]

$$\Delta Q_2(s_2^0, a_1^0, a_2^1) = \alpha(r_2(s_2^1) + \gamma Q_2(s_2^1, a_1^1, a_2^2) - Q_2(s_2^0, a_1^0, a_2^1))$$

Where Q_2 is the value function of agent 2 as estimated by agent 1, A_2 is the action set of agent 2, r_2 is the reward function for agent 2, This updating is done after action a_2^2. Here we do not take MAX in estimating future Q-values, because we try to avoid making any assumption about the actions of agent 2. [6]

The above formulation does not put the double estimation to use in learning "optimal" value functions (e.g., in the sense of encouraging convergence to optimal joint outcomes in some sense). Remedying this shortcoming, utilizing some further rationality assumptions (level-1 and level-2), we can have instead the following set of equations:

For self-estimation,

$$\Delta Q_1(s_1^1, a_2^1, a_1^1) = \alpha(r_1(s_1^2) + \gamma \max_{a_1^2 \in A_1} Q_1(s_1^2, a_2^{best}, a_1^2) - Q_1(s_1^1, a_2^1, a_1^1))$$

where $a_2^{best} = argmax_{a_2' \in A_2} Q_2(s_2^1, a_1^1, a_2')$. Here the assumption of rationality of one's opponent (hereby termed the level-1 assumption of rationality) is essential: Agent 1 assumes that agent 2 will (likely or eventually) choose the best action for its own sake (and hence the use of a_2^{best}). [7]

For opponent-estimation,

$$\Delta Q_2(s_2^0, a_1^0, a_2^1) = \alpha(r_2(s_2^1) + \gamma \max_{a_2^2 \in A_2} Q_2(s_2^1, a_1^{best}, a_2^2) - Q_2(s_2^0, a_1^0, a_2^1))$$

[5] We generally assume that an agent can observe its opponent's payoffs.

[6] (1) Note that in the above equations, we use different action orders for the two different estimations: from agent 2 to agent 1 (for learning Q_1), or from agent 1 to agent 2 (for learning Q_2). This is because it is easier to formulate value functions this way: the action of the other agent has already been performed at the time of the action for which the value function is being learned. (2) At the beginning of a game, the same updating equations may be used with dummy actions being padded at the beginning of a sequence (that is, turning $[a_1^0, a_2^0, a_1^1, a_2^1, ...]$ into $[...., a_1^{-1}, a_2^{-1}, a_1^0, a_2^0, a_1^1, a_2^1, ...]$). (3) Similarly, in case of termination without action by agent 2, we can assume agent 2 performs a pseudo-action "end".

[7] We use a_2^{best} even though we know the actual action taken by agent 2. This is because we want to take into consideration what is the best for agents, even though they may be taking suboptimal actions occasionally for reasons of exploration or simply wandering around (in a way similar to the idea of "trembling hands"; Osborne and Rubinstein 1994).

Where $a_1^{best} = argmax_{a'_1 \in A_1} Q_1(s_1^1, a_2^1, a'_1)$. Here beside making the level-1 assumption in terms of taking the MAX when estimating future Q-values, we utilize also the level-2 assumption of rationality: Agent 1 assumes that agent 2 will assume that agent 1 will (likely or eventually) choose the best action (and hence the use of a_1^{best}). If we do not want to make the level-2 assumption, we can use instead the previous equation for opponent-estimation (while using the current equation for self-estimation if we stick to the level-1 assumption). [8]

The two different sets of learning equations for value functions may be combined. As suggested by Claus and Boutilier (1998), because one set encourages the adoption of optimal actions (which may lead to jointly optimal outcomes) and the other set learns on the basis of current action probabilities, the combination may tend to "steer" both agents toward jointly optimal outcomes (in some sense). To combine the two sets of equations, we use the following equation (cf. Claus and Boutilier 1998) for self-estimation,

$$Q_1^{(total)}(*,*,*) = \mu_1 Q_1^{(r)}(*,*,*) + (1-\mu_1) Q_1^{(nr)}(*,*,*)$$

where (r) indicates the Q-values resulting from the second set of equations using double estimation with the level-1 and level-2 rationality assumptions, (nr) indicates the Q-values from the first set of equations without such assumptions, $(total)$ indicates the combined Q-values, μ_1 is the weighting factor that determines the relative contributions from the two sets of equations. Similarly, for opponent-estimation, we have

$$Q_2^{(total)}(*,*,*) = \mu_2 Q_2^{(r)}(*,*,*) + (1-\mu_2) Q_2^{(nr)}(*,*,*)$$

where μ_2 is the weighting factor that determines the relative contributions from the two sets of equations in determining the values of opponent-estimation, which may or may not be the same as μ_1. (Note that this combination is different from Claus and Boutilier (1998), in that we do not need to compute the probabilities of opponent actions and their respective Q-values because we are dealing with extensive games in which actions are taken sequentially and alternately.)

Action selection in the course of a game is based on $Q_1^{(total)}$. Note that exactly the same equations may be applied for the other agent (agent 2), if we change the subscript 1 to 2 and vice versa.

4 Some Experiments

In this section, we will explore the effects of adding levels 1 and 2 rationality assumptions to the level-0 assumption. We need to compare different levels of rationality assumptions, in various game domains, through empirical means, in

[8] Level 0 and level 1 will not be the same. because the actual action by agent 2 involves random exploration or involves an entirely different learning algorithm (e.g., GA).

terms of (1) whether learning will converge, (2) the frequency of convergence and/or the rate at which it converges, (3) if it converges, what kind of attractor it converges to (e.g., in terms of Nash equilibria or Pareto efficiency). In the present paper, we focus mostly on optimality of joint or individual outcomes.

We experiment with a range of domains, which differ in terms of (1) different lengths (or depths, i.e., numbers of steps), (2) different numbers of action choices at each step (i.e., different widths), or (3) cooperativeness. Such variations are important, given the fact that learning methods may have widely varied performance when tested in different domains, "flavored" by the domain characteristics (see e.g. Shoham and Tenenholtz 1994 for instances).

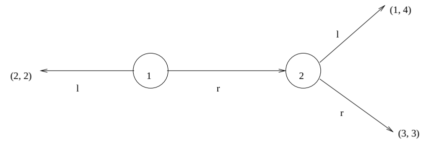
Both agents will be learning on-line, together; i.e., both start out from scratch and learn while exploring. The best actions have also to be learned on-line, not estimated a priori. In experiments, we vary the following parameters associated with our RL algorithms: (1) learning rates and discount rates, (2) combination weights (for generating Q^{total}), to obtain the best performance possible for each algorithm. We also try different temperature parameters (i.e. τ) in the Boltzmann equation. However, we perform the periodic test of agents by choosing the best actions (according to Q-values), thereby generating learning curves.

Our results show that higher levels of rationality assumptions may have varied effects, depending on a large number of factors. They may help, hurt, make a difference that is neither, or make no difference at all, on the performance of learning.

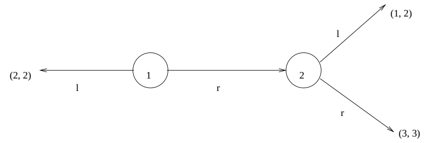
4.1 The Left/Right (LR) Game

Three variations of this game are described by Figure 1. Let us analyze the three cases of this game, comparing the level-0 and level-1 assumption in terms of learning performance. Note that in this game, level 2 is meaningless due to short sequences (of 1 or 2 steps only). Note also that for agent 2, opponent-estimation, and thus level 1 or 2, is irrelevant, since there will be no more actions after its action.

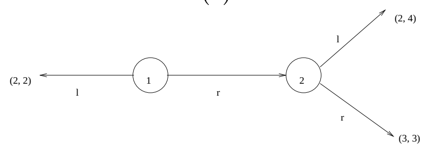
- In case 1, we expect using level 1 will speed up convergence and help the performance of agent 1. This is because using a_2^{best} (that is, using level 1), agent 1 knows that it will receive a reward of 1 if it does r; using a_2 (level 0 alone), agent 1 will receive an average of 1 and 3 if it does r, which is the same as that resulting from its action l. So, with level 1, the action sequence should converge more quickly to $(l, \perp)$. However, even with level 0 alone, the best action for agent 1 should be l eventually (which is a Nash equilibrium). This is because eventually agent 2 will settle on l (in order to receive a payoff of 4 instead of 3), and consequently the difference for agent 1 is between a payoff of 2 and a payoff of 1 (resulting from its actions l and r respectively) and thus the better action should be chosen eventually.
  In addition to speeding up convergence, using level 1 may also help to improve the performance of agent 1 by identifying the best action. Since the interests of agent 1 and 2 do not coincide in this case, helping the performance of agent 1 means hurting the performance of agent 2.



(1)



(2)



(3)

Fig. 1. Three cases of the Left/Right game. The numbers in circles indicate agents. l and r are two possible actions. The pair of numbers in parentheses indicate payoffs (where the first number is the payoff for agent 1 and the second for agent 2).

– In case 2, the use of level 1 should improve the speed of convergence as before. However, in this case, it also helps the performance of both, because in this case, the best action sequence is the same for both agents, or in other words, their interests coincide. The use of level 1 helps agent 1 to identify the best action for agent 2 being r (through opponent-estimation) and consequently the best action for agent 1 being r (through self-estimation), thus resulting in the action sequence (r, r), which is the best for both (and a Nash equilibrium). It thus improves the performance of both during learning and speeds up the convergence.

– In case 3, we expect that the use of level 1 does not have much effect on the speed of convergence because there is not much ambiguity in this situation. We also expect the level-1 curve of agent 1 to be somewhat worse, because, especially early on during learning, self-estimation based on a_2^{best} leads to a payoff of 2 for agent 1, but self-estimation based on a_2 leads to an average of 2 and 3 for agent 1. Therefore, the level-0 learner should be able to accrue more payoffs early on. However, eventually, agent 1 will receive a payoff of 2 regardless of its action (l or r) in the end, because agent 2 will settle on action l when action r is taken by agent 1 (because it leads to a payoff of 4, instead of 3). Both outcomes are Nash equilibria. In addition, when level 1 is used, agent 2 may be also worse off early on because it may be forced to receive a payoff of 2 due to the action l taken by agent 1, which may be more favored early on by agent 1 with the use of the level-1 assumption.

See the learning curves in Figures 2, 3, and 4, which fully confirmed our expectation. In the curves, we can compare level 0 and level 1 (with the combination of the two sets of equations). The figures contain learning curves averaged over 20 runs for each 10 iterations. That is, in the figures, the x axis is number of iterations (averaged over 20 runs) and the y axis is the performance measure (individual payoffs or total payoffs).

In the experiments, we randomly initialized the Q-values to some very small random positive values before the learning began, which represented initial (random) guesses. We did so for both agents and both levels. [9] We varied the combination weights. In the LR game, for agent 1, self-estimation was in the form of $Q_1(s, begin, a_1)$. Dummy actions ("begin") were padded at the beginning to make the equations work. The state s remained the same and was irrelevant in this game. The opponent-estimation was in the form of $Q_2(s, a_1, a_2)$, where a_1 was the action by agent 1 in the previous step and a_2 was the action by agent 2 in the current step. Payoffs were given at the end as indicated in the game trees (see Figure 1). For agent 2, opponent-estimation was irrelevant. For the self-estimation of agent 2, there was no need to consider actions by agent 1, because, for agent 2, actions by agent 1 were always r. The parameter settings were as follows: $\alpha = 0.05$, $\gamma = 0.8$, $\tau = 0.5$, and $\mu_1 = \mu_2 = 0.8$.

The results show that the level-1 assumption is beneficial, except in situations in which the agent fails to take into account randomness in the other agent.

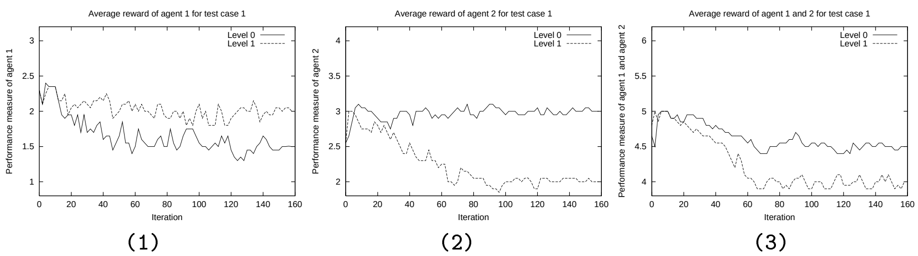
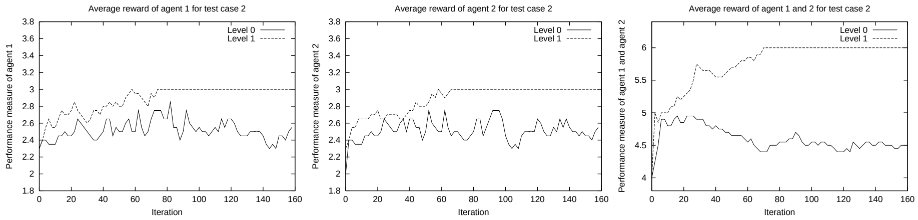


Fig. 2. The learning performance in case 1 of the left/right game. (1) The payoff of agent 1. (2) The payoff of agent 2. (3) The total payoff.

4.2 The Repeated Prisoner's Dilemma (RPD) Game

In this game (the repeated play of a strategic game – Prisoner's Dilemma), actions are organized into rounds. A "round" refers to a pair of actions by the two agents, which are simultaneous and not mutually observable at the time of

[9] So initially, the best actions of agent 2 as seen by agent 1 can be totally off. Agent 1 will then learn the true best action for agent 2 through the (simultaneous) learning process.



(1) (2) (3)

Fig. 3. The learning performance in case 2 of the left/right game. (1) The payoff of agent 1. (2) The payoff of agent 2. (3) The total payoff.

action. The action choices (C or D) and the payoff table are shown in figure 5. A fixed number of rounds is used, which is set at 8. The partial observability is captured through the use of information sets. An information set is implemented by a state that includes all the observable previous actions, up to a certain length (say, 1, 2, or 5).

In the RPD game, agent 1's self-estimation is in the form of $Q_1(s, a_2, a_1)$, in which a_2 is the action by agent 2 in the previous round (which is the immediately preceding *observable* action) and a_1 is the action by agent 1 in the current round. Likewise, the actions in s' needed for self-estimation (as in $Q(s', a'_2, a'_1)$) are the action by agent 2 in the current round and the action by agent 1 in the next round, respectively. The opponent-estimation is in the form of $Q_2(s, a_1, a_2)$, where a_1 is the action by agent 1 in the previous round (which is the immediately preceding *observable* action) and a_2 is the action by agent 2 in the current round. The next actions in s' needed for opponent-estimation (as in $Q(s', a'_1, a'_2)$) are the action by agent 1 in the current round and the action by agent 2 in the next round, respectively. State s consists of past sequences up to depth x (where x is 0 in our experiments because in this game such history is obvious from the previous action of the opponent). Dummy actions ("begin") were padded at the beginning to make the equations work. It is essentially the same for agent 2.

(1) (2) (3)

Fig. 4. The learning performance in case 3 of the left/right game. (1) The payoff of agent 1. (2) The payoff of agent 2. (3) The total payoff.

We compare here the case of learning with the level-1/2 rationality assumption and the case without. In a single-play PD game, the use of the level-1/2 assumption will not lead to a higher likelihood of jointly optimal outcomes, because mutual defection is the best (the most mutually rational) choice. [10] On the other hand, in the RPD game, an ideal outcome is as follows: If the opponent (agent 2) selects C the previous round, agent 1 would be better off selecting C, because in this case a_2^{best} is C and so the opponent will likely select C the next round, which leads to a high payoff (for both). (C is a_2^{best} under this circumstance because it is more likely to lead to further cooperation, which leads to higher payoffs.) Otherwise, if agent 1 selects D, then the opponent will likely select D the next round, which leads to worse payoffs for both. The use of the level-1/2 assumption can serve the purpose of encouraging agent 1 (and likewise agent 2) to play C, because, despite the actual action by agent 2, agent 1 "pretends" that the best action (i.e., C) is taken by agent 2 and is thus encouraged to take action C also. The learning curves are shown in Figure 6. The parameter settings are as follows: $\alpha = 0.05$, $\gamma = 0.8$, $\tau = 0.5$, and $\mu_1 = \mu_2 = 0.8$.

	C	D
C	3,3	0,5
D	5,0	1,1

Fig. 5. The game of prisoner's dilemma. Each row indicates a possible action by agent 1 (C or D). Each column indicates a possible action by agent 2 (C or D). In each pair, the first number indicates payoff for agent 1 and the second for agent 2.

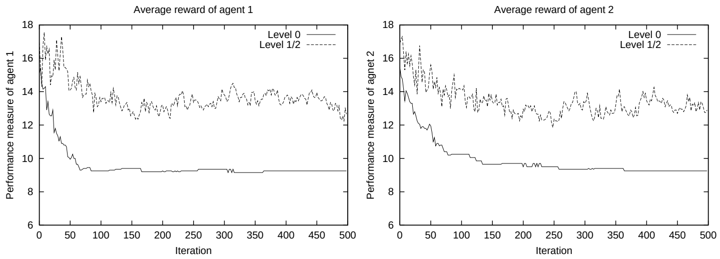
However, we do not expect such improvement over level 0 to happen in every learning run and under every parameter setting, because alternative (worse) scenarios are also likely. Under different parameter settings (e.g., increasing the temperature to 0.7), we indeed obtained opposite results as well. [11]

4.3 The Poker (P) Game

In the poker game, there are two players: the gambler and the dealer. First the gambler has the choice of betting one dollar or passing. Then the dealer has the same choice. Finally the gambler has another chance of betting (if the gambler chose to pass previously and the dealer chose to bet). There are three cards: J, Q, and K, two of which will be dealt to the two agents. The agent who receives the higher card wins the pot.

[10] Shoham and Tennenholtz (1994) managed to reach a jointly optimal outcome, but through using a dumb learning rule which unnecessarily limits even the level-0 rationality of agents with no exploration and no sense of optimality.

[11] In future work, we shall extend the length of history, as contained in states, to see what effect it may bring.



(1) (2)

Fig. 6. The learning performance in the RPD game. (1) The payoff of agent 1. (2) The payoff of agent 2.

For agent 1 (the gambler), self-estimation is in the form of $Q_1(s, a_2, a_1)$, in which a_2 is the action by agent 2 (the dealer) in the immediately preceding step and a_1 is the action by agent 1 for the current step. The state s is the information about the card of agent 1 and the actions up to the point where these two actions (a_2 and a_1) are taken, up to a depth x (where $x = 0$ in our current experiments). The opponent-estimation is in the form of $Q_2(s, a_1, a_2)$, where a_1 is the action by agent 1 in the immediately preceding step (which may be empty, in which case a dummy action "begin" is added) and a_2 is the action by agent 2 for the current step.

There are some difficulties in this game. The previous actions and an agent's own card provide the only context for learning decision making. For example, the gambler knows only his own card (which is the state) and the dealer's action. The dealer knows his own card and the gambler's action. Each agent has to estimate its own move and the opponent's next move without the benefit of knowing the opponent's card, which determines its and the opponent's best actions. But in self-estimation, the gambler, for example, can learn its own action based on the eventual payoff. In opponent-estimation, the gambler can learn the dealer's action also based on the payoff it generates eventually. Of course the payoff will vary, depending on the card the opponent has, which the estimating agent has no knowledge of. But we believe that opponent-estimation can still be useful because it can capture some statistical regularities in the card distribution, and thus the statistical regularities in action selection, and it may lead to better average payoffs.

For each agent, we plot the cases of J, Q, and K in separate figures, with each figure containing multiple curves under different levels of rationality assumptions. We assume that there is a 60% probability that the dealer will receive K (and 20% receiving J and 20% receiving Q). We randomly assign a card to the gambler that is different from the one assigned to the dealer. The reason for doing this is that since it is much more likely that the dealer will receive K, the gambler's response will be more tailored to that setting. So we can study the dynamics of the interaction of the two agents in that setting. We compare curves (level 0 vs. level 1/2) under this condition, in terms of the performance of the gambler (since the performance of the dealer is simply the opposite). We tried the following combinations of temperatures: dealer-temp= 10 or 30, and gambler-temp = 0.5, 10, or 30. We also tested the following combinations of levels of rationality assumptions: dealer-level = 0 or 1, [12] and gambler-level = 0 or 1/2. [13] The performance of dealer-temp=10/dealer-level=0 and gambler-temp=30/gambler-level=0 vs. 1/2, which is fairly representative, is shown in Figure 7. The effect of the level-1/2 assumption is helpful sometimes: It helps the performance of the gambler when the gambler receives J or Q, which is the more common situation for the gambler to deal with (because the dealer receives K 60% of the time).

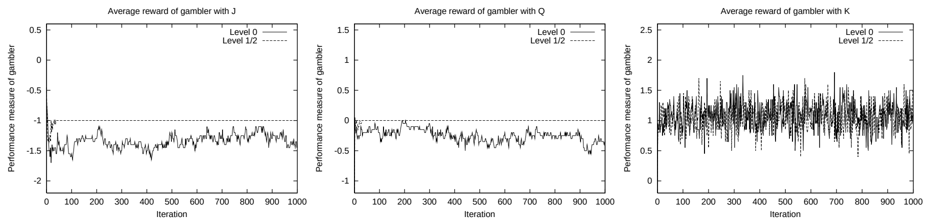


Fig. 7. The learning performance in the Poker game, in terms of the payoff of the gambler (three cases).

5 Discussions

The difference in the results of these three tasks may be attributed to the complexities of the tasks. The LR game is the simplest game we examined, and we can easily analyze the settings and anticipate the outcome. The Poker game is the most complex: Too many factors are involved in determining the learning process and as a result, we failed to clearly delineate the conditions under which the level-1/2 assumption is useful. Generally speaking, however, the more complex a task is, the more likely an agent is to assume something incorrect about its opponent during opponent-estimation (Hu and Wellman 1998 b). Thus, it is more likely to lead to worsened performance (as a result of wrong or incomplete information about opponents). The LR and RPD games also pointed to the fact that randomness (and other factors) that was not taken into account in level-1 or level-2 opponent-estimation led to worsened performance. In all, the three games point to the conclusion that the effect of higher levels of rationality

[12] The dealer's level-2 assumption is irrelevant, since there can only be one more step in an action sequence.

[13] $\alpha = 0.05$, $\gamma = 0.8$, and $\mu_1 = \mu_2 = 0.8$.

assumptions is not uniform: It may either help or hurt performance and needs to be understood on a case-by-case basis.

Judging from the results of the above experiments, the necessity of the benefit or the harm of higher levels of rationality assumptions on the part of an agent regarding opponents is unclear. In contrast to some previous results, we show here that such assumptions are not necessarily harmful, which seemed to be what was commonly perceived in the literature (e.g., with regard to auction games; Hu and Wellman 1998 b).

Some limitations and shortcomings of the present study include that it deals only with a fixed opponent: We assume that throughout a game, agents are fixed, not randomly drawn from a population. It is not clear how the results extend to cases of random encounters of samples out of a large population of opponents. Another shortcoming is the lack of deeper levels of rationality assumptions: We have not yet explored deeper levels of rationality assumptions beyond level 2. We may need longer chains of actions in order to test deeper levels. We also need to include longer history (information sets) in states.

Finally, it should be noted that our RL approach here is different from multi-agent RL that involves a central controller (such as Sun and Peterson 1999, Tham 1995) or cooperative game learning scenarios in which bidding and other cooperative mechanisms may be used (e.g., Sun and Sessions 1999). We do not assume here any explicit act of cooperation, or any central control that coordinates or even commands cooperation. The present approach is more generic.

References

R. Bellman, (1957). *Dynamic Programming.* Princeton University Press, Princeton, NJ.

D. Bertsekas and J. Tsitsiklis, (1996). *Neuro-Dynamic Programming.* Athena Scientific, Belmont, MA.

C. Claus and C. Boutilier, (1998). The dynamics of reinforcement learning in cooperative multiagent systems. *Proceedings of AAAI'98.* AAAI Press, San Mateo, CA.

D. Fudenberg and D. Levine, (1998). *The Theory of Learning in Games.* MIT Press, Cambridge, MA.

T. Haynes and S. Sen, (1996). Co-adaptation in a team. *International Journal of Computational Intelligence and Organizations.*

J. Hu and M. Wellman, (1998 a). Multiagent reinforcement learning: theoretical framework and an algorithm. *Proceedings of International Conference on Machine Learning*, 242-250. Morgan Kaufmann, San Francisco, CA.

J. Hu and M. Wellman, (1998 b). Online learning about other agents in a dynamic multiagent system. *Second International Conference on Autonomous Agents.* ACM Press, New York.

M. Littman, (1994). Markov games as a framework for multi-agent reinfocement learning. *Proc. of the 11th International conference on Machine Learning*, 157-163. Morgan Kaufmann, San Francisco, CA.

M. Osborne and A. Rubinstein, (1994). *A Course on Game Theory.* MIT Press, Cambridge, MA.

R. Salustowicz, M. Wiering, and J. Schmidhuber, (1998). Learning team strategies: soccer case studies. *Machine Learning.* 1998

S. Sen and M. Sekaran, (1998). Individual learning of coordination knowledge. *Journal of Experimental and Theoretical Artificial Intelligence*, 10, 333-356.

Y. Shoham and M. Tennenholtz, (1994). Co-learning and the evolution of social activity. Technical Report STAN-CS-TR-94-1511, Stanford University.

S. Singh, T. Jaakkola, and M. Jordan, (1994). Reinforcement learning with soft state aggregation. In: S.J. Hanson J. Cowan and C. L. Giles, eds. *Advances in Neural Information Processing Systems 7.* Morgan Kaufmann, San Mateo, CA.

R. Sun and T. Peterson, (1999). Multi-agent reinforcement learning: weighting and partitioning. *Neural Networks*, Vol.12, No.4-5. pp.127-153.

R. Sun and C. Sessions, (1999). Bidding in reinforcement learning: a paradigm for multi-agent systems. *Proc. of The Third International Conference on Autonomous Agents (AGENTS'99)*, Seattle, WA.

M. Tan, (1993). Multi-agent reinforcement learning: independent vs. cooperative agents. *Proceedings of Machine Learning Conference.* Morgan Kaufmann, San Francisco, CA.

C. Tham, (1995). Reinforcement learning of multiple tasks using a hierarchical CMAC architecture. *Robotics and Autonomous Systems.* 15, 247-274.

M. Vidal and E.H. Durfee, (1998). Learning nested models in an information economy. *Journal of Experimental and Theoretical Artificial Intelligence*, 10 (3), 291-308.

C. Watkins, (1989). *Learning with Delayed Rewards.* Ph.D Thesis, Cambridge University, Cambridge, UK.

G. Weiss, (1995). Distributed reinforcement learning. *Robotics and Autonomous Systems*, 15, 135-142.