Journal of Machine Learning Research 4 (2003) 1039-1069 Submitted 11/01; Revised 10/02; Published 11/03

Nash Q-Learning for General-Sum Stochastic Games

Junling Hu JUNLING@TALKAI.COM
Talkai Research
843 Roble Ave., 2
Menlo Park, CA 94025, USA

Michael P. Wellman WELLMAN@UMICH.EDU
Artificial Intelligence Laboratory
University of Michigan
Ann Arbor, MI 48109-2110, USA

Editor: Craig Boutilier

Abstract

We extend Q-learning to a noncooperative multiagent context, using the framework of general-sum stochastic games. A learning agent maintains Q-functions over joint actions, and performs updates based on assuming Nash equilibrium behavior over the current Q-values. This learning protocol provably converges given certain restrictions on the stage games (defined by Q-values) that arise during learning. Experiments with a pair of two-player grid games suggest that such restrictions on the game structure are not necessarily required. Stage games encountered during learning in both grid environments violate the conditions. However, learning consistently converges in the first grid game, which has a unique equilibrium Q-function, but sometimes fails to converge in the second, which has three different equilibrium Q-functions. In a comparison of offline learning performance in both games, we find agents are more likely to reach a joint optimal path with Nash Q-learning than with a single-agent Q-learning method. When at least one agent adopts Nash Q-learning, the performance of both agents is better than using single-agent Q-learning. We have also implemented an online version of Nash Q-learning that balances exploration with exploitation, yielding improved performance.

Keywords: Reinforcement Learning, Q-learning, Multiagent Learning

1. Introduction

Researchers investigating learning in the context of multiagent systems have been particularly attracted to reinforcement learning techniques (Kaelbling et al., 1996, Sutton and Barto, 1998), perhaps because they do not require environment models and they allow agents to take actions while they learn. In typical multiagent systems, agents lack full information about their counterparts, and thus the multiagent environment constantly changes as agents learn about each other and adapt their behaviors accordingly. Among reinforcement techniques, Q-learning (Watkins, 1989, Watkins and Dayan, 1992) has been especially well-studied, and possesses a firm foundation in the theory of Markov decision processes. It is also quite easy to use, and has seen wide application, for example to such areas as (to arbitrarily choose four diverse instances) cellular telephone channel allocation (Singh and Bertsekas, 1996), spoken dialog systems (Walker, 2000), robotic control (Hagen, 2001), and computer vision (Bandera et al., 1996). Although the single-agent properties do not transfer directly to the multiagent context, its ease of application does, and the method has been employed in such multiagent domains as robotic soccer (Stone and Sutton, 2001, Balch, 1997), predator-and-prey pursuit games (Tan, 1993, De Jong, 1997, Ono and Fukumoto, 1996), and Internet pricebots (Kephart and Tesauro, 2000).

Whereas it is possible to apply Q-learning in a straightforward fashion to each agent in a multiagent system, doing so (as recognized in several of the studies cited above) neglects two issues specific to the multiagent context. First, the environment consists of other agents who are similarly adapting, thus the environment is no longer stationary, and the familiar theoretical guarantees no longer apply. Second, the nonstationarity of the environment is not generated by an arbitrary stochastic process, but rather by other agents, who might be presumed rational or at least regular in some important way. We might expect that accounting for this explicitly would be advantageous to the learner. Indeed, several studies (Littman, 1994, Claus and Boutilier, 1998, Hu and Wellman, 2000) have demonstrated situations where an agent who considers the effect of joint actions outperforms a corresponding agent who learns only in terms of its own actions. Boutilier (1999) studies the possibility of reasoning explicitly about coordination mechanisms, including learning processes, as a way of improving joint performance in coordination games.

In extending Q-learning to multiagent environments, we adopt the framework of general-sum stochastic games. In a *stochastic game*, each agent's reward depends on the joint action of all agents and the current state, and state transitions obey the Markov property. The stochastic game model includes Markov decision processes as a special case where there is only one agent in the system. *General-sum games* allow the agents' rewards to be arbitrarily related. As special cases, zero-sum games are instances where agents' rewards are always negatively related, and in coordination games rewards are always positively related. In other cases, agents may have both compatible and conflicting interests. For example, in a market system, the buyer and seller have compatible interests in reaching a deal, but have conflicting interests in the direction of price.

The baseline solution concept for general-sum games is the *Nash equilibrium* (Nash, 1951). In a Nash equilibrium, each player effectively holds a correct expectation about the other players' behaviors, and acts rationally with respect to this expectation. Acting rationally means the agent's strategy is a best response to the others' strategies. Any deviation would make that agent worse off. Despite its limitations, such as non-uniqueness, Nash equilibrium serves as the fundamental solution concept for general-sum games.

In single-agent systems, the concept of optimal Q-value can be naturally defined in terms of an agent maximizing its own expected payoffs with respect to a stochastic environment. In multiagent systems, Q-values are contingent on other agents' strategies. In the framework of general-sum stochastic games, we define optimal Q-values as Q-values received in a Nash equilibrium, and refer to them as *Nash Q-values*. The goal of learning is to find Nash Q-values through repeated play. Based on learned Q-values, our agent can then derive the Nash equilibrium and choose its actions accordingly.

In our algorithm, called *Nash Q-learning* (NashQ), the agent attempts to learn its equilibrium Q-values, starting from an arbitrary guess. Toward this end, the Nash Q-learning agent maintains a model of other agents' Q-values and uses that information to update its own Q-values. The updating rule is based on the expectation that agents would take their equilibrium actions in each state.

Our goal is to find the best strategy for our agent, relative to how other agents play in the game. In order to do this, our agents have to learn about other agents' strategies, and construct a best response. Learning about other agents' strategies involves forming conjectures on other agents' behavior (Wellman and Hu, 1998). One can adopt a purely behavioral approach, inferring agent's policies directly based on observed action patterns. Alternatively, one can base conjectures on a model that presumes the agents are rational in some sense, and attempt to learn their underlying preferences. If rewards are observable, it is possible to construct such a model. The Nash equilibrium approach can then be considered as one way to form conjectures, based on game-theoretic conceptions of rationality.

We have proved the convergence of Nash Q-learning, albeit under highly restrictive technical conditions. Specifically, the learning process converges to Nash Q-values if every stage game (defined by interim Q-values) that arises during learning has a global optimum point, and the agents update according to values at this point. Alternatively, it will also converge if every stage game has a saddle point, and agents update in terms of these. In general, properties of stage games during learning are difficult to ensure. Nonetheless, establishing sufficient convergence conditions for this learning process may provide a useful starting point for analysis of extended methods or interesting special cases.

We have constructed two grid-world games to test our learning algorithm. Our experiments confirm that the technical conditions for our theorem can be easily violated during the learning process. Nevertheless, they also indicate that learning reliably converges in Grid Game 1, which has three equally-valued global optimal points in equilibrium, but does not always converge in Grid Game 2, which has neither a global optimal point nor a saddle point, but three sets of other types of Nash Q-values in equilibrium.

For both games, we evaluate the offline learning performance of several variants of multiagent Q-learning. When agents learn in an environment where the other agent acts randomly, we find agents are more likely to reach an optimal joint path with Nash Q-learning than with separate single-agent Q-learning. When at least one agent adopts the Nash Q-learning method, both agents are better off. We have also implemented an online version of Nash Q-learning method that balances exploration with exploitation, yielding improved performance.

Experimentation with the algorithm is complicated by the fact that there might be multiple Nash equilibrium points for a stage game during learning. In our experiments, we choose one Nash equilibrium either based on the expected reward it brings, or based on the order it is ranked in solution list. Such an order is determined solely by the action sequence, which has little to do with the property of Nash equilibrium.

In the next section, we introduce the basic concepts and background knowledge for our learning method. In Section 3, we define the Nash Q-learning algorithm, and analyze its computational complexity. We prove that the algorithm converges under specified conditions in Section 4. Section 5 presents our experimental results, followed by sections summarizing related literature and discussing the contributions of this work.

2. Background Concepts

As noted above, our NashQ algorithm generalizes single-agent Q-learning to stochastic games by employing an equilibrium operator in place of expected utility maximization. Our description adopts notation and terminology from the established frameworks of Q-learning and game theory.

2.1 Single-Agent Q-Learning

Q-learning (Watkins and Dayan, 1992) defines a learning method within a *Markov decision process*.

Definition 1 *A Markov Decision Process is a tuple $\langle S, A, r, p \rangle$, where S is the discrete state space, A is the discrete action space, $r : S \times A \to R$ is the reward function of the agent, and $p : S \times A \to \Delta(S)$ is the transition function, where $\Delta(S)$ is the set of probability distributions over state space S.*

In a Markov decision process, an agent's objective is to find a strategy (policy) π so as to maximize the sum of discounted expected rewards,

$$v(s,\pi) = \sum_{t=0}^{\infty} \beta^t E(r_t|\pi, s_0 = s), \tag{1}$$

where s is a particular state, s_0 indicates the initial state, r_t is the reward at time t, $\beta \in [0,1)$ is the discount factor. $v(s,\pi)$ represents the *value* for state s under strategy π. A *strategy* is a plan for playing a game. Here $\pi = (\pi_0, \ldots, \pi_t, \ldots)$ is defined over the entire course of the game, where π_t is called the *decision rule* at time t. A decision rule is a function $\pi_t : \mathbf{H}_t \to \Delta(A)$, where $\mathbf{H}_t$ is the space of possible histories at time t, with each $H_t \in \mathbf{H}_t$, $H_t = (s_0, a_0, \ldots, s_{t-1}, a_{t-1}, s_t)$, and $\Delta(A)$ is the space of probability distributions over the agent's actions. π is called a *stationary strategy* if $\pi_t = \bar{\pi}$ for all t, that is, the decision rule is independent of time. π is called a *behavior strategy* if its decision rule may depend on the history of the game play, $\pi_t = f_t(H_t)$.

The standard solution to the problem above is through an iterative search method (Puterman, 1994) that searches for a fixed point of the following *Bellman* equation:

$$v(s,\pi^*) = \max_a \Big\{ r(s,a) + \beta \sum_{s'} p(s'|s,a) v(s',\pi^*) \Big\}, \tag{2}$$

where $r(s,a)$ is the reward for taking action a at state s, s' is the next state, and $p(s'|s,a)$ is the probability of transiting to state s' after taking action a in state s. A solution π^* that satisfies (2) is guaranteed to be an optimal policy.

A learning problem arises when the agent does not know the reward function or the state transition probabilities. If an agent directly learns about its optimal policy without knowing either the reward function or the state transition function, such an approach is called *model-free reinforcement learning*, of which Q-learning is one example.

The basic idea of Q-learning is that we can define a function Q (henceforth, the *Q-function*) such that

$$Q^*(s,a) = r(s,a) + \beta \sum_{s'} p(s'|s,a) v(s',\pi^*). \tag{3}$$

By this definition, $Q^*(s,a)$ is the total discounted reward of taking action a in state s and then following the optimal policy thereafter. By equation (2) we have

$$v(s,\pi^*) = \max_a Q^*(s,a).$$

If we know $Q^*(s,a)$, then the optimal policy π^* can be found by simply identifying the action that maximizes $Q^*(s,a)$ under the state s. The problem is then reduced to finding the function $Q^*(s,a)$ instead of searching for the optimal value of $v(s,\pi^*)$.

Q-learning provides us with a simple updating procedure, in which the agent starts with arbitrary initial values of $Q(s,a)$ for all $s \in S, a \in A$, and updates the Q-values as follows:

$$Q_{t+1}(s_t,a_t) = (1-\alpha_t) Q_t(s_t,a_t) + \alpha_t \Big[r_t + \beta \max_a Q_t(s_{t+1},a) \Big], \tag{4}$$

where $\alpha_t \in [0,1)$ is the learning rate sequence. Watkins and Dayan (1992) proved that sequence (4) converges to $Q^*(s,a)$ under the assumption that all states and actions have been visited infinitely often and the learning rate satisfies certain constraints.

2.2 Stochastic Games

The framework of stochastic games (Filar and Vrieze, 1997, Thusijsman, 1992) models multiagent systems with discrete time[1] and noncooperative nature. We employ the term "noncooperative" in the technical game-theoretic sense, where it means that agents pursue their individual goals and cannot form an enforceable agreement on their joint actions (unless this is modeled explicitly in the game itself). For a detailed distinction between cooperative and noncooperative games, see the discussion by Harsanyi and Selten (1988).

In a stochastic game, agents choose actions simultaneously. The state space and action space are assumed to be discrete. A standard formal definition (Thusijsman, 1992) follows.

Definition 2 *An n-player stochastic game Γ is a tuple $\langle S, A^1, \ldots, A^n, r^1, \ldots, r^n, p \rangle$, where S is the state space, A^i is the action space of player i ($i = 1, \ldots, n$), $r^i : S \times A^1 \times \cdots \times A^n \to R$ is the payoff function for player i, $p : S \times A^1 \times \cdots \times A^n \to \Delta(S)$ is the transition probability map, where $\Delta(S)$ is the set of probability distributions over state space S.*

Given state s, agents independently choose actions $a^1, \ldots, a^n$, and receive rewards $r^i(s, a^1, \ldots, a^n)$, $i = 1, \ldots, n$. The state then transits to the next state s' based on fixed transition probabilities, satisfying the constraint

$$\sum_{s' \in S} p(s'|s, a^1, \ldots, a^n) = 1.$$

In a *discounted stochastic game*, the objective of each player is to maximize the discounted sum of rewards, with discount factor $\beta \in [0,1)$. Let π^i be the strategy of player i. For a given initial state s, player i tries to maximize

$$v^i(s, \pi^1, \pi^2, \ldots, \pi^n) = \sum_{t=0}^{\infty} \beta^t E(r_t^1 | \pi^1, \pi^2, \ldots, \pi^n, s_0 = s).$$

2.3 Equilibrium Strategies

A Nash equilibrium is a joint strategy where each agent's is a best response to the others'. For a stochastic game, each agent's strategy is defined over the entire time horizon of the game.

Definition 3 *In stochastic game Γ, a Nash equilibrium point is a tuple of n strategies $(\pi_*^1, \ldots, \pi_*^n)$ such that for all $s \in S$ and $i = 1, \ldots, n$,*

$$v^i(s, \pi_*^1, \ldots, \pi_*^n) \geq v^i(s, \pi_*^1, \ldots, \pi_*^{i-1}, \pi^i, \pi_*^{i+1}, \ldots, \pi_*^n) \quad \text{for all } \pi^i \in \Pi^i,$$

where Π^i is the set of strategies available to agent i.

The strategies that constitute a Nash equilibrium can in general be behavior strategies or stationary strategies. The following result shows that there always exists an equilibrium in stationary strategies.

[1] For a model of continuous-time multiagent systems, see literature on *differential games* (Isaacs, 1975, Petrosjan and Zenkevich, 1996).

Theorem 4 *(Fink, 1964) Every n-player discounted stochastic game possesses at least one Nash equilibrium point in stationary strategies.*

In this paper, we limit our study to stationary strategies. Non-stationary strategies, which allow conditioning of action on history of play, are more complex, and relatively less well-studied in the stochastic game framework. Work on learning in infinitely repeated games (Fudenberg and Levine, 1998) suggests that there are generally a great multiplicity of non-stationary equilibria. This fact is partially demonstrated by Folk Theorems (Osborne and Rubinstein, 1994).

3. Multiagent Q-learning

We extend Q-learning to multiagent systems, based on the framework of stochastic games. First, we redefine Q-values for multiagent case, and then present the algorithm for learning such Q-values. Then we provide an analysis for computational complexity of this algorithm.

3.1 Nash Q-Values

To adapt Q-learning to the multiagent context, the first step is recognizing the need to consider joint actions, rather than merely individual actions. For an *n*-agent system, the Q-function for any individual agent becomes $Q(s,a^1,\ldots,a^n)$, rather than the single-agent Q-function, $Q(s,a)$. Given the extended notion of Q-function, and Nash equilibrium as a solution concept, we define a *Nash Q-value* as the expected sum of discounted rewards when all agents follow specified Nash equilibrium strategies from the next period on. This definition differs from the single-agent case, where the future rewards are based only on the agent's own optimal strategy.

More precisely, we refer to Q^i_* as a *Nash Q-function* for agent *i*.

Definition 5 *Agent i's* Nash Q-function *is defined over* $(s,a^1,\ldots,a^n)$*, as the sum of Agent i's current reward plus its future rewards when all agents follow a joint Nash equilibrium strategy. That is,*

$$\begin{aligned} Q^i_*(s,a^1,\ldots,a^n) = \\ r^i(s,a^1,\ldots,a^n) + \beta \sum_{s' \in S} p(s'|s,a^1,\ldots,a^n) v^i(s',\pi^1_*,\ldots,\pi^n_*), \end{aligned} \tag{5}$$

where $(\pi^1_*,\ldots,\pi^n_*)$ *is the joint Nash equilibrium strategy,* $r^i(s,a^1,\ldots,a^n)$ *is agent i's one-period reward in state s and under joint action* $(a^1,\ldots,a^n)$*,* $v^i(s',\pi^1_*,\ldots,\pi^n_*)$ *is agent i's total discounted reward over infinite periods starting from state* s' *given that agents follow the equilibrium strategies.*

In the case of multiple equilibria, different Nash strategy profiles may support different Nash Q-functions.

Table 1 summarizes the difference between single agent systems and multiagent systems on the meaning of Q-function and Q-values.

3.2 The Nash Q-Learning Algorithm

The Q-learning algorithm we propose resembles standard single-agent Q-learning in many ways, but differs on one crucial element: how to use the Q-values of the next state to update those of the current state. Our multiagent Q-learning algorithm updates with future Nash equilibrium payoffs, whereas single-agent Q-learning updates are based on the agent's own maximum payoff. In order to

	Multiagent	**Single-Agent**
Q-function	$Q(s,a^1,\ldots,a^n)$	$Q(s,a)$
"Optimal" Q-value	Current reward + Future rewards when all agents play specified Nash equilibrium strategies from the next period onward (Equation (5))	Current reward + Future rewards by playing the optimal strategy from the next period onward (Equation (3))

Table 1: Definitions of Q-values

learn these Nash equilibrium payoffs, the agent must observe not only its own reward, but those of others as well. For environments where this is not feasible, some observable proxy for other-agent rewards must be identified, with results dependent on how closely the proxy is related to actual rewards.

Before presenting the algorithm, we need to clarify the distinction between Nash equilibria for a *stage game* (one-period game), and for the stochastic game (many periods).

Definition 6 *An n-player stage game is defined as $(M^1,\ldots,M^n)$, where for $k=1,\ldots,n$, M^k is agent k's payoff function over the space of joint actions, $M^k=\{r^k(a^1,\ldots,a^n)|a^1\in A^1,\ldots,a^n\in A^n\}$, and r^k is the reward for agent k.*

Let σ^{-k} be the product of strategies of all agents other than k, $\sigma^{-k}\equiv\sigma^1\cdots\sigma^{k-1}\cdot\sigma^{k+1}\cdots\sigma^n$.

Definition 7 *A joint strategy $(\sigma^1,\ldots,\sigma^n)$ constitutes a* Nash equilibrium *for the stage game $(M^1,\ldots,M^n)$ if, for $k=1,\ldots,n$,*

$$\sigma^k\sigma^{-k}M^k\geq\hat{\sigma}^k\sigma^{-k}M^k \quad \text{for all } \sigma^k\in\hat{\sigma}(A^k).$$

Our learning agent, indexed by i, learns about its Q-values by forming an arbitrary guess at time 0. One simple guess would be letting $Q^i_0(s,a^1,\ldots,a^n)=0$ for all $s\in S, a^1\in A^1,\ldots,a^n\in A^n$. At each time t, agent i observes the current state, and takes its action. After that, it observes its own reward, actions taken by all other agents, others' rewards, and the new state s'. It then calculates a Nash equilibrium $\pi^1(s')\cdots\pi^n(s')$ for the stage game $(Q^1_t(s'),\ldots,Q^n_t(s'))$, and updates its Q-values according to

$$Q^i_{t+1}(s,a^1,\ldots,a^n)=(1-\alpha_t)Q^i_t(s,a^1,\ldots,a^n)+\alpha_t\left[r^i_t+\beta NashQ^i_t(s')\right], \tag{6}$$

where

$$NashQ^i_t(s')=\pi^1(s')\cdots\pi^n(s')\cdot Q^i_t(s'), \tag{7}$$

Different methods for selecting among multiple Nash equilibria will in general yield different updates. $NashQ^i_t(s')$ is agent i's payoff in state s' for the selected equilibrium. Note that $\pi^1(s')\cdots\pi^n(s')\cdot Q^i_t(s')$ is a scalar. This learning algorithm is summarized in Table 2.

In order to calculate the Nash equilibrium $(\pi^1(s'),\ldots,\pi^n(s'))$, agent i would need to know $Q^1_t(s'),\ldots,Q^n_t(s')$. Information about other agents' Q-values is not given, so agent i must learn about them too. Agent i forms conjectures about those Q-functions at the beginning of play, for example, $Q^j_0(s,a^1,\ldots,a^n)=0$ for all j and all $s,a^1,\ldots,a^n$. As the game proceeds, agent i observes

```
Initialize:
  Let $t = 0$, get the initial state $s_0$.
  Let the learning agent be indexed by $i$.
  For all $s \in S$ and $a^j \in A^j$, $j = 1, \ldots, n$, let $Q_t^j(s, a^1, \ldots, a^n) = 0$.
Loop
  Choose action $a_t^i$.
  Observe $r_t^1, \ldots, r_t^n; a_t^1, \ldots, a_t^n$, and $s_{t+1} = s'$
  Update $Q_t^j$ for $j = 1, \ldots, n$
    $Q_{t+1}^j(s, a^1, \ldots, a^n) = (1 - \alpha_t) Q_t^j(s, a^1, \ldots, a^n) + \alpha_t [r_t^j + \beta NashQ_t^j(s')]$
    where $\alpha_t \in (0, 1)$ is the learning rate, and $NashQ_t^k(s')$ is defined in (7)
  Let $t := t + 1$.
```

Table 2: The Nash Q-learning algorithm

other agents' immediate rewards and previous actions. That information can then be used to update agent i's conjectures on other agents' Q-functions. Agent i updates its beliefs about agent j's Q-function, according to the same updating rule (6) it applies to its own,

$$Q_{t+1}^j(s, a^1, \ldots, a^n) = (1 - \alpha_t) Q_t^j(s, a^1, \ldots, a^n) + \alpha_t \left[r_t^j + \beta NashQ_t^j(s') \right]. \tag{8}$$

Note that $\alpha_t = 0$ for $(s, a^1, \ldots, a^n) \neq (s_t, a_t^1, \ldots, a_t^n)$. Therefore (8) does not update all the entries in the Q-functions. It updates only the entry corresponding to the current state and the actions chosen by the agents. Such updating is called *asynchronous updating*.

3.3 Complexity of the Learning Algorithm

According to the description above, the learning agent needs to maintain n Q-functions, one for each agent in the system. These Q-functions are maintained internally by the learning agent, assuming that it can observe other agents' actions and rewards.

The learning agent updates $(Q^1, \ldots, Q^n)$, where each Q^j, $j = 1, \ldots, n$, is made of $Q^j(s, a^1, \ldots, a^n)$ for all $s, a^1, \ldots, a^n$. Let $|S|$ be the number of states, and let $|A^i|$ be the size of agent i's action space A^i. Assuming $|A^1| = \cdots = |A^n| = |A|$, the total number of entries in Q^k is $|S| \cdot |A|^n$. Since our learning agent has to maintain n Q-tables, the total space requirement is $n|S| \cdot |A|^n$. Therefore our learning algorithm, in terms of space complexity, is linear in the number of states, polynomial in the number of actions, but exponential in the number of agents.[2]

The algorithm's running time is dominated by the calculation of Nash equilibrium used in the Q-function update. The computational complexity of finding an equilibrium in matrix games is unknown. Commonly used algorithms for 2-player games have exponential worst-case behavior, and approximate methods are typically employed for n-player games (McKelvey and McLennan, 1996).

[2] Given known locality in agent interaction, it is sometimes possible to achieve more compact representations using graphical models (Kearns et al., 2001, Koller and Milch, 2001).

4. Convergence

We would like to prove the convergence of Q_t^i to an equilibrium Q_*^i for the learning agent i. The value of Q_*^i is determined by the joint strategies of all agents. That means our agent has to learn Q-values of all the agents and derive strategies from them. The learning objective is $(Q_*^1,\dots,Q_*^n)$, and we have to show the convergence of $(Q_t^1,\dots,Q_t^n)$ to $(Q_*^1,\dots,Q_*^n)$.

4.1 Convergence Proof

Our convergence proof requires two basic assumptions about infinite sampling and decaying of learning rate. These two assumptions are similar to those in single-agent Q-learning.

Assumption 1 *Every state $s \in S$ and action $a^k \in A^k$ for $k = 1,\dots,n$, are visited infinitely often.*

Assumption 2 *The learning rate α_t satisfies the following conditions for all $s,t,a^1,\dots,a^n$:*

1. $0 \le \alpha_t(s,a^1,\dots,a^n) < 1, \sum_{t=0}^{\infty} \alpha_t(s,a^1,\dots,a^n) = \infty, \sum_{t=0}^{\infty}[\alpha_t(s,a^1,\dots,a^n)]^2 < \infty$*, and the latter two hold uniformly and with probability 1.*
2. $\alpha_t(s,a^1,\dots,a^n) = 0$ if $(s,a^1,\dots,a^n) \neq (s_t,a_t^1,\dots,a_t^n)$.

The second item in Assumption 2 states that the agent updates only the Q-function element corresponding to current state s_t and actions $a_t^1,\dots,a_t^n$.

Our proof relies on the following lemma by Szepesvári and Littman (1999), which establishes the convergence of a general Q-learning process updated by a pseudo-contraction operator. Let $\mathbb{Q}$ be the space of all Q functions.

Lemma 8 (Szepesvári and Littman (1999), Corollary 5) *Assume that α_t satisfies Assumption 2 and the mapping $P_t : \mathbb{Q} \to \mathbb{Q}$ satisfies the following condition: there exists a number $0 < \gamma < 1$ and a sequence $\lambda_t \ge 0$ converging to zero with probability 1 such that $\| P_tQ - P_tQ_* \| \le \gamma \| Q - Q_* \| + \lambda_t$ for all $Q \in \mathbb{Q}$ and $Q_* = E[P_tQ_*]$, then the iteration defined by*

$$Q_{t+1} = (1-\alpha_t)Q_t + \alpha_t[P_tQ_t] \tag{9}$$

converges to Q_* with probability 1.

Note that P_t in Lemma 8 is a pseudo-contraction operator because a "true" contraction operator should map every two points in the space closer to each other, which means $\| P_tQ - P_t\hat{Q} \| \le \gamma \| Q - \hat{Q} \|$ for all $Q, \hat{Q} \in \mathbb{Q}$. Even when $\lambda_t = 0$, P_t is still not a contraction operator because it only maps every $Q \in \mathbb{Q}$ closer to Q_*, not mapping any two points in $\mathbb{Q}$ closer.

For an n-player stochastic game, we define the operator P_t as follows.

Definition 9 *Let $Q = (Q^1,\dots,Q^n)$, where $Q^k \in \mathbb{Q}^k$ for $k = 1,\dots,n$, and $\mathbb{Q} = \mathbb{Q}^1 \times \cdots \times \mathbb{Q}^n$. $P_t : \mathbb{Q} \to \mathbb{Q}$ is a mapping on the complete metric space $\mathbb{Q}$ into $\mathbb{Q}$, $P_tQ = (P_tQ^1,\dots,P_tQ^n)$, where*

$$P_tQ^k(s,a^1,\dots,a^n) = r_t^k(s,a^1,\dots,a^n) + \beta\pi^1(s')\dots\pi^n(s')Q^k(s'), \text{ for } k = 1,\dots,n,$$

where s' is the state at time $t+1$, and $(\pi^1(s'),\dots,\pi^n(s'))$ is a Nash equilibrium solution for the stage game $(Q^1(s'),\dots,Q^n(s'))$.

We prove the equation $Q_* = E[P_t Q_*]$ in Lemma 11, which depends on the following theorem.

Lemma 10 (Filar and Vrieze (1997), Theorem 4.6.5) *The following assertions are equivalent:*

1. $(\pi_*^1, \ldots, \pi_*^n)$ *is a Nash equilibrium point in a discounted stochastic game with equilibrium payoff* $\Big(v^1(\pi_*^1, \ldots, \pi_*^n), \ldots, v^n(\pi_*^1, \ldots, \pi_*^n)\Big)$, *where* $v^k(\pi_*^1, \ldots, \pi_*^n) = \Big(v^k(s^1, \pi_*^1, \ldots, \pi_*^n), \ldots, v^k(s^m, \pi_*^1, \ldots, \pi_*^n)\Big)$, $k = 1, \ldots, n$.

2. *For each* $s \in S$, *the tuple* $\big(\pi_*^1(s), \ldots, \pi_*^n(s)\big)$ *constitutes a Nash equilibrium point in the stage game* $\Big(Q_*^1(s), \ldots, Q_*^n(s)\Big)$ *with Nash equilibrium payoffs* $\Big(v^1(s, \pi_*^1, \ldots, \pi_*^n), \ldots, v^n(s, \pi_*^1, \ldots, \pi_*^n)\Big)$, *where for* $k = 1, \ldots, n$,

$$Q_*^k(s, a^1, \ldots, a^n) = r^k(s, a^1, \ldots, a^n) + \beta \sum_{s' \in S} p(s'|s, a^1, \ldots, a^n) v^k(s', \pi_*^1, \ldots, \pi_*^n). \tag{10}$$

This lemma links agent k's optimal value v^k in the entire stochastic game to its Nash equilibrium payoff in the stage game $(Q_*^1(s), \ldots, Q_*^n(s))$. In other words, $v^k(s) = \pi^1(s) \ldots \pi^n(s)$
$Q_*^k(s)$. This relationship leads to the following lemma.

Lemma 11 *For an n-player stochastic game,* $E[P_t Q_*] = Q_*$, *where* $Q_* = (Q_*^1, \ldots, Q_*^n)$.

Proof. By Lemma 10, given that $v^k(s', \pi_*^1, \ldots, \pi_*^n)$ is agent k's Nash equilibrium payoff for the stage game $(Q_*^1(s'), \ldots, Q_*^n(s')$, and $(\pi_*^1(s), \ldots, \pi_*^n(s))$ is its Nash equilibrium point, we have $v^k(s', \pi_*^1, \ldots, \pi_*^n) = \pi_*^1(s') \cdots \pi_*^n(s') Q_*^k(s')$. Based on Equation (10) we have

$$\begin{aligned}
& Q_*^k(s, a^1, \ldots, a^n) \\
= \;& r^k(s, a^1, \ldots, a^n) + \beta \sum_{s' \in S} p(s'|s, a^1, \ldots, a^n) \pi_*^1(s') \cdots \pi_*^n(s') Q_*^k(s') \\
= \;& \sum_{s' \in S} p(s'|s, a^1, \ldots, a^n) \Big(r^k(s, a^1, \ldots, a^n) + \beta \pi_*^1(s') \cdots \pi_*^n(s') Q_*^k(s') \Big) \\
= \;& E[P_t^k Q_*^k(s, a^1, \ldots, a^n)],
\end{aligned}$$

for all s, $a^1, \ldots, a^n$. Thus $Q_*^k = E[P_t Q_*^k]$. Since this holds for all k, $E[P_t Q_*] = Q_*$. □

Now the only task left is to show that the P_t operator is a pseudo-contraction operator. We prove a stronger version than required by Lemma 8. Our P_t satisfies $\| P_t Q - P_t \hat{Q} \| \leq \beta \| Q - \hat{Q} \|$ for all $Q, \hat{Q} \in \mathbb{Q}$. In other words, P_t is a *real* contraction operator. In order for this condition to hold, we have to restrict the domain of the Q-functions during learning. Our restrictions focus on stage games with special types of Nash equilibrium points: *global optima*, and *saddles*.

Definition 12 *A joint strategy* $(\sigma^1, \ldots, \sigma^n)$ *of the stage game* $(M^1, \ldots, M^n)$ *is a* global optimal point *if every agent receives its highest payoff at this point. That is, for all k,*

$$\sigma M^k \geq \hat{\sigma} M^k \quad \textit{for all } \hat{\sigma} \in \sigma(A).$$

A global optimal point is always a Nash equilibrium. It is easy to show that all global optima have equal values.

Game 1	*Left*	*Right*
Up	10, 9	0, 3
Down	3, 0	-1, 2

Game 2	*Left*	*Right*
Up	5, 5	0, 6
Down	6, 0	2, 2

Game 3	*Left*	*Right*
Up	10, 9	0, 3
Down	3, 0	2, 2

Figure 1: Examples of different types of stage games

Definition 13 *A joint strategy* $(\sigma^1, \ldots, \sigma^n)$ *of the stage game* $(M^1, \ldots, M^n)$ *is a* saddle point *if (1) it is a Nash equilibrium, and (2) each agent would receive a higher payoff when at least one of the other agents deviates. That is, for all k,*

$$\begin{aligned} \sigma^k \sigma^{-k} M^k &\geq \hat{\sigma}^k \sigma^{-k} M^k \quad \textit{for all } \hat{\sigma}^k \in \sigma(A^k), \\ \sigma^k \sigma^{-k} M^k &\leq \sigma^k \hat{\sigma}^{-k} M^k \quad \textit{for all } \hat{\sigma}^{-k} \in \sigma(A^{-k}). \end{aligned}$$

All saddle points of a stage game are equivalent in their values.

Lemma 14 *Let* $\sigma = (\sigma^1, \ldots, \sigma^n)$ *and* $\delta = (\delta^1, \ldots, \delta^n)$ *be saddle points of the n-player stage game* $(M^1, \ldots, M^n)$*. Then for all k,* $\sigma M^k = \delta M^k$*.*

Proof. By definition of a saddle point, for every $k, k = 1, \ldots, n$,

$$\begin{aligned} \sigma^k \sigma^{-k} M^k &\geq \delta^k \sigma^{-k} M^k, \quad \text{and} \qquad (11) \\ \delta^k \delta^{-k} M^k &\leq \delta^k \sigma^{-k} M^k. \qquad (12) \end{aligned}$$

Combining (11) and (12), we have

$$\sigma^k\ \sigma^{-k} M^k \geq \delta^k \delta^{-k} M^k. \tag{13}$$

Similarly, by the definition of a saddle point we can prove that

$$\delta^k \delta^{-k} M^k \geq \sigma^k \sigma^{-k} M^k. \tag{14}$$

By (13) and (14), the only consistent solution is

$$\delta^k\ \delta^{-k} M^k = \sigma^k \sigma^{-k} M^k.$$

□

Examples of stage games that have a global optimal point or a saddle point are shown in Figure 1. In each stage game, player 1 has two action choices: *Up* and *Down*. Player 2's action choices are *Left* and *Right*. Player 1's payoffs are the first numbers in each cell, with the second number denoting player 2's payoff. The first game has only one Nash equilibrium, with values (10, 9), which is a global optimal point. The second game also has a unique Nash equilibrium, in this case a saddle point, valued at (2, 2). The third game has two Nash equilibria: a global optimum, (10, 9), and a saddle, (2, 2).

Our convergence proof requires that the stage games encountered during learning have global optima, or alternatively, that they all have saddle points. Moreover, it mandates that the learner consistently choose either global optima or saddle points in updating its Q-values.

Assumption 3 *One of the following conditions holds during learning.*[3]

Condition A. *Every stage game $(Q_t^1(s),\ldots,Q_t^n(s))$, for all t and s, has a global optimal point, and agents' payoffs in this equilibrium are used to update their Q-functions.*

Condition B. *Every stage game $(Q_t^1(s),\ldots,Q_t^n(s))$, for all t and s, has a saddle point, and agents' payoffs in this equilibrium are used to update their Q-functions.*

We further define the distance between two Q-functions.

Definition 15 *For $Q,\hat{Q}\in\mathbb{Q}$, define*

$$\begin{aligned} \| Q-\hat{Q} \| &\equiv \max_j \max_s \| Q^j(s)-\hat{Q}^j(s) \|_{(j,s)} \\ &\equiv \max_j \max_s \max_{a^1,\ldots,a^n} |Q^j(s,a^1,\ldots,a^n)-\hat{Q}^j(s,a^1,\ldots,a^n)|. \end{aligned}$$

Given Assumption 3, we can establish that P_t is a contraction mapping operator.

Lemma 16 $\| P_tQ-P_t\hat{Q} \| \leq \beta \| Q-\hat{Q} \|$ *for all $Q,\hat{Q}\in\mathbb{Q}$.*

Proof.

$$\begin{aligned} \| P_tQ-P_t\hat{Q} \| &= \max_j \| P_tQ^j-P_t\hat{Q}^j \|_{(j)} \\ &= \max_j \max_s \mid \beta\pi^1(s)\cdots\pi^n(s)Q^j(s)-\beta\hat{\pi}^1(s)\cdots\hat{\pi}^n(s)\hat{Q}^j(s) \mid \\ &= \max_j \beta \mid \pi^1(s)\cdots\pi^n(s)Q^j(s)-\hat{\pi}^1(s)\cdots\hat{\pi}^n(s)\hat{Q}^j(s) \mid \end{aligned}$$

We proceed to prove that

$$|\pi^1(s)\cdots\pi^n(s)Q^j(s)-\hat{\pi}^1(s)\cdots\hat{\pi}^n(s)\hat{Q}^j(s)| \leq \| Q^j(s)-\hat{Q}^j(s) \| .$$

To simplify notation, we use σ^j to represent $\pi^j(s)$, and $\hat{\sigma}^j$ to represent $\hat{\pi}^j(s)$. The proposition we want to prove is

$$|\sigma^j\sigma^{-j}Q^j(s)-\hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s)| \leq \| Q^j(s)-\hat{Q}^j(s) \| .$$

Case 1: Suppose both $(\sigma^1,\ldots,\sigma^n)$ and $(\hat{\sigma}^1,\ldots,\hat{\sigma}^n)$ satisfy Condition A in Assumption 3, which means they are global optimal points.

If $\sigma^j\sigma^{-j}Q^j(s) \geq \hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s)$, we have

$$\begin{aligned} &\sigma^j\sigma^{-j}Q^j(s)-\hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s) \\ &\quad\leq \sigma^j\sigma^{-j}Q^j(s)-\sigma^j\sigma^{-j}\hat{Q}^j(s) \\ &\quad= \sum_{a^1,\ldots,a^n} \sigma^1(a^1)\cdots\sigma^n(a^n)\big(Q^j(s,a^1,\ldots,a^n)-\hat{Q}^j(s,a^1,\ldots,a^n)\big) \\ &\quad\leq \sum_{a^1,\ldots,a^n} \sigma^1(a^1)\cdots\sigma^n(a^n) \| Q^j(s)-\hat{Q}^j(s) \| \qquad (15) \\ &\quad= \| Q^j(s)-\hat{Q}^j(s) \|, \end{aligned}$$

[3] In our statement of this assumption in previous writings (Hu and Wellman, 1998, Hu, 1999), we neglected to include the qualification that the *same* condition be satisfied by all stage games. We have made the qualification more explicit subsequently (Hu and Wellman, 2000). As Bowling (2000) has observed, the distinction is essential.

The second inequality (15) derives from the fact that $\| Q^j(s) - \hat{Q}^j(s) \| = \max_{a^1,\ldots,a^n} |Q^j(s, a^1,\ldots,a^n) - \hat{Q}^j(s,a^1,\ldots,a^n)|$.

If $\sigma^j\sigma^{-j}Q^j(s) \leq \hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s)$, then

$$\hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s) - \sigma^j\sigma^{-j}Q^j(s) \leq \hat{\sigma}^j\hat{\sigma}^{-j}Q^j(s) - \hat{\sigma}^j\hat{\sigma}^{-j}Q^j(s),$$

and the rest of the proof is similar to the above.

Case 2: Suppose both Nash equilibria satisfy Condition B of Assumption 3, meaning they are saddle points.

If $\sigma^j\sigma^{-j}Q^j(s) \geq \hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s)$, we have

$$\begin{aligned}
\sigma^j\sigma^{-j}Q^j(s) - \hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s) &\leq \sigma^j\sigma^{-j}Q^j(s) - \sigma^j\hat{\sigma}^{-j}\hat{Q}^j(s) && (16)\\
&\leq \sigma^j\hat{\sigma}^{-j}Q^j(s) - \sigma^j\hat{\sigma}^{-j}\hat{Q}^j(s) && (17)\\
&\leq \| Q^j(s) - \hat{Q}^j(s) \|,
\end{aligned}$$

The first inequality (16) derives from the Nash equilibrium property of $\pi^1_*(s)$. Inequality (17) derives from condition B of Assumption 3.

If $\sigma^j\sigma^{-j}Q^j(s) \leq \hat{\sigma}^j\hat{\sigma}^{-j}\hat{Q}^j(s)$, a similar proof applies.

Thus

$$\begin{aligned}
\| P_tQ - P_t\hat{Q} \| &\leq \max_j \max_s \beta|\pi^1(s)\cdots\pi^n(s)Q^j(s) - \hat{\pi}^1(s)\cdots\hat{\pi}^n(s)\hat{Q}^j(s)|\\
&\leq \max_j \max_s \beta \| Q^j(s) - \hat{Q}^j(s) \|\\
&= \beta \| Q - \hat{Q} \| .
\end{aligned}$$

□

We can now present our main result: that the process induced by NashQ updates in (8) converges to Nash Q-values.

Theorem 17 *Under Assumptions 1–3, the sequence $Q_t = (Q^1_t, \ldots, Q^n_t)$, updated by*

$$Q^k_{t+1}(s,a^1,\ldots,a^n) = (1-\alpha_t)Q^k_t(s,a^1,\ldots,a^n) + \quad \alpha_t\left(r^k_t + \beta\pi^1(s')\cdots\pi^n(s')Q^k_t(s')\right) \quad \textit{for } k = 1,\ldots,n,$$

where $(\pi^1(s'),\ldots,\pi^n(s'))$ is the appropriate type of Nash equilibrium solution for the stage game $(Q^1_t(s'),\ldots,Q^n_t(s'))$, converges to the Nash Q-value $Q_* = (Q^1_*,\ldots,Q^n_*)$.

Proof. Our proof is direct application of Lemma 8, which establishes convergence given two conditions. First, P_t is a contraction operator, by Lemma 16, which entails that it is also a pseudo-contraction operator. Second, the fixed point condition, $E(P_tQ_*) = Q_*$, is established by Lemma 11. Therefore, the following process

$$Q_{t+1} = (1-\alpha_t)Q_t + \alpha_t[P_tQ_t]$$

converges to Q_*. □

(Q^1_t, Q^2_t)	*Left*	*Right*
Up	10, 9	0, 3
Down	3, 0	-1, 2

(Q^1_*, Q^2_*)	*Left*	*Right*
Up	5, 5	0, 6
Down	6, 0	2, 2

Figure 2: Two stage games with different types of Nash equilibria

Lemma 10 establishes that a Nash solution for the converged set of Q-values corresponds to a Nash equilibrium point for the overall stochastic game. Given an equilibrium in stationary strategies (which must always exist, by Theorem 4), the corresponding Q-function will be a fixed-point of the NashQ update.

4.2 On the Conditions for Convergence

Our convergence result does not depend on the agents' action choices during learning, as long as every action and state are visited infinitely often. However, the proof crucially depends on the restriction on stage games during learning because the Nash equilibrium operator is in general not a contraction operator. Figure 2 presents an example where Assumption 3 is violated.

The stage game (Q^1_t, Q^2_t) has a global optimal point with values (10, 9). The stage game (Q^1_*, Q^2_*) has a saddle point valued at (2, 2). Then $\| Q^1_t - Q^1_* \| = \max_{a^1,a^2} |Q^1_t(a^1,a^2) - Q^1_*(a^1,a^2)| = |10-5| = 5$. $|\pi^1\pi^2 Q^1_t - \pi^1_*\pi^2_* Q^1_*| = |10-2| = 8 \geq \| Q^1_t - Q^1_* \|$. Thus P_t does not satisfy the contraction mapping property.

In general, it would be unusual for stage games during learning to maintain adherence to Assumption 3. Suppose we start with an initial stage game that satisfies the assumption, say, $Q^i_0(s,a^1,a^2) = 0$, for all s, a^1, a^2 and $i = 1, 2$. The stage game (Q^1_0, Q^2_0) has both a global optimal point and a saddle point. During learning, elements of Q^1_0 are updated asynchronously, thus the property would not be preserved for (Q^1_t, Q^2_t).

Nevertheless, in our experiments reported below, we found that convergence is not necessarily so sensitive to properties of the stage games during learning. In a game that satisfies the property at equilibrium, we found consistent convergence despite the fact that stage games during learning do not satisfy Assumption 3. This suggests that there may be some potential to relax the conditions in our convergence proof, at least for some classes of games.

5. Experiments in Grid-World Games

We test our Nash Q-learning algorithm by applying it to two grid-world games. We investigate the convergence of this algorithm as well as its performance relative to the single-agent Q-learning method.

Despite their simplicity, grid-world games possess all the key elements of dynamic games: location- or state-specific actions, qualitative transitions (agents moving around), and immediate and long-term rewards. In the study of single-agent reinforcement learning, Sutton and Barto (1998) and Mitchell (1997) employ grid games to illustrate their learning algorithms. Littman (1994) experimented with a two-player zero-sum game on a grid world.

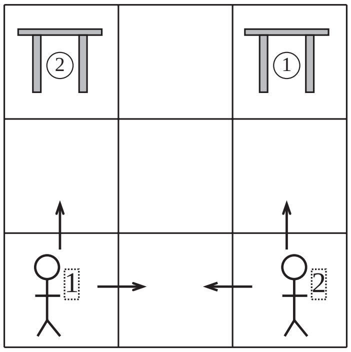

Figure 3: Grid Game 1

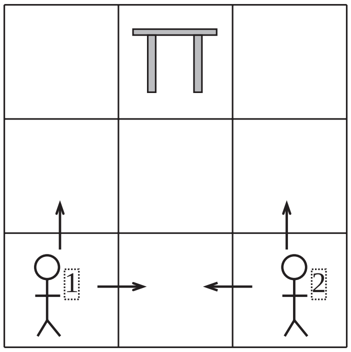

Figure 4: Grid Game 2

5.1 Two Grid-World Games

The two grid-world games we constructed are shown in Figures 3 and 4. One game has deterministic moves, and the other has probabilistic transitions. In both games, two agents start from respective lower corners, trying to reach their goal cells in the top row. An agent can move only one cell a time, and in four possible directions: *Left, Right, Up, Down*. If two agents attempt to move into the same cell (excluding a goal cell), they are bounced back to their previous cells. The game ends as soon as an agent reaches its goal. Reaching the goal earns a positive reward. In case both agents reach their goal cells at the same time, both are rewarded with positive payoffs.

The objective of an agent in this game is therefore to reach its goal with a minimum number of steps. The fact that the other agent's "winning" does not preclude our agent's winning makes agents more prone to coordination.

We assume that agents do not know the locations of their goals at the beginning of the learning period. Furthermore, agents do not know their own and the other agent's payoff functions.[4] Agents choose their actions simultaneously. They can observe the previous actions of both agents and the current state (the joint position of both agents). They also observe the immediate rewards after both agents choose their actions.

5.1.1 REPRESENTATION AS STOCHASTIC GAMES

The action space of agent i, $i = 1,2$, is $A^i = \{Left, Right, Down, Up\}$. The state space is $S = \{(0\ 1),(0\ 2),\dots,(8\ 7)\}$, where a state $s = (l^1, l^2)$ represents the agents' joint location. Agent i's location l^i is represented by position index, as shown in Figure 5.[5]

If an agent reaches the goal position, it receives a reward of 100. If it reaches another position without colliding with the other agent, its reward is zero. If it collides with the other agent, it receives -1 and both agents are bounced back to their previous positions. Let $L(l,a)$ be the potential new

[4] Note that a payoff function is a correspondence from all state-action tuples to rewards. An agent may be able to observe a particular reward, but still lack the knowledge of the overall payoff function.

[5] Given that two agents cannot occupy the same position, and excluding the cases where at least one agent is in its goal cell, the number of possible joint positions is 57 for Grid Game 1 and 56 for Grid Game 2.

6	7	8
3	4	5
0	1	2

Figure 5: Location index for the grid-world games

location resulting from choosing action a in position l. The reward function is, for $i = 1, 2$,

$$r_t^i = \begin{cases} 100 & \text{if } L(l_t^i, a_t^i) = Goal_i \\ -1 & \text{if } L(l_t^1, a_t^1) = L(l_t^2, a_t^2) \text{ and } L(l_t^2, a_t^2) \neq Goal_j,\ j = 1,2 \\ 0 & \text{otherwise.} \end{cases}$$

The state transitions are deterministic in Grid Game 1. In Grid Game 2, state transitions are deterministic except the following: if an agent chooses *Up* from position 0 or 2, it moves up with probability 0.5 and remains in its previous position with probability 0.5. Thus, when both agents choose action *Up* from state (0 2), the next state is equally likely (probability 0.25 each) to be (0 2), (3 2), (0 5), or (3 5). When agent 1 chooses *Up* and agent 2 chooses *Left* from state (0 2), the probabilities for reaching the new states are: $P((0\ 1)|(0\ 2), Up, Left) = 0.5$, and $P((3\ 1)|(0\ 2), Up, Left) = 0.5$. Similarly, we have $P((1\ 2)|(0\ 2), Right, Up) = 0.5$, and $P((1\ 5)|(0\ 2), Right, Up) = 0.5$.

5.1.2 NASH Q-VALUES

A Nash equilibrium consists of a pair of strategies (π_*^1, π_*^2) in which each strategy is a best response to the other. We limit our study to stationary strategies for the reasons discussed in Section 2.3. As defined above, a stationary strategy assigns probability distributions over an agent's actions based on the state, regardless of the history of the game. That means if the agent visits the state s at two different times, its action choice would be the same each time. A stationary strategy is generally written as $\pi = (\bar{\pi}, \bar{\pi}, \ldots)$, where $\bar{\pi} = \big(\pi(s^1), \ldots, \pi(s^m)\big)$.

Note that whenever an agent's policy depends only on its location (assuming it is sequence of pure strategies), the policy defines a *path*, that is, a sequence of locations from the starting position to the final destination. Two shortest paths that do not interfere with each other constitute a Nash equilibrium, since each path (strategy) is a best response to the other.

An example strategy for Agent 1 in Grid Game 1 is shown in Table 3. In the right column, a particular action (e.g., *Up*) is shorthand for the probability distribution assigning probability one to that action. The notation (l^1 *any*) refers to any state where the first agent is in location l^1. States that cannot be reached given the path are omitted in the table. The strategy shown represents the left path in the first graph of Figure 6. The reader can verify that this is a best response to Agent 2's path in that graph.

Figure 6 shows several other pure-strategy Nash equilibrium paths. Symmetric variants of these are also equilibria, not shown. The Nash equilibrium paths for Grid Game 2 are depicted in Figure 7.

STATE s	$\pi^1(s)$
(0 *any*)	*Up*
(3 *any*)	*Right*
(4 *any*)	*Right*
(5 *any*)	*Up*

Table 3: A stationary strategy for agent 1 in Grid Game 1

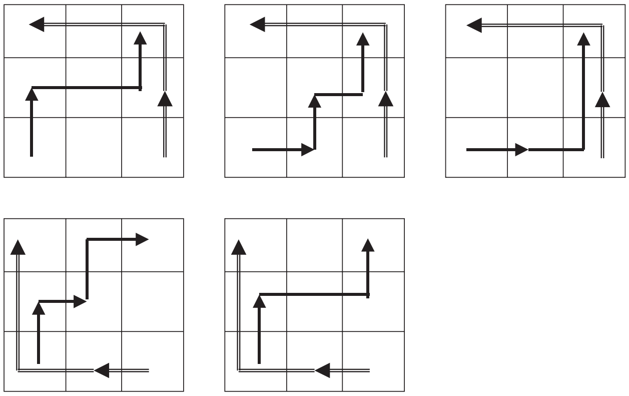

Figure 6: Selected Nash equilibrium paths, Grid Game 1

The value of the game for agent 1 is defined as its accumulated reward when both agents follow their Nash equilibrium strategies,

$$v^1(s_0) = \sum_t \beta^t E(r_t | \pi^1_*, \pi^2_*, s_0).$$

In Grid Game 1 and initial state $s_0 = (0\ 2)$, this becomes, given $\beta = 0.99$,

$$\begin{aligned} v^1(s_0) &= 0 + 0.99 \cdot 0 + 0.99^2 \cdot 0 + 0.99^3 \cdot 100 \\ &= 97.0. \end{aligned}$$

Based on the values for each state, we can then derive the Nash Q-values for agent 1 in state s_0,

$$Q^1(s_0, a^1, a^2) = r^1(s_0, a^1, a^2) + \beta \sum_{s'} p(s'|s_0, a^1, a^2) v^1(s').$$

Therefore $Q^1_*(s_0, Right, Left) = -1 + 0.99 v^1((0\,2)) = 95.1$, and $Q^1_*(s_0, Up, Up) = 0 + 0.99 v^1((3\,5)) = 97.0$.

The Nash Q-values for both agents in state (0 2) of Grid Game 1 are shown in Table 4. There are three Nash equilibria for this stage game $(Q^1(s_0), Q^2(s_0))$, and each is a global optimal point with the value (97.0, 97.0).

For Grid Game 2, we can derive the optimal values similarly for agent 1:

$$\begin{aligned} v^1((0\ 1)) &= 0 + 0.99 \cdot 0 + 0.99^2 \cdot 0 = 0, \\ v^1((0\ x)) &= 0, \text{ for } x = 3, \ldots, 8, \\ v^1((1\ 2)) &= 0 + 0.99 \cdot 100 = 99, \\ v^1((1\ 3)) &= 0 + 0.99 \cdot 100 = 99 = v^1((1\ 5)), \\ v^1((1\ x)) &= 0, \text{ for } x = 4, 6, 8. \end{aligned}$$

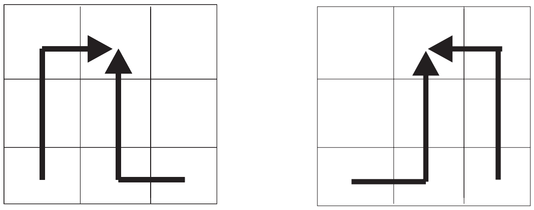

Figure 7: Nash equilibrium paths, Grid Game 2

	Left	*Up*
Right	95.1, 95.1	97.0, 97.0
Up	97.0, 97.0	97.0, 97.0

Table 4: Grid Game 1: Nash Q-values in state (0 2)

However, the value for $v^1((0\ 2)) = v^1(s_0)$ can be calculated only in expectation, because the agents have a 0.5 probability of staying in their current location if they choose *Up*. We solve $v^1(s_0)$ from the stage game $(Q^1_*(s_0), Q^2_*(s_0))$.

The Nash Q-values for agent 1 in state $s_0 = (0\ 2)$ are

$$\begin{aligned}
Q^1_*(s_0, \mathit{Right}, \mathit{Left}) &= -1 + 0.99 v^1((s_0)), \\
Q^1_*(s_0, \mathit{Right}, \mathit{Up}) &= 0 + 0.99(\frac{1}{2} v^1((1\ 2)) + \frac{1}{2} v^1((1\ 3)) = 98, \\
Q^1_*(s_0, \mathit{Up}, \mathit{Left}) &= 0 + 0.99(\frac{1}{2} v^1((0\ 1)) + \frac{1}{2} v^1((3\ 1)) = 0.99(0 + \frac{1}{2} 99) = 49, \\
Q^1_*(s_0, \mathit{Up}, \mathit{Up}) &= 0 + 0.99(\frac{1}{4} v^1((0\ 2)) + \frac{1}{4} v^1((0\ 5)) + \frac{1}{4} v^1((3\ 2)) + \frac{1}{4} v^1((3\ 5))) \\
&= 0.99(\frac{1}{4} v^1((0\ 2)) + 0 + \frac{1}{4} 99 + \frac{1}{4} 99) \\
&= 0.99 \frac{1}{4} v^1(s_0) + 49.
\end{aligned}$$

These Q-values and those of agent 2 are shown in Table 5.1.2. We define $R_i \equiv v^i(s_0)$ to be agent i's optimal value by following Nash equilibrium strategies starting from state s_0, where $s_0 = (0\ 2)$.

Given Table 5.1.2, there are two potential pure strategy Nash equilibria: (*Up*, *Left*) and (*Right*, *Up*). If (*Up*, *Left*) is the Nash equilibrium for stage game $(Q^1_*(s_0), Q^2_*(s_0))$, then $v^1(s_0) = \pi^1(s_0)\pi^2(s_0)Q^1(s_0) = 49$, and then $Q^1_*(s_0, \mathit{Right}, \mathit{Left}) = 47$. Therefore we can derive all the Q-values as shown in the first table in Figure 8. If (*Right*, *Up*) is the Nash equilibrium, $v^1(s_0) = 98$, and we can derive another set of Nash Q-values as shown in the second table in Figure 8. In addition, there exists a mixed-strategy Nash equilibrium for stage game $(Q^1_*(s_0), Q^2_*(s_0))$, which is $(\pi^1(s_0), \pi^2(s_0)) = (\{P(\mathit{Right}) = 0.97, P(\mathit{Up}) = 0.03)\}, \{P(\mathit{Left}) = 0.97, P(\mathit{Up}) = 0.03\})$, and the Nash Q-values are shown in the third table of Figure 8.[6]

As we can see, there exist three sets of different Nash equilibrium Q-values for Grid Game 2. This is essentially different from Grid Game 1, which has a unique Nash Q-value for every state-

[6] See Appendix A for a derivation.

	Left	*Up*
Right	$-1+0.99R_1, -1+0.99R_2$	98, 49
Up	49, 98	$49+\frac{1}{4}0.99R_1, 49+\frac{1}{4}0.99R_2$

Table 5: Grid Game 2: Nash Q-values in state (0 2)

	Left	*Up*
Right	47, 96	98, 49
Up	49, 98	61, 73

	Left	*Up*
Right	96, 47	98, 49
Up	49, 98	73, 61

	Left	*Up*
Right	47.87, 47.87	98, 49
Up	49, 98	61.2, 61.2

Figure 8: Three Nash Q-values for state (0 2) in Grid Game 2

action tuple. The convergence of learning becomes problematic when there are multiple Nash Q-values. Furthermore, none of the Nash equilibria of the stage game $(Q^1_*(s_0), Q^2_*(s_0))$ of Grid Game 2 is a global optimal or saddle point, whereas in Grid Game 1 each Nash equilibrium is a global optimum.

5.1.3 THE LEARNING PROCESS

A learning agent, say agent 1, initializes $Q^1(s,a^1,a^2) = 0$ and $Q^2(s,a^1,a^2) = 0$ for all s,a^1,a^2. Note that these are agent 1's internal beliefs. They have nothing to do with agent 2's beliefs. Here we are interested in how agent 1 learns the Nash Q-functions. Since learning is offline, it does not matter whether agent 1 is wrong about agent 1's actual strategies during the learning process. But agent 1 has learned the equilibrium strategies, which stipulate what both agents will do when they both know enough information about the game and both act rationally.

A game starts from the initial state (0 2). After observing the current state, agents choose their actions simultaneously. They then observe the new state, both agents' rewards, and the action taken by the other agent. The learning agent updates its Q-functions according to (8). In the new state, agents repeat the process above. When at least one agent moves into its goal position, the game restarts. In the new episode, each agent is randomly assigned a new position (except its goal cell). The learning agent keeps the Q-values learned from previous episodes. The training stops after 5000 episodes. Each episode on average takes about eight steps. So one experiment usually requires about 40,000 steps. The total number of state-action tuples in Grid Game 1 is 424. Thus each tuple is visited 95 times on average. The learning rate is defined as the inverse of the number of visits. More specifically, $\alpha_t(s,a^1,a^2) = \frac{1}{n_t(s,a^1,a^2)}$, where $n_t(s,a^1,a^2)$ is the number of times the tuple (s,a^1,a^2) has been visited. It is easy to show that this definition satisfies the conditions in Assumption 2. When $\alpha_t = \frac{1}{95} \approx 0.01$, the results from new visits hardly change the Q-values already learned.

When updating the Q-values, the agent applies a Nash equilibrium value from the stage game $(Q^1(s'), Q^2(s'))$, possibly necessitating a choice among multiple Nash equilibria. In our implemen-

	Left	*Up*
Right	-1, -1	49, 0
Up	0, 0	0, 97

Table 6: Grid Game 1: Q-values in state (0 2) after 20 episodes if always choosing the first Nash

	Left	*Up*
Right	31, 31	0, 65
Up	0, 0	49, 49

Table 7: Grid Game 2: Q-values in state (0 2) after 61 episodes if always choosing the first Nash

tation, we calculate Nash equilibria using the Lemke-Howson method (Cottle et al., 1992), which can be employed to generate equilibria in a fixed order.[7] We then define a *first-Nash* learning agent as one that updates its Q-values using the first Nash equilibrium generated. A *second-Nash* agent employs the second if there are multiple equilibria, otherwise it uses the only available one. A *best-expected-Nash* agent picks the Nash equilibrium that yields the highest expected payoff to itself.

5.2 Experimental Results

5.2.1 Q-FUNCTIONS DURING LEARNING

Examination of Q-values during learning reveals that both grid games violate Assumption 3. Table 6 shows the results for first-Nash agents playing Grid Game 1 from state (0 2) after certain learning episodes. The only Nash equilibrium in this stage game, (*Right*, *Up*), is neither a global optimum nor a saddle point. In Grid Game 2, we find a similar violation of Assumption 3. The unique Nash equilibrium (*Up*, *Up*) in the stage game shown in Table 7 is neither a global optimum nor a saddle point.

5.2.2 CONVERGENCE RESULTS: THE FINAL Q-FUNCTIONS

After 5000 episodes of training, we find that agents' Q-values stabilize at certain values. Some learning results are reported in Table 8 and 9. Note that a learning agent always uses the same Nash equilibrium value to update its own Q-function and that of the other agent.

We can see that the results in Table 8 are close to the theoretical derivation in Table 4, and the results in Table 9 are close to the theoretical derivation in the first table of Figure 8. Although it is impossible to validate theoretical asymptotic convergence with a finite trial, our examples confirm that the learned Q-functions possess the same equilibrium strategies as Q_*.

For each state s, we derive a Nash equilibrium $(\pi^1(s), \pi^2(s))$ from the stage game comprising the learned Q-functions, $(Q^1(s), Q^2(s))$. We then compare this solution to a Nash equilibrium derived from theory. Results from our experiments are shown in Table 10. In Grid Game 1, our learning reaches a Nash equilibrium 100% of the time (out of 50 runs) regardless of the updating choice during learning. In Grid Game 2, however, learning does not always converge to a Nash equilibrium joint strategy.

[7] The Lemke-Howson algorithm is quite effective in practice (despite exponential worst-case behavior), but limited to two-player (bimatrix) games.

	Left	*Up*
Right	86, 87	83, 85
Up	96, 91	95, 95

Table 8: Final Q-values in state (0 2) if choosing the first Nash in Grid Game 1

		Left	*Up*
Agent 1	*Right*	39, 84	97, 51
	Up	46, 93	59, 74

Table 9: Final Q-values in state (0 2) if choosing the first Nash in Grid Game 2

5.3 Offline Learning Performance

In offline learning, examples (training data) for improving the learned function are gathered in the training period, and performance is measured in the test period. In our experiments, we employ a training period of 5000 episodes. The performance of the test period is measured by the reward an agent receives when both agents follow their learned strategies, starting from the initial positions at the lower corners.

Since learning is internal to each agent, two agents might learn different sets of Q-functions, which yield different Nash equilibrium solutions. For example, agent 2 might learn a Nash equilibrium corresponding to the joint path in the last graph of Figure 6. In that case, agent 2 will always choose *Left* in state (0 2). Agent 1 might learn the third graph of Figure 6, and therefore always choose *Right* in state (0 2). In this case agent 1's strategy is not a best response to agent 2's strategy.

We implement four types of learning agents: first Nash, second Nash, best-expected Nash, and *single*–the single-agent Q-learning method specified in (4).

The experimental results for Grid Game 1 are shown in Table 11. For each case, we ran 50 trials and calculated the fraction that reach an equilibrium joint path. As we can see from the table, when both agents employ single-agent Q-learning, they reach a Nash equilibrium only 20% of the time. This is not surprising since the single-agent learner never models the other agent's strategic attitudes. When one agent is a Nash agent and the other is a single-agent learner, the chance of reaching a Nash equilibrium increases to 62%. When both agents are Nash agents, but use different update selection rules, they end up with a Nash equilibrium 80% of the time.[8] Finally, when both agents are Nash learners and use the same updating rule, they end up with a Nash equilibrium solution 100% of the time.

The experimental results for Grid Game 2 are shown in Table 12. As we can see from the table, when both agents are single-agent learners, they reach a Nash equilibrium 50% of the time. When one agent is a Nash learner, the chance of reaching a Nash equilibrium increases to 51.3% on average. When both agents are Nash learners, but use different updating rules, they end up with a Nash equilibrium 55% of the time on average. Finally, 79% of trials with two Nash learners using the same updating rule end up with a Nash equilibrium.

Note that during learning, agents choose their action randomly, based on a uniform distribution. Though the agent's own policy during learning does not affect theoretical convergence in the limit

[8] Note that when both agents employ best-expected Nash, they may choose different Nash equilibria. A Nash equilibrium solution giving the best expected payoff to one agent may not lead to the best expected payoff for another agent. Thus, we classify trials with two best-expected Nash learners in the category of those using different update selection rules.

	Equilibrium selection	Percentage of runs that reach a Nash equilibrium
Grid Game 1	First Nash	100%
	Second Nash	100%
Grid Game 2	First Nash	68%
	Second Nash	90%

Table 10: Convergence Results

LEARNING STRATEGY		RESULTS OF LEARNING
AGENT 1	AGENT 2	PERCENT THAT REACH A NASH EQUILIBRIUM
SINGLE	SINGLE	20%
SINGLE	FIRST NASH	60%
	SECOND NASH	50%
	BEST EXPECTED NASH	76%
FIRST NASH	SECOND NASH	60%
	BEST EXPECTED NASH	76%
SECOND NASH	BEST EXPECTED NASH	84%
BEST EXPECTED NASH	BEST EXPECTED NASH	100%
FIRST NASH	FIRST NASH	100%
SECOND NASH	SECOND NASH	100%

Table 11: Learning performance in Grid Game 1

(as long as every state and action are visited infinitely often), the policy of other agents does influence the single-agent Q-learner, as its perceived environment includes the other agents implicitly. For methods that explicitly consider joint actions, such as NashQ, the joint policy affects only the path to convergence.

For Grid Game 2, the single-single outcome can be explained by noting that given its counterpart is playing randomly, the agent has two optimal policies: entering the center and following the wall. To see their equivalence, note that if agent 1 chooses *Right*, it will successfully move with probability 0.5 because agent 2 will choose *Left* with equal probability. If agent 1 chooses *Up*, it will successfully move with probability 0.5 because this is the defined transition probability. The value of achieving either location are the same, and therefore the two initial actions have the same value. Given that the agents learn that there are two equal-valued strategies, there is a 0.5 probability that they will happen to choose the complementary ones, hence the result.

Alternative training regimens, in which agents' policies are updated during learning, should be expected to improve the single-agent Q-learning results. Indeed, both Bowling and Veloso (2002) and Greenwald and Hall (2003) found this to be the case in Grid Game 2 and other environments,

5.4 Online Learning Performance

In online learning, an agent is evaluated based on rewards accrued while it learns. Whereas action choices in the training periods of offline learning are important only with respect to gaining

LEARNING STRATEGY		RESULTS OF LEARNING
AGENT 1	AGENT 2	PERCENT THAT REACH A NASH EQUILIBRIUM
SINGLE	SINGLE	50%
SINGLE	FIRST NASH	54%
	SECOND NASH	62%
	BEST EXPECTED NASH	38%
FIRST NASH	SECOND NASH	64%
	BEST EXPECTED NASH	78%
SECOND NASH	BEST EXPECTED NASH	36%
BEST EXPECTED NASH	BEST EXPECTED NASH	42%
FIRST NASH	FIRST NASH	68%
SECOND NASH	SECOND NASH	90%

Table 12: Learning performance in Grid Game 2

information, in online learning the agent must weigh value for learning (*exploration*) against direct performance value (*exploitation*).

The Q-learning literature recounts many studies on the balance between exploration and exploitation. In our investigation, we adopt an ε-Greedy exploration strategy, as described, for example, by Singh et al. (2000). In this strategy, the agent explores with probability $\varepsilon_t(s)$ and chooses the optimal action with probability $1-\varepsilon_t(s)$. Singh et al. (2000) proved that when $\varepsilon_t(s) = c/n_t(s)$ with $0 < c < 1$, the learning policy satisfies the GLIE (Greedy in the Limit with Infinite Exploration) property.

We test agent 1's performance under three different strategies: *exploit*, *explore*, and *exploit-and-explore*. In the exploit strategy, an agent always chooses the Nash equilibrium learned so far if the state has been visited at least once ($n_t(s) \geq 1$). If $n_t(s) = 0$, the agent will choose an action randomly. In the explore strategy, an agent chooses an action randomly. In the exploit-and-explore strategy, the agent chooses the Nash equilibrium action with probability $\varepsilon_t(s) = \frac{1}{1+n_t(s)}$ and chooses a random action with probability $1-\varepsilon_t(s)$.

Our results are shown in Figure 9. The experiments show that when the other agent plays exploit, the exploit-and-explore strategy gives agent 1 higher payoff than the exploit strategy. When agent 2 is an explore or exploit-and-explore agent, agent 1's payoff from exploit is very close to that from exploit-and-explore, though the latter is still a little higher. One explanation for this is that the exploit strategy is very easily trapped in local optima. When one agent uses the exploit-and-explore strategy, it helps that agent to find an action that is not in conflict with another agent, so that both agents will benefit.

6. Related Work

Littman (1994) designed a Minimax-Q learning algorithm for zero-sum stochastic games. A convergence proof for that algorithm was provided subsequently by Littman and Szepesvári (1996). Claus and Boutilier (1998) implemented a joint action learner, incorporating other agents' actions into its Q-function, for a repeated coordination game. That paper was first presented in a AAAI-97 workshop, and motivated our own work on incorporating joint actions into Q-functions in the

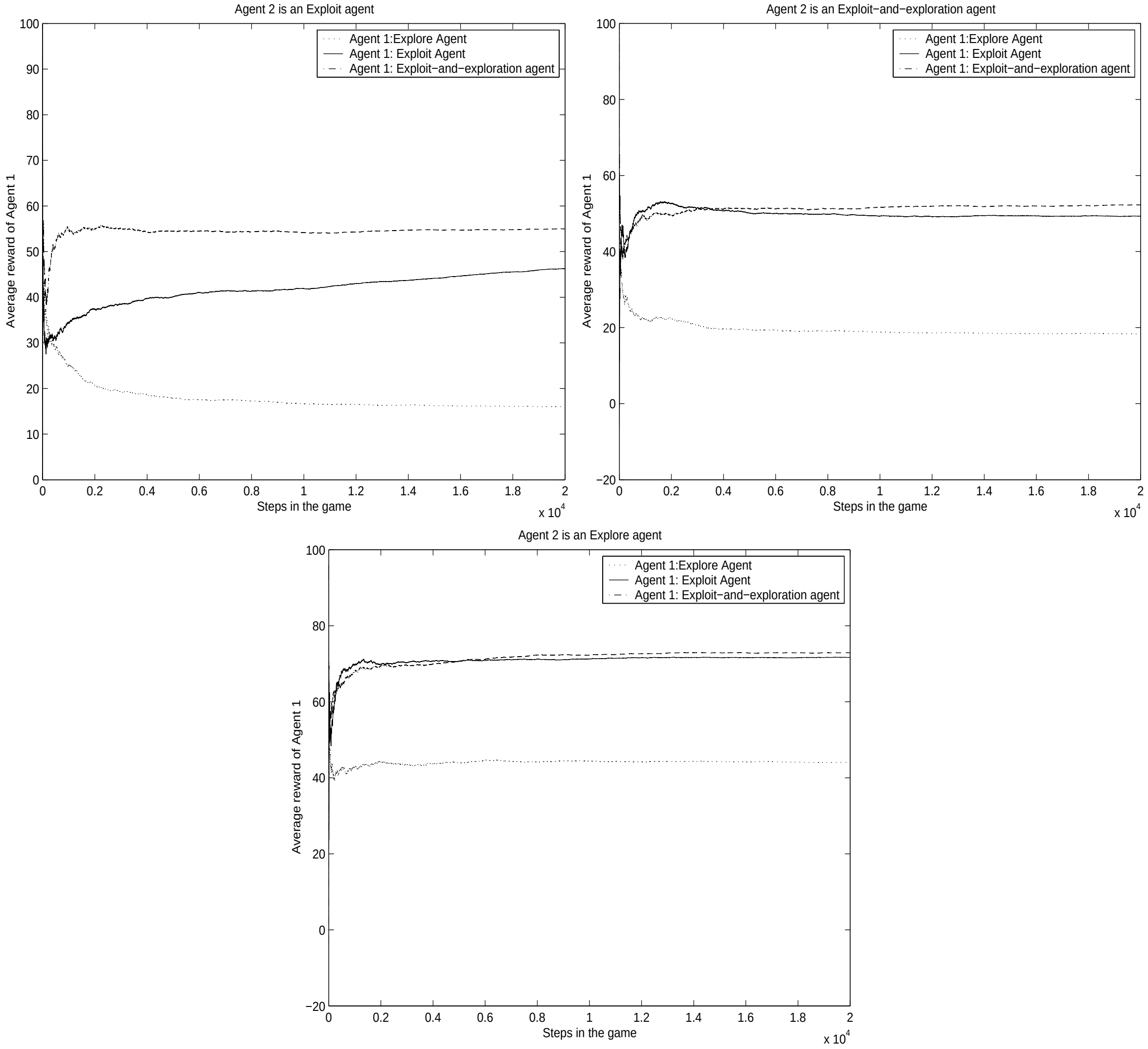

Figure 9: Online performance of agent 1 in three different environments

general-sum context. Our Nash Q-learning algorithm (Hu and Wellman, 1998) was the subject of further clarification from Bowling (2000) and illustrations from Littman (2001b).

The Friend-or-Foe Q-learning (FFQ) algorithm (Littman, 2001a) overcomes the need for our assumptions about stage games during learning in special cases where the stochastic game is known to be a coordination game or zero-sum. In coordination games—where there is perfect correlation between agents' payoffs—an agent employs Friend-Q, which selects actions to maximize an agent's own payoff. In zero-sum games—characterized by perfect *negative* correlation between agents' payoffs—an agent uses Foe-Q, which is another name for Minimax-Q. In both cases, the correlations reduce the agents' learning problem to that of learning its own Q-function. Littman shows that FFQ converges generally, and to equilibrium solutions when the friend or foe attributions are correct. His analysis of our two grid-world games (Section 5) further illuminates the relation between NashQ and FFQ.

Sridharan and Tesauro (2000) have investigated the behavior of single-agent Q-learners in a dynamic pricing game. Bowling and Veloso (2002) present a variant of single-agent Q-learning, where rather than adopt the policy implicit in the Q-table, the agent performs hill-climbing in the space of probability distributions over actions. This enables the agent to maintain a mixed strategy, while learning a best response to the other agents. Empirical results (including an experiment with our Grid Game 2) suggest that the rate of policy hill climbing should depend on the estimated quality of the current policy.

Researchers continue to refine Q-learning for zero-sum stochastic games. Recent developments include the work by Brafman and Tennenholtz (2000, 2001), who developed a polynomial-time algorithm that attains near-optimal average reward in zero-sum stochastic games. Banerjee et al. (2001) designed a Minimax-SARSA algorithm for zero-sum stochastic games, and presented evidence of faster convergence than Minimax-Q.

One of the drawbacks of Q-learning is that each state-action tuple has to be visited infinitely often. Q(λ) learning (Peng and Williams, 1996, Wiering and Schmidhuber, 1998) promises an interesting way to speed up the learning process. To apply Q-learning online, we seek a general criterion for setting the exploitation rate, which would maximize the agent's expected payoff. Meuleau and Bourgine (1999) studied such a criterion in single-agent multi-state environments. Chalkiadakis and Boutilier (2003) proposed using Bayesian beliefs about other agents's strategies to guide the calculation. Since fully Bayesian updating is computationally demanding, in practice they update strategies based on fictitious play.

This entire line of work has raised questions about the fundamental goals of research in multiagent reinforcement learning. Shoham et al. (2003) take issue with the focus on convergence toward equilibrium, and propose four alternative, well-defined problems that multiagent learning might plausibly address.

7. Conclusions

We have presented NashQ, a multiagent Q-learning method in the framework of general-sum stochastic games. NashQ generalizes single-agent Q-learning to multiagent environments by updating its Q-function based on the presumption that agents choose Nash-equilibrium actions. Given some highly restrictive assumptions on the form of stage games during learning, the method is guaranteed to converge. Empirical evaluation on a pair of small but interesting grid games shows that the method can often find equilibria despite the violations of our theoretical assumptions. In particular, in a game possessing multiple equilibria with identical values, the method converges to equilibrium with relatively high frequency. In a second game with a variety of equilibrium Q-values, the observed likelihood of reaching an equilibrium is reduced. In both cases, however, employing NashQ improves the prospect for reaching equilibrium over single-agent Q-learning, at least in the offline learning mode.

Although NashQ is defined for the general-sum case, the conditions for guaranteed convergence to equilibrium do not cover a correspondingly general class of environments. As Littman (2001a,b) has observed, the technical conditions are actually limited to cases—coordination and adversarial equilibria—for which simpler methods are sufficient. Moreover, the formal condition (Assumption 3) underlying the convergence theorem is defined in terms of the stage games *as perceived during learning*, so cannot be evaluated in terms of the actual game being learned. Of what value, then, are the more generally cast algorithm and result? We believe there are two main contributions of the analysis presented herein.

First, it provides a starting point for further theoretical work on multiagent Q-learning. Certainly there is much middle ground between the extreme points of pure coordination and pure zero-sum, and much of this ground will also be usefully cast as special cases of Nash equilibrium.[9] The NashQ analysis thereby provides a known sufficient condition for convergence, which may be relaxed in particular circumstances by taking advantage of other known structure in the game or equilibrium-selection procedure.

Second, at present there is *no* multiagent learning method offering directly applicable performance guarantees for general-sum stochastic games. NashQ is based on intuitive generalization of both the single-agent and minimax methods, and remains a plausible candidate for the large body of environments for which no known method is guaranteed to work well. One might expect it to perform best in games with unique or concentrated equilibria, though further study is required to support strong conclusions about its relative merits for particular classes of multiagent learning problems.

Perhaps more promising than NashQ itself are the many conceivable extensions and variants that have already begun to appear in the literature. For example, the "extended optimal response" approach (Suematsu and Hayashi, 2002) maintains Q-tables for all agents, and anticipates the other agents' actions based on a balance of their presumed optimal and observed behaviors. Another interesting direction is reflected in the work of Greenwald and Hall (2003), who propose a version of multiagent Q-learning that employs *correlated equilibrium* in place of the Nash operator applied by NashQ. This appears to offer certain computational and convergence advantages, while requiring some collaboration in the learning process itself.

Other equilibrium concepts—such as any of the many Nash "refinements" defined by game theorists—specify by immediate analogy a variety of multiagent Q-learning. It is our hope that an understanding of NashQ can assist search through all the plausible variants and hybrids, toward the ultimate goal of designing effective learning algorithms for dynamic multiagent domains.

Appendix A. A Mixed Strategy Nash Equilibrium for Grid Game 2

We are interested in determining if there exists a mixed strategy Nash equilibrium for Grid Game 2. Let $R_i \equiv v^i(s_0)$ be agent i's optimal value by following Nash equilibrium strategies starting from state s_0, where $s_0 = (0\ 2)$. The two agents' Nash Q-values for taking different joint actions in state s_0 are shown in Table 5.1.2. The Nash Q-value is defined as the sum of discounted rewards of taking a joint action at current state (in state s_0) and then following the Nash equilibrium strategies thereafter.

Let $(p, 1-p)$ be agent 1's probabilities of taking action *Right* and *Up*, and $(q, 1-q)$ be agent 2's probabilities of taking action *Left* and *Up*. Then agent 1's problem is

$$\begin{array}{ll} \max\limits_{p} & p[q(-1+0.99R_1)+(1-q)98]+(1-p)[49q+(1-q)(49+\frac{1}{4}0.99R_1] \\ s.t. & \qquad\qquad\qquad\qquad p \geq 0 \end{array}$$

[9] Indeed, the classes of games that *possess*, respectively, coordination or adversarial equilibria, are already far relaxed from the pure coordination and zero-sum games, which were the focus of original studies establishing behaviors of the equivalents of Friend-Q and Foe-Q.

	Left	*Up*
Right	47.87, 47.87	98, 49
Up	49, 98	61.2, 61.2

Table 13: Nash Q-values in state (0 2)

Let μ be the Kuhn-Tucker multiplier on the constraint, so that the Lagrangian takes the form:

$$L = p[0.99R_1q + 98 - 99q] + (1-p)[49 + 0.2475R_1 - 0.2475R_1q] - \mu p$$

The maximization condition requires that $\dfrac{\partial L}{\partial p} = 0$. Therefore we get

$$0.99R_1q + 98 - 99q - 49 - 0.2475R_1 + 0.2475R_1q = \mu. \tag{18}$$

Since we are interested in mixed strategy solution where $p > 0$, the slackness condition implies that $\mu = 0$. Based on equation (18), we get

$$R_1 = \frac{99q - 49}{1.2375q - 0.2475}. \tag{19}$$

By symmetry in agent 1 and 2's payoffs, we also have

$$R_2 = \frac{99p - 49}{1.2375p - 0.2475}.$$

By definition, $R_1 = v^1(s_0)$. According to Lemma 10, $v^1(s_0) = \pi^1(s_0)\pi^2(s_0)Q^1(s_0)$. Since $\pi^1(s_0) = (p \;\; 1-p)$ and $\pi^2(s_0) = (q \;\; 1-q)$, therefore

$$\begin{aligned} R_1 &= (p \;\; 1-p)\begin{pmatrix} -1 + 0.99R_1 & 98 \\ 49 & 49 + \frac{1}{4}0.99R_1 \end{pmatrix}\begin{pmatrix} q \\ 1-q \end{pmatrix} \\ &= pq(-1 + 0.99R_1) + (1-q)p98 + 49q(1-p) + (1-q)(1-p)(49 + 0.2475R_1). \end{aligned}$$

From the above equation and (19), we have

$$-24.745 + 1.7325q + 24.5025q^2 = 0,$$

and solving yields $q = 0.97$.

By symmetry in the two agents' payoff matrices, we have $p = 0.97$.

Therefore we have $R_1 = R_2 = \frac{99 \times 0.97 - 49}{1.2375 \times 0.97 - 0.2475} = 49.36$. We can then rewrite Table 5.1.2 as Table 13.

Acknowledgments

We thank Craig Boutilier, Michael Littman, Csaba Szepesvári, Jerzy Filar, and Olvi Mangasarian for helpful discussions. The anonymous referees provided invaluable criticism and suggestions.

References

Tucker Balch. Learning roles: Behavioral diversity in robot teams. In Sandip Sen, editor, *Collected Papers from the AAAI-97 Workshop on Multiagent Learning*. AAAI Press, 1997.

Cesar Bandera, Francisco J. Vico, Jose M. Bravo, Mance E. Harmon, and Leemon C. Baird. Residual Q-learning applied to visual attention. In *Thirteenth International Conference on Machine Learning*, pages 20–27, Bari, Italy, 1996.

Bikramjit Banerjee, Sandip Sen, and Jing Peng. Fast concurrent reinforcement learners. In *Seventeenth International Joint Conference on Artificial Intelligence*, pages 825–830, Seattle, 2001.

Craig Boutilier. Sequential optimality and coordination in multiagent systems. In *Sixteenth International Joint Conference on Artificial Intelligence*, pages 478–485, Stockholm, 1999.

Michael Bowling. Convergence problems of general-sum multiagent reinforcement learning. In *Seventeenth International Conference on Machine Learning*, pages 89–94, Stanford, 2000.

Michael Bowling and Manuela Veloso. Multiagent learning using a variable learning rate. *Artificial Intelligence*, 136:215–250, 2002.

Ronen I. Brafman and Moshe Tennenholtz. A near-optimal polynomial time algorithm for learning in certain classes of stochastic games. *Artificial Intelligence*, 121(1-2):31–47, 2000.

Ronen I. Brafman and Moshe Tennenholtz. A general polynomial time algorithm for near-optimal reinforcement learning. In *Seventeenth International Joint Conference on Artificial Intelligence*, pages 953–958, Seattle, 2001.

Georgios Chalkiadakis and Craig Boutilier. Coordination in multiagent reinforcement learning: a bayesian approach. In *The Proceedings of the Second International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003)*, pages 709–716, 2003.

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

Richard W. Cottle, J.-S. Pang, and R. E. Stone. *The Linear Complementarity Problem*. Academic Press, New York, 1992.

Edwin De Jong. Non-random exploration bonuses for online reinforcement learning. In Sandip Sen, editor, *Collected Papers from the AAAI-97 Workshop on Multiagent Learning*. AAAI Press, 1997.

Jerzy Filar and Koos Vrieze. *Competitive Markov Decision Processes*. Springer-Verlag, 1997.

A. M. Fink. Equilibrium in a stochastic *n*-person game. *Journal of Science in Hiroshima University, Series A-I,* 28:89–93, 1964.

Drew Fudenberg and David K. Levine. *The Theory of Learning in Games*. The MIT Press, 1998.

Amy Greenwald and Keith Hall. Correlated Q-learning. In *Proceedings of the Twentieth International Conference on Machine Learning (ICML-2003)*, pages 242–249, Washington DC, 2003.

Stephan H. G. ten Hagen. *Continuous State Space Q-Learning for Control of Nonlinear Systems*. PhD thesis, University of Amsterdam, February 2001. http://carol.wins.uva.nl/˜stephanh/phdthesis/.

John C. Harsanyi and Reinhard Selten. *A General Theory of Equilibrium Selection in Games*. The MIT Press, 1988.

Junling Hu. *Learning in Dynamic Noncooperative Multiagent Systems*. PhD thesis, University of Michigan, Ann Arbor, Michigan, July 1999.

Junling Hu and Michael P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *Fifteenth International Conference on Machine Learning*, pages 242–250, Madison, WI, 1998.

Junling Hu and Michael P. Wellman. Experimental results on Q-learning for general-sum stochastic games. In *Seventeenth International Conference on Machine Learning*, pages 407–414, Stanford, 2000.

Rufus Isaacs. *Differential Games: A Mathematical Theory with Applications to Warfare and Pursuit, Control and Optimization*. R. E. Krieger Pub. Co., 1975.

Leslie Kaelbling, Michael L. Littman, and Andrew W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996.

Michael Kearns, Michael L. Littman, and Satinder Singh. Graphical models for game theory. In *Seventeenth Conference on Uncertainty in Artificial Intelligence*, pages 253–260, Seattle, 2001.

Jeffrey O. Kephart and Gerald J. Tesauro. Pseudo-convergent Q-learning by competitive pricebots. In *Seventeenth International Conference on Machine Learning*, pages 463–470, Stanford, 2000.

Daphne Koller and B. Milch. Multi-agent influence diagrams for representing and solving games. In *Seventeenth International Joint Conference on Artificial Intelligence*, pages 1027–1034, Seattle, Washington, 2001.

Michael L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *Eleventh International Conference on Machine Learning*, pages 157–163. New Brunswick, 1994.

Michael L. Littman. Friend-or-foe Q-learning in general-sum games. In *Eighteenth International Conference on Machine Learning*, pages 322–328, Williams College, MA, 2001a.

Michael L. Littman. Value-function reinforcement learning in Markov games. *Cognitive Systems Research*, 2:55–66, 2001b.

Michael L. Littman and Csaba Szepesvári. A generalized reinforcement-learning model: Convergence and applications. In *Thirteenth International Conference on Machine Learning*, pages 310–318, 1996.

Richard D. McKelvey and Andrew McLennan. Computation of equilibria in finite games. In *Handbook of Computational Economics*, volume 1. Elsevier, 1996.

Nicolas Meuleau and Paul Bourgine. Exploration of multi-state environments: Local measures and back-propagation of uncertainty. *Machine Learning*, 35(2):117–154, 1999.

Tom Mitchell. *Machine Learning*, pages 367–390. McGraw-Hill, 1997.

John F. Nash. Non-cooperative games. *Annals of Mathematics*, 54:286–295, 1951.

Norihiko Ono and Kenji Fukumoto. Mulit-agent reinforcement learning: A modular approach. In *Second International Conference on Multiagent Systems*, pages 252–258, Kyoto, 1996.

Martin J. Osborne and Ariel Rubinstein. *A Course in Game Theory*. MIT Press, 1994.

Jing Peng and Ronald Williams. Incremental multi-step Q-learning. *Machine Learning*, 22:283–290, 1996.

Leon A. Petrosjan and Nikolay A. Zenkevich. *Game Theory*. World Scientific, Singapore, 1996.

Martin L. Puterman. *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. John Wiley & Sons, New York, 1994.

Yoav Shoham, Rob Powers, and Trond Grenager. Multi-agent reinforcement learning: A critical survey. Technical report, Stanford University, 2003.

Satinder Singh and Dimitri Bertsekas. Reinforcement learning for dynamic channel allocation in cellular telephone systems. In *Advances in Neural Information Processing Systems*, volume 9, pages 974–980. MIT Press, 1996.

Satinder Singh, Tommi Jaakkola, Michael L. Littman, and Csaba Szepesvári. Convergence results for single-step on-policy reinforcement learning algorithms. *Machine Learning*, 38(3):287–308, 2000.

Manu Sridharan and Gerald Tesauro. Multi-agent Q-learning and regression trees for automated pricing decisions. In *Seventeenth International Conference on Machine Learning*, pages 927–934, Stanford, 2000.

Peter Stone and Richard S. Sutton. Scaling reinforcement learning toward RoboCup soccer. In *Proc. 18th International Conf. on Machine Learning*, pages 537–544. Morgan Kaufmann, San Francisco, CA, 2001.

Nobuo Suematsu and Akira Hayashi. A multiagent reinforcement learning algorithm using extended optimal response. In *First International Joint Conference on Autonomous Agents and Multiagent Systems*, pages 370–377, Bologna, 2002.

Richard S. Sutton and Andrew G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

Csaba Szepesvári and Michael L. Littman. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Computation*, 11:8:2017–2059, 1999.

Ming Tan. Multi-agent reinforcement learning: Independent vs. cooperative agents. In *Tenth International Conference on Machine Learning*, pages 330–337, Amherst, MA, 1993.

Frank Thusijsman. *Optimality and Equilibria in Stochastic Games*. Centrum voor Wiskunde en Informatica, Amsterdam, 1992.

Marilyn A. Walker. An application of reinforcement learning to dialogue strategy selection in a spoken dialogue system for email. *Journal of Artificial Intelligence Research*, 12:387–416, 2000.

Christopher J. C. H. Watkins. *Learning from Delayed Rewards*. PhD thesis, King's College, Cambridge, England, 1989.

Christopher J. C. H. Watkins and Peter Dayan. Q-learning. *Machine Learning*, 3:279–292, 1992.

Michael P. Wellman and Junling Hu. Conjectural equilibrium in multiagent learning. *Machine Learning*, 33:179–200, 1998.

Marco Wiering and Jurgen Schmidhuber. Fast online Q(λ). *Machine Learning*, 33:105–115, 1998.