A Reinforcement Learning (Nash-R) Algorithm for Average Reward Irreducible Stochastic Games

Jun Li
Jun.Li@warnerbros.com
Kandethody Ramachandran
ram@cas.usf.edu
Department of Mathematics and Statistics
University of South Florida
Tapas K. Das *
das@eng.usf.edu
Department of Industrial and Management Systems Engineering
University of South Florida
4202 E. Fowler Avenue
Tampa, Florida 33620, USA

Editor: Leslie Pack Kaelbling

Abstract

A large class of sequential decision making problems under uncertainty with multiple competing decision makers can be modeled as stochastic games. It can be considered that the stochastic games are multiplayer extensions of Markov decision processes (MDPs). In this paper, we develop a reinforcement learning algorithm to obtain average reward equilibrium for irreducible stochastic games. In our approach, we first define a sequence of auxiliary matrix games having equivalent equilibrium points and values to the above mentioned stochastic games. We then extend the theory of RL algorithms for average reward MDPs to learn the elements of the matrix games. A convergence analysis of the algorithm is developed from the study of the asymptotic behavior of its two time scale stochastic approximation scheme and the stability of the associated ordinary differential equations (ODEs). The learning algorithm is tested using a grid world game.

Keywords: Stochastic game, matrix game, reinforcement learning, stochastic approximation, ODE limit

1. Introduction

Many industrial decision making problems such as inventory management, supply chain management, and airline yield management are inherently sequential. In these sequential problems, based on the observed system state, a decision maker chooses an action. The action results in two outcomes: an immediate reward, and a new state at the next decision epoch. An action is then chosen for the new state, and thus the system continues. Markov/semi-Markov decision process (MDP/SMDP) Puterman (1994), Das et al. (1999) models are used to study these noncompetitive sequential decision making problems, if their underlying stochastic models are Markov chains.

Stochastic games are extensions of the noncompetitive decision problems, and consist of multiple competing decision makers (also referred to as agents or players). The collective actions of the players dictate the next system state and also the individual rewards of the decision makers. A stochastic game is dynamic in the sense that the decision environment is nonstationary to each decision maker. The nonstationarity arises from the changes in the behavior of the other decision makers with time. Also, an inherent aspect of a noncooperative game is the conflict of interest among the players. Hence, making decisions in a noncooperative game environment is challenging. Game theory provides a framework to analyze the conflicts of all agents' interest Rasmusen (2001), Myerson (1997). In this paper we first combine knowledge from the theory of MDPs and the game theory to show how average reward irreducible stochastic games can be represented as a series of equivalent matrix games. Since solution methodologies exist for matrix games Mckelvey et al., Nanduri and Das (2007), the above equivalence enables us to solve stochastic games. Later in the paper, we develop a two time scale reinforcement learning algorithm for obtaining the equivalent matrix games and provide a convergence analysis. Methods used for solving the resulting matrix games are not explicitly discussed in this paper.

Stochastic games have received some attention in recent years from mathematicians, computer scientists, and engineers Shapley (1997), Filar and Vrieze (1997), Altman and Spieksma (1997), Altman and Pourtallier (2000). Many dynamic programming and reinforcement learning based techniques Borkar (2002), Littman (2001, 1994), Brafman and Tennenholtz (2000), Hu and Wellman (1998), Bowling and Veloso (2000), Ravulapati et al. (2004), Rao et al. (2002) have been presented as solution methods for certain classes of games. Based on the reward criterion, stochastic games can generally be divided into discounted reward stochastic games and average reward stochastic games. Most of the above referenced work studied discounted reward games, since the discount factor that applies to many real life applications allows analytical advantages over the average reward criterion. But when decisions are made frequently or the reward criterion do not need to be described in economic terms, the average reward criterion is often more appropriate. In this paper we focus our attention on the average reward criterion.

Discounted reward stochastic games are defined as games without nonzero stopping probabilities. In such games, where future payoffs are discounted using a discounting factor β Filar and Vrieze (1997), the payoffs eventually approach zero. This can be construed as the termination of the game. Hence the discounted reward stochastic games can be viewed as transient stochastic games with stopping probability $1 - \beta$ independent of the actions taken. For finite state space and bounded rewards, the discounted rewards are continuous in the strategies. So every discounted reward stochastic game possesses at least one equilibrium point. The fact that average rewards may not be continuous in the strategies makes the average reward stochastic games more challenging. However, for some special categories of games, such as irreducible games, the average reward is continuous in the strategies. Our focus here is on finite state space and bounded reward irreducible stochastic games with the expected average reward optimality criterion in the infinite horizon. We present for these games the mathematical structure of their Nash equilibrium policies, which is critical to developing learning based solution algorithms.

In what follows, we first present an outline of the theory of MDPs and the game theory, in order to establish notation and the theoretical foundation for stochastic games. We then present, in Section 4, discounted stochastic games and define a sequence of auxiliary matrix games that are shown to be equivalent. In Section 5, the average reward stochastic games are defined along with the associated matrix games that are shown to be equivalent. In the following section, we develop a Nash-R learning algorithm for iteratively obtaining the elements of the equivalent matrices for average reward irreducible stochastic games. Section 6 also contains the convergence analysis for the Nash-R algorithm using a stochastic approximation method with two-time scales. In Section 7, the learning algorithm is tested and benchmarked against other learning approaches using a grid-world game.

2. Single Decision Maker Problems–Markov Decision Processes

In sequential decision making problems, the decision maker's goal is to choose a sequence of actions that maximizes the utility of the system based on a given criterion. Such problems with an underlying Markovian structure can be defined within the framework of MDPs (Puterman (1994), Filar and Vrieze (1997), Abounadi et al. (2002), Gosavi (2004)) denoted by a tuple $< S, A, P, R >$. The elements of the tuple are as follows.
S denotes the finite set of states of the environment.
A denotes the finite set of actions available to the decision maker (agent), with $A(s)$ denoting the subset of actions available at state s.
P denotes the set of all transition probability matrices $P^a,\ a \in A$. An element $p(s'|s,a)$ of P^a gives the one step probability of reaching state s', when action a is taken in state s.
$R : S \times A \to \mathbb{R}$ is the reward function that gives the immediate reward corresponding to every action in each state. An element $r(s,a)$ of R denotes the reward for action a in state s.

For MDPs, the consequences of the decisions are uncertain but the environment is stationary. A probability distribution of consequences are associated with each alternative action, and this distribution does not change over time once the decision policy is given. For a given policy, a reward is received in each period. The agent's job is to find a policy π, mapping states to actions, that maximizes some measure of the rewards received. The MDPs are either finite-horizon models or infinite-horizon models. The infinite-horizon *discounted reward* model takes the long-run reward of the agent into account, but rewards that are received in the future are geometrically discounted according to discount factor β (where $0 \leq \beta < 1$). The expected discounted reward for an infinite-horizon MDP starting in state $s \in S$ can be given as $V_\beta(s) = E(\sum_{t=0}^{\infty} \beta^t r_t)$, where r_t is the reward received at the t^{th} decision epoch. Another optimality criterion is long-run expected *average-reward*, which can be given by $g(s) = \limsup_{T\to\infty} E(\frac{1}{T}\sum_{t=0}^{T-1} r_t)$. The average reward g is also referred to as the gain of this system. A deterministic policy is one that assigns a probability value of 1 to an action in each state. Every MDP has a deterministic stationary optimal policy Puterman (1994). For unichain average reward MDPs where there is only one recurrent class of states, and possibly a set of transient states, the gain is identical for all the states.

Under this unichain assumption, *limsup* is a limit for any stationary policy (Abounadi et al. (2001)). Henceforth, we will use the *limit* notation instead of *limsup*. Since in average reward MDPs the sum of the rewards can be unbounded, a surrogate used is called *bias h*, which is defined for a starting state $s \in S$ and for stationary policies as:

$$h(s) = \lim_{T \to \infty} E_s \sum_{t=0}^{T-1} [r_t - g(s)].$$

The bias is interpreted as the expected total difference between the reward and the stationary reward.

The optimality equation for discounted reward MDPs is given as:

$$V_\beta(s) = \max_{a \in A(s)} \left(r(s,a) + \beta \sum_{s' \in S} p(s'|s,a) V_\beta(s') \right). \tag{1}$$

The average reward of unichain MDPs can be seen as the limiting discounted reward with discount factor β approaching 1. Using truncated Laurent series expansion of the discounted value function (1) (refer to pp314-355 for details Puterman (1994)), we can write the value function in vector form for the average reward MDP in terms of gain (or, average reward) and bias as follows:

$$g + (I - P)h = r. \tag{2}$$

From (2), the optimality equation for a unichain average reward MDP may be expressed in component notation as

$$0 = \max_{a \in A(s)} \left(r(s,a) - g + \sum_{s' \in S} p(s'|s,a) h(s') - h(s) \right).$$

The above equation can be rewritten in Bellman's optimality equation form as

$$h^*(s) + g^* = \max_{a \in A(s)} \left(r(s,a) + \sum_{s'} p(s'|s,a) h^*(s') \right). \tag{3}$$

If the reward functions and the transition probability functions are known, dynamic programming algorithms such as value iteration or policy iteration can be used to find the optimal policy. In real life, for many stochastic decision problems, the transition probability matrices and the reward functions are not easily available. A common approach used is to simulate the system and learn the model first. This approach is called model-based learning. The framework of dynamic Bayesian networks (DBNs) can be used to describe a certain class of MDPs in a compact way Ghahramani (1998). A more common approach is model-free learning, which is suitable for solving problems with very large state spaces. In recent years, reinforcement learning methods have shown to yield optimal or near-optimal solutions to large MDPs. Two classes of asynchronous model-free learning algorithms for discounted reward and average reward models are Q-learning and R-learning respectively Kaelbling et al. (1996), Tsitsiklis (1994), Mahadevan (1996). In the next section, we give a description of stochastic games.

3. Stochastic Games

Stochastic games have two or more decision makers whose interests are coupled since their state transition probabilities and/or rewards are coupled. Stochastic games having Markovian state transition properties are also called Markov games or competitive Markov decision processes (CMDP) Filar and Vrieze (1997), which are an extension of MDP to noncooperative scenario. We consider finite state/action noncooperative games, where all decision makers make their decision independently and noncooperatively to maximize their individual payoff criterion.

A stochastic game can be defined by a tuple $< n, S, A^1, ..., A^n, P, \tilde{R}^1, ..., \tilde{R}^n >$, the elements of which are as follows.

n denotes the number of agents/players/decision makers.

S denotes the finite set of states of the environment.

$A^1, ..., A^n$ denote the collection of finite set of admissible actions to the agents $1, ..., n$, where $A^i = \{A^i(s) : s \in S\}$ and $m^i(s) = \left|A^i(s)\right|, i = 1, ..., n$, are the cardinalities of the action spaces in state s. Let $\mathbf{A} = A^1 \times ... \times A^n$. To play the game, each agent chooses an action $a^i \in A^i$ resulting in a joint action vector $\mathbf{a} = (a^1, ..., a^n)$. Let $\mathbf{a}^{-i} = (a^1, ..., a^{i-1}, a^{i+1}, ..., a^n)$ be vector of actions of all players except the agent i. We will use this vector notation whenever there is no confusion. Same notation will be used later for mixed strategies.

P denotes the set of transition probability matrices, where $p(s' \mid s, \mathbf{a})$ is the transition probability of reaching state s' as a result of a joint action $\mathbf{a}$ by all of the n players.

$r^k : S\times \mathbf{A} \to \mathbb{R}$ gives the immediate reward gained by the player k for each set of actions of each agents in each state. $\tilde{R}^k$ is the reward matrix of player k. Throughout the paper we assume that $p(s' \mid s, \mathbf{a})$ and $r^k(s, \mathbf{a})$ are continuous in $\mathbf{a}$.

In a stochastic game, the transition probabilities and the reward functions depend on the choices made by all agents. Thus, from the perspective of an agent, the game environment is nonstationary during its evolution phase. However, for irreducible stochastic games, optimal strategies constitute stationary policies and hence it is sufficient to consider only the stationary strategies Filar and Vrieze (1997). We define $\pi^k(s)$ as the mixed strategy at state s for agent i, which is the probability distribution over available action set, $A^k(s)$, of player k. Thus $\pi^k(s) = \{\pi^k(s, a) : a \in A^k(s)\}$, where $\pi^k(s, a)$ denotes the probability of player k choosing action a in state s, and $\sum_{a\in A^k(s)} \pi^k(s, a) = 1$. Then $\pi = (\pi^1, ..., \pi^n)$ denotes a joint mixed strategy, also called a policy. A pure action $a \in A^k(s, a)$ can be treated as a mixed strategy π^k for which $\pi^k(a) = 1$.

Under policy π, the transition probability can be given as

$$p(s' \mid s, \pi) = \sum_{a^1=1}^{m^1(s)} \cdots \sum_{a^n=1}^{m^n(s)} p(s' \mid s, a^1, ..., a^n)\pi^n(s, a^n) \cdots \pi^1(s, a^1).$$

The immediate expected reward of player k induced by a mixed strategy π in a state s is given by

$$r^k(s, \pi) = \sum_{a^1=1}^{m^1(s)} ... \sum_{a^n=1}^{m^n(s)} r^k(s, a^1, ..., a^n)\pi^n(s, a^n)...\pi^1(s, a^1).$$

Then the overall discounted value of a policy π to player k starting in state s can be given as

$$V_{\beta}^{k}(s,\pi)=\sum_{t=0}^{\infty}\beta^{t}E_{s}(r_{t}^{k})=\sum_{t=0}^{\infty}\beta^{t}\sum_{s'\in S}p^{t}(s'\mid s,\pi)r^{k}(s',\pi),$$

where $p^t(.)$ denotes an element of the t^{th} power of the transition probability matrix P. The overall average value of a policy π to player k starting in state s can be given as

$$V_{\alpha}^{k}(s,\pi)=T\rightarrow\infty\limsup\frac{1}{T}\sum_{t=0}^{T-1}E_{s}(r_{t}^{k})=T\rightarrow\infty\limsup\frac{1}{T}\sum_{t=0}^{T-1}p^{t}(s'\mid s,\pi)r^{k}(s',\pi).$$

We will assume the following unichain condition for the average reward stochastic games.

Assumption A1: (Ergodicity) If there is at least one player that uses average reward criterion, then unichain ergodic structure holds. That is, under any stationary policy π, the process is irreducible Markov chain with one ergodic class.
Assumption A1 will imply that *limsup* is a *limit* under any stationary policy Abounadi et al. (2001). Thus, hereon, we will use *limit* instead of *limsup*.

3.1 Matrix Games

A matrix game can be defined by a tuple $< n, A^1, \ldots, A^n, \tilde{R}^1, \ldots, \tilde{R}^n >$. The elements of the tuple are as follows.
n denotes the number of players.
A^k denotes the set of actions available to player k.
$r^k : A^1 \times \ldots \times A^n \rightarrow \mathbb{R}$ is the payoff function for player k, where an element $r^k(a^1, \ldots, a^n)$ is the payoff to player k when the players choose actions a^1 through a^n.
$\tilde{R}^k$ for all k, can be written as an n-dimensional matrix as follows

$$\tilde{R}^{k}=\Big[r^{k}(a^{1},a^{2},\cdots,a^{n})\Big]_{a^{1}=1,\ldots,a^{n}=1}^{a^{1}=m^{1}(s),\ldots,a^{n}=m^{n}(s)}.$$

The players select actions from the set of available actions with the goal of maximizing their payoffs which depends on all the players' actions. The concept of *Nash equilibrium* is used to describe the strategy as being the most rational behavior by the players acting to maximize their payoffs. So for a matrix game, a pure strategy Nash equilibrium is an action profile $(a_*^1, \cdots, a_*^n)$, for which $r^k(a_*^k, a_*^{-k}) \geq r^k(a^k, a_*^{-k})$, $\forall a^k \in A^k$, and $k = 1, 2, \cdots, n$. The equilibrium values denoted by $Val[\cdot]$ for player k with payoff matrices $\tilde{R}^k$ is obtained as $Val[\tilde{R}^k] = r^k(a_*^1, \cdots, a_*^n)$. The appealing feature of the Nash equilibrium is that any unilateral deviation from it by any player is not worthwhile. A mixed strategy Nash equilibrium for matrix games is a vector $(\pi_*^1, \cdots, \pi_*^n)$, for which we can write

$$\sum_{a^{1}=1}^{m^{1}(s)}\cdots\sum_{a^{n}=1}^{m^{n}(s)}\pi_{*}^{k}(a^{k})\pi_{*}^{-k}(a^{-k})r^{k}(a^{k},a^{-k})\geq\sum_{a^{1}=1}^{m^{1}(s)}\cdots\sum_{a^{n}=1}^{m^{n}(s)}\pi^{k}(a^{k})\pi_{*}^{-k}(a^{-k})r^{k}(a^{k},a^{-k})$$

where $\pi_*^{-k}(a^{-k}) = \pi_*^1(a^1)\cdots\pi_*^{k-1}(a^{k-1}).\pi_*^{k+1}(a^{k+1})\cdots\pi_*^n(a^n)$.

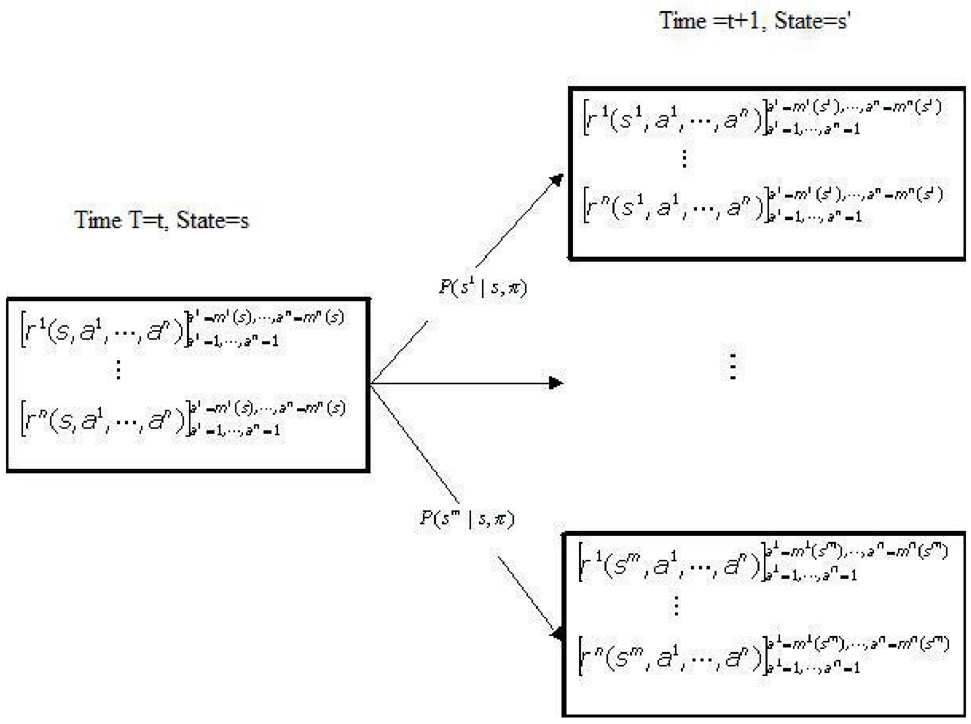
Time =t+1, State=s'
Time T=t, State=s
$P(s^1 \mid s, \pi)$
$P(s^m \mid s, \pi)$
$[r^1(s, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s),\cdots,a^n=m^n(s)}$
$[r^n(s, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s),\cdots,a^n=m^n(s)}$
$[r^1(s^1, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s^1),\cdots,a^n=m^n(s^1)}$
$[r^n(s^1, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s^1),\cdots,a^n=m^n(s^1)}$
$[r^1(s^m, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s^m),\cdots,a^n=m^n(s^m)}$
$[r^n(s^m, a^1, \cdots, a^n)]_{a^1=1,\cdots,a^n=1}^{a^1=m^1(s^m),\cdots,a^n=m^n(s^m)}$

Figure 1: Stochastic game as multi-state matrix game

A matrix game may not have a pure strategy Nash equilibrium, but it always has a mixed strategy Nash equilibrium Nash (1951). There exist methods for solving Nash equilibrium of finite nonzero-sum matrix games McKelvey and McLennan (1996), Wilson (1971), Mckelvey et al.. Since in matrix games, there are no transition probability functions, matrix games are static. Also matrix games can be viewed as recursive stochastic games with a single state. On the other hand, stochastic games can be viewed as extensions of matrix games from a single state to a multi-state environment, as depicted in Figure 1. This viewpoint is utilized in this paper.

As shown in the Figure 1, when system time $T = t$, the system is at state s, and $\tilde{R}^k$ is the immediate payoff matrix for player k. Thus, each state s has associate with it an n-dimensional matrix game. If the players choose strategy π at time t, then at time $T = t + 1$, the system evolves to state $s' \in \{s^1, \ldots, s^m\}$ according to the transition probability $p(s'|s, \pi)$, where m is the total number of states. For each of the new possible states, there is a corresponding matrix game. So a stochastic game evolves through a series of matrix games, which are connected by transition probabilities. But one can not just solve for the immediate payoffs of the matrix games separately to get the equilibrium strategies for the states, since in addition to the immediate payoffs the opportunities in future states must also be considered.

In the next section, to motivate the matrix game based abstraction of stochastic games, we describe the discounted reward stochastic games and its connection to matrix games.

4. Discounted Reward Stochastic Games

Due to the economic meaning of the discount factor and due to the mathematical convenience it provides to bound the infinite sum, discounted stochastic games have been studied extensively. Most of what is presented in this subsection are from Hu and Wellman (2003), which presents a multi-player extension of the two-player games in Filar and Vrieze (1997).

The discounted stochastic games are defined as games in which future payoffs are discounted by a discount factor β. The total discounted reward is given by

$$V_\beta^k(\pi) = \sum_{t=0}^{\infty} \beta^t P^t(\pi) r^k(\pi). \tag{4}$$

The strategy π_* denotes a Nash equilibrium point for discounted stochastic game if $V_\beta^k(\pi_*) \geq V_\beta^k(\pi_*^{-k}, \pi^k)$ for all players $k = 1, ...n$. The discounted reward given in (4) can be rewritten in component notation in terms of expected immediate reward and the expected discounted value of the next state as follows

$$V_\beta^k(s, \pi) = r^k(s, \pi) + \beta \sum_{s' \in S} p(s' \mid s, \pi) V_\beta^k(s', \pi), \tag{5}$$

from which the definition of Nash equilibrium can be expanded to

$$\begin{aligned} r^k(s, \pi_*) + \beta \sum_{s' \in S} p &(s' \mid s, \pi) V_\beta^k(s', \pi_*) \\ &\geq r^k(s, \pi_*^{-k}, \pi^k) + \beta \sum_{s' \in S} p(s' \mid s, \pi_*^{-k}, \pi^k) V_\beta^k(s', \pi_*^{-k}, \pi^k). \end{aligned} \tag{6}$$

Directly solving for Nash equilibrium using the inequality (6) is difficult, even when the reward functions and transition probabilities are available. However, since methods are available to solve for Nash equilibrium of matrix games, researchers have attempted to study discounted reward stochastic games using their connection to matrix games Hu and Wellman (2003).

Filar and Vrieze Filar and Vrieze (1997) combined the theories of discounted Markov decision processes and Matrix games to develop an auxiliary bi-matrix game for two player discounted stochastic games. In what follows, we extend the above technique to n-player games and construct n-dimensional equivalent auxiliary matrices $Q^k(.)$ for all players $k = 1, ..., n$.

The elements of the $Q^k(.)$ matrices are payoffs for all possible pure action sets $\mathbf{a}$, which take into account both the immediate reward and the future opportunities. For $s \in S$, the matrix with size $m^1(s) \times m^2(s) \times ... \times m^n(s)$ for the k^{th} player can be given by

$$Q^k(s) = \left[r^k(s, a^1, ..., a^n) + \beta \sum_{s' \in S} p(s' \mid s, a^1, ..., a^n) V_\beta^k(s', \pi_*) \right]_{a^1=1,...,a^n=1}^{a^1=m^1(s),...,a^n=m^n(s)} \tag{7}$$

where $V^k_\beta(s', \pi_*)$ is the equilibrium value for the stochastic game starting at state s' for player k. Note that this auxiliary matrix, $Q^k(.)$ captures the information from the matrix game resulting from the pure strategies as well as the equilibrium payoff of the stochastic game. This enables us to establish a connection between the matrix games and stochastic games.

If the players use a mixed strategy π, the value of the above matrix game for player k can be obtained as:

$$\begin{aligned}
Val[Q^k(s), \pi) &= \sum_{a^1=1}^{m^1(s)} ... \sum_{a^n=1}^{m^n(s)} \pi^1(s, a^1)...\pi^n(s, a^n) Q^k(s, a^1, ..., a^n) \\
&= \sum_{a^1=1}^{m^1(s)} ... \sum_{a^n=1}^{m^n(s)} \pi^1(s, a^1)...\pi^n(s, a^n) \{ r^k(s, a^1, ..., a^n) + \\
&\quad \beta \sum_{s' \in S} p(s' \mid s, a^1, ..., a^n) V^k_\beta(s', \pi_*) \\
&= r^k(s, \pi) + \beta \sum_{s' \in S} p(s' \mid s, \pi) V^k_\beta(s', \pi_*).
\end{aligned}$$

For the matrix game (7), the equilibrium point π_* is defined such that

$$Val[Q^k(s), \pi_*] \geq Val[Q^k(s), \pi_*^{-k}, \pi^k]. \tag{8}$$

which can be expanded to

$$\begin{aligned}
r^k(s, \pi_*) + \beta & \sum_{s' \in S} p(s' \mid s, \pi_*).V^k_\beta(s', \pi_*) \geq \\
& r^k(s, \pi_*^{-k}, \pi^k) + \beta \sum_{s' \in S} p(s' \mid s, \pi_*^{-k}, \pi^k) V^k_\beta(s', \pi_*).
\end{aligned} \tag{9}$$

The difference between the definitions of Nash equilibria for discounted reward stochastic game (6) and auxiliary game (10) is subtle. The only difference is that the last term in (6) is $V^k_\beta(s', \pi_*^{-k}, \pi^k)$, where as the corresponding term in (10) is $V^k_\beta(s', \pi_*)$. The following result establishes the connection between the matrix games and the discounted reward stochastic games.

Theorem 1 *The following are equivalent:*
(i) π_* is an equilibrium point in the discounted reward stochastic game with equilibrium payoffs $(V^1_\beta(\pi_*), ..., V^n_\beta(\pi_*))$.
(ii) For each $s \in S$, the strategy $\pi_*(s)$ constitutes an equilibrium point in the static n-dimentional matrix game $(Q^1(s), ..., Q^n(s))$ with equilibrium payoffs $(Val[Q^1(s), \pi_*], \cdots, Val[Q^n(s), \pi_*])$. The entry of $Q^k(s)$ corresponding to actions $\mathbf{a} = (a^1, \cdots, a^n)$ is given by

$$Q^k(s, \mathbf{a}) = r^i(s, \mathbf{a}) + \beta \sum_{s' \in S} p(s' \mid s, \mathbf{a}) V^i_\beta(s', \pi_*), \textit{ for } i = 1, ..., n, \textit{ where } \mathbf{a} \in \prod_{i=1}^{n} A^i(s).$$

Proof: If (i) is true, it means that π_*^k is the best response of player k to the equilibrium responses of other players. Then player k is not able to improve his/her reward by deviating from the strategy π_*^k even for one step (a similar argument can be found in many cited works including Filar and Vrieze (1997), Herings and Peeters (2000). Then we write that $V_\beta^k(s, \pi_*(s)) \geq r^k(s, \pi_*^{-k}(s), \pi^k(s)) + \beta \sum_{s' \in S} p(s' \mid s, \pi) V_\beta^k(s', \pi_*(s))$. Now, using (5) and (8), $V_\beta^k(s, \pi_*(s)) = Val[Q^k(s), \pi_*]$, which implies inequality (10). So $\pi_*(s)$ is the equilibrium point of the matrix game in (ii).

If (ii) is true, then the equilibrium points satisfy the inequality (10). Also, note that by the definition of Nash equilibrium, $\mathrm{V}_\beta^k(s, \pi_*) \geq V_\beta^k(s, \pi_*^{-k}, \pi^k)$Thus, using (6) and (10)

$$\begin{aligned} Val[Q^k(s), \pi_*] &= V_\beta^k(s, \pi_*(s)) \\ &\geq r^k(s, \pi_*^{-k}, \pi^k) + \beta \sum_{s' \in S} p(s' \mid s, \pi_*^{-k}, \pi^k) V_\beta^k(s', \pi_*) \\ &\geq r^k(s, \pi_*^{-k}, \pi^k) + \beta \sum_{s' \in S} p(s' \mid s, \pi_*^{-k}, \pi^k) V_\beta^k(s', \pi_*^{-k}, \pi^k) \end{aligned}$$

So $\pi_*(s)$ is the equilibrium point of the stochastic game in (i).

Note that we can not interpret any of the elements of (7) as the expected discounted reward gained when all players play the corresponding pure action set in that state. This is due to the fact that the expected utility in stationary strategies does not hold for the class of stochastic games Herings and Peeters (2000). Once the matrices $Q^k(\cdot)$ are constructed, linear or quadratic programming methods can be used to solve for Nash equilibrium. However, to construct the static matrix games, we need to know the reward functions $r^k(\cdot)$, transition probability functions $P(\cdot)$, and the equilibrium value $V_\beta^k(\pi_*)$. We may know the reward and transition probability, but not the equilibrium value before the problem is solved. A nonlinear complementarity programming approach for solving two player matrix games is presented in Filar and Vrieze (1997).

We note that, the entries in this matrix game (7) have similar structure to the Bellman's optimality equation for discounted MDP. Well known algorithms to solve Bellman's discounted optimality equation are value iteration and policy iteration. An extension of the value iteration and redefinition of the value operator to solve stochastic games ws presented in Shapley (1997). There exist learning algorithms that attempt to learn the entries of the $Q^k(\cdot)$ matrices. The matrices are updated during each stage and are expected to converge to their optimal forms. Minmax Q-learning algorithm for discounted zero-sum games is presented in Littman (1994). A Nash Q-learning for discounted general-sum games is presented in Hu and Wellman (2003). Both Minmax Q-learning and Nash-Q learning algorithms are extensions of the model-free reinforcement Q-learning Kaelbling et al. (1996), Sutton and Barto (1998). A summary of the available stochastic game algorithms can be found in Bowling and Veloso (2000).

For MDPs, there is only one optimal value for each state. But for stochastic games, there may be several Nash equilibrium points and perhaps several equilibrium values. Theorem 1 says that each equilibrium value has a corresponding matrix game. So there can be several possible equivalent matrix games for each state of a stochastic game. But in a machine learning based approach, where each agent usually can only learn one matrix game, assuring or verifying that all the matrices learned by different players correspond to the same equilibrium value is difficult. In other words, it is possible that strategies learned by the players may not constitute a Nash equilibrium policy. For zero-sum games with unique Nash equilibrium value, such a problem does not arise. Nash-Q algorithm Hu and Wellman (1998) for general-sum games is developed only for problems having a unique Nash equilibrium value.

5. Average Reward Stochastic Games

In this section, for average reward stochastic games, we will construct auxiliary matrix games having equivalent Nash equilibria and values. In Li (2003), average reward stochastic games were explored by examining a relationship between discounted reward V_β and average reward V_α. In that work, for some special cases (e.g., irreducible games), the limiting average criterion was treated as the limit of the β-discounted criterion with $\beta \to 1$. This was accomplished by using the Hardy-Littlewood Theorem Filar and Vrieze (1997), and the Laurent series expansion approach in obtaining the average reward optimality function in terms of gain and bias from the discounted optimality function. The resulting optimality equation is as follows.

$$h^k(\pi_*) = r^k(\pi_*) - V_\alpha^k(\pi_*) + P(\pi_*)h^k(\pi_*). \tag{10}$$

We call $V_\alpha^k(\pi_*)$ the *gain equilibrium value*, and $h^k(\pi_*)$ the *bias equilibrium value*. A similar structure of equilibrium points for average reward stochastic games with countable states was also derived in Altman and Spieksma (1997). From (10), we can write

$$h^k(\pi_*) \geq r^k(\pi_*^{-k}, \pi^k) - V_\alpha^k(\pi_*) + P(\pi_*^{-k}, \pi^k)h^k(\pi_*). \tag{11}$$

For n-player average reward game, construct an n-dimentional equivalent auxiliary matrices $R^k(.)$ for all players $k = 1, ..., n$, similar to the discounted games. The elements of these matrices are payoffs for all possible pure action sets $\mathbf{a}$, which take into account both the immediate reward and the future opportunities. For $s \in S$ the matrix with size $m^1(s) \times m^2(s) \times .. \times m^n(s)$ for the k^{th} player can be given by

$$R^k(s) = \left[r^k(s, a^1, ..., a^n) - V_\alpha^k(\pi_*) + \sum_{s' \in S} p(s' \mid s, a^1, ..., a^n) h^k(s', \pi_*) \right]_{a^1=1,...,a^n=1}^{a^1=m^1(s),...,a^n=m^n(s)} \tag{12}$$

where $h^k(s', \pi_*)$ is the bias equilibrium value for the stochastic game starting at state s' for player k. For a mixed strategy π

$$\begin{aligned} Val[R^k(s), \pi] &= \sum_{a^1=1}^{m^1(s)} ... \sum_{a^n=1}^{m^n(s)} \pi^1(s, a^1)...\pi^n(s, a^n) R^k(s, a^1, ..., a^n) \\ &= r^k(s, \pi) - V_\alpha^k(\pi_*) + \sum_{s' \in S} P(s' \mid s, \pi) h^k(s', \pi_*). \end{aligned}$$

For the average reward matrix game (12), the equilibrium point π_* is defined such that

$$Val[R^k(s), \pi_*] \geq Val[R^k(s), \pi_*^{-k}, \pi^k]. \tag{13}$$

which can be expanded to

$$\begin{aligned} r^k(s, \pi_*) - V_\alpha^k(\pi_*) + \sum_{s' \in S} P(s' \mid s, \pi) h^k(s', \pi_*) \geq & \\ r^k(s, \pi_*^{-k}, \pi^k) - V_\alpha^k(\pi_*) + \sum_{s' \in S} P(s' \mid s, \pi_*^{-k}, \pi^k) h^k(s', \pi_*). & \end{aligned} \tag{14}$$

The connection between average reward irreducible stochastic games and the average reward matrix games are given in the following result.

Theorem 2 *The following are equivalent:*

(i) π_* is an equilibrium point in the average reward irreducible stochastic game with bias equilibrium value $h^k(\pi_*)$ and gain equilibrium value $V_\alpha^k(\pi_*)$ for $k = 1, 2, ..., n$.

(ii) For each fixed $s \in S$, the strategy set $\pi_*(s)$ constitutes an equilibrium point in the static n-dimensional equivalent matrix game $(R^1(s), ...R^n(s))$ with bias equilibrium value $h^k(s, \pi_*)$ and gain equilibrium value $Val[R^k(s), \pi_*]$ for k=1,...,n.

Proof of this result follows in a similar manner to that of Theorem 1, by using inequalities (11) and (14).

For games with finite number of players with finite action choices, the following result gives the existence of Nash equilibrium.

Theorem 3 *For every n-player game with finite number of actions, there exists a stationary policy π^* which is a Nash equilibrium (Fink (1964)).*

6. Learning Nash Equilibrium for Irreducible Average Reward Stochastic Games

Reinforcement learning is an efficient method of mapping states to actions that maximizes a numerical reward signal for sequential stochastic decision making problems Kaelbling et al. (1996), Sutton and Barto (1998). The learner is not told which actions to take, but instead must discover the actions that yield the most reward by trying them. In reinforcement learning, an agent need not explicitly model the environment (as in the case of dynamic programming), since the agent's actions are directly based on the rewards. Reinforcement learning (RL) has been successful at finding optimal control policies for a single agent operating in a stationary environment Kaelbling et al. (1996), Sutton and Barto (1998). In recent years RL has been applied to discounted reward stochastic games Borkar (2002), Littman (1994), Hu and Wellman (1998). In this section, a reinforcement learning algorithm for average reward stochastic games is presented. Average reward RL algorithms are commonly referred to as the R-learning algorithms, hence we call our algorithm Nash-R learning.

6.1 A Nash R-learning algorithm

We consider a multiplayer irreducible stochastic game on a finite state space and a finite action space. We are interested in finding stationary Nash equilibrium policies. The structure of Nash equilibria is given in (10), which is similar to Bellman's optimality equation for average reward MDP. Since it can be shown using Theorem 2 that $Val[R^k(s), \pi_*] = h^k(s, \pi_*)$, we can rewrite (12) as follows,

$$R^k(s, a^1, \dots, a^n) = r^k(s, a^1, \dots, a^n) - V^k_\alpha(\pi_*) + \sum_{s' \in S} p(s'|s, a^1, \dots, a^n) Val[R^k(s'), \pi_*], \quad (15)$$

where π_* is the Nash equilibrium point for matrix game $R(\cdot)$, and

$$Val[R^k(s), \pi_*] = r^k(s, \pi_*) - V^k_\alpha(\pi_*) + \sum_{s' \in S} P(s'|s, \pi_*) Val[R^k(s'), \pi_*]. \quad (16)$$

The fixed point of equation (15) is the auxiliary matrices for the stochastic game, whose value is the Nash equilibrium bias value $h^k(\pi_*)$.

In most real applications, the transition probabilities are extremely difficult to obtain. Hence we use adaptive learning mechanisms such as simulation based reinforcement learning to intelligently build $R^k(\cdot)$ matrices. There exist three main classes of reinforcement learning (RL) mechanisms: value iteration based methods such as Q-learning and R-learning algorithms, policy iteration based methods, and combination of value and policy iteration based methods referred to as the Actor-Critic algorithm Borkar (2002). In this paper, we use a value iteration based approach. In this approach that uses value function based RL algorithms Szepesvari and Littman (1999), an estimate of the optimal value function is built gradually from the decision maker's experiences, and these estimates are often used for control of the learning process. For stochastic games, we extend a value function based two time scale R-learning algorithm Abounadi et al. (2001) to a multi-agent scenario. It is considered that each player learns not only her own R values but also other players' R values based on the observation of all other players' immediate rewards and chosen actions. Hence the auxiliary matrices $R(\cdot)$ are developed simultaneously by each player using the learning scheme presented below.

Let at the t^{th} stage, the system state is s and the action combination $(a^1, \dots, a^n)$ is chosen by the players. Also let the system state be s' at the $(t+1)^{th}$ stage. The $R(\cdot)$ matrix and the average reward value for the k^{th} player are updated as follows

$$R^k_{t+1}(s, \mathbf{a}) = R^k_t(s, \mathbf{a}) + \alpha_t[r^k(s, \mathbf{a}) + Val[R^k_t(s'), \pi_{*t}] - R^k_t(s, \mathbf{a}) - G^k_t] \quad (17)$$

$$G^k_{t+1}(s, \mathbf{a}) = G^k_t(s, \mathbf{a}) + \beta_t Val[R^k_t(s'), \pi_{*t}] \quad (18)$$

For any t, since we assume that $r(t)$ is bounded, $\{G^k_t\}$ is also bounded.

In the above updating scheme, π_{*t} represents a Nash equilibrium policy of the matrix game $R_t(s')$ at the t^{th} stage. The element $Val[R^k_t(s'), \pi_{*t}]$ reprerents the Nash equilibrium value for player k in state s' which is obtained as

$$Val[R^k_t(s'), \pi_{*t}] = \sum_{a^1, \dots, a^n} \pi^1_{*t}(s', a^1) \dots \pi^n_{*t}(s', a^n) R^k_t(s', a^1, \dots, a^n).$$

Following Hu and Wellman (2003), we call the $Val[R_t^k(s), \pi_{*t}]$ as Nash R-function. Ideally, for convergence analysis, one needs unique mixed strategy Nash equilibrium. This is an unrealistic assumption. The concept of R-function requires that all players follow a specific Nash equilibrium at each iteration. When there are multiple equilibrium for the stochastic game, choice of different Nash equilibrium by the players may result in different Nash R-functions. The particular choice of Nash equilibrium that we follow in our algorithm implementation is based on Assumption A4 (presented later). In our numerical experiments we explore convergence behavior when this assumption is violated.

The algorithm described in (17) and (18) presents a two time scale iteration scheme, in which the parameters α_t and β_t represent the step sizes at time t. The simulation process on which the learning scheme is applied is assumed to satisfy the following assumption.

Assumption A2: Every state and every action of the players are visited infinitely often.
Assumption A2 can be relaxed, Borkar (2002). In the above two time scale learning scheme, two diminishing step sizes α_t and β_t are needed to update the $R_t(\cdot)$ matrices and the average reward G_t. We make the following assumption for α_t and β_t.

Assumption A3: $\{\alpha_t\} \subset (0, \infty)$, with $\sum_t \alpha_t = \infty$, $\sum_t \alpha_t^2 \leq \infty$.
$\{\beta_t\} \subset (0, \infty)$, with $\sum_t \beta_t = \infty$, $\sum_t \beta_t^2 \leq \infty$. Also, $\beta_t = o(\alpha_t)$, which denotes $\lim_{t \to \infty} \frac{\beta_t}{\alpha_t} = 0$.

The assumption $\beta_t = o(\alpha_t)$ made above implies that the updating of G_t in iteration (18) proceeds at a "slower rate" than R_t in the iteration (17). Hence, in the analysis of R_t, G_t can be viewed as quasistatic. It can be seen from (17) that updating of $R_t(\cdot)$ matrices requires the Nash equilibrium value $Val[R_t^k(s'), \pi_{*t}]$ which in turn requires the Nash equilibrium policy π_{*t} for stage matrix game $R_t(s')$. The following assumption is needed for the Nash equilibrium policy π_{*t}.

Assumption A4: A Nash equilibrium policy π_{*t} for any n-dimensional stage matrix game $[R_t^1(\cdot), \ldots, R_t^n(\cdot)]$ satisfies one and only one of the following properties.

1. The Nash equilibrium is global optimal for which $Val[R_t^k(s), \pi_{t*}] \geq Val[R_t^k(s), \pi]$, $k = 1, \ldots, n$,$\forall s \in S$, $\forall \pi$.
2. The Nash equilibrium satisfies $Val[R_t^k(s), \pi_{*t}] \leq Val[R_t^k(s), (\pi^{-k}, \pi_{*t}^k)]$, $k = 1, \ldots, n$, $\forall s \in S$, $\forall \pi$. This property implies that an agent receives a higher payoff when other agents deviate from the Nash equilibrium strategy.

The Assumption A4 guarantees that there is only one Nash equilibrium value Hu and Wellman (2003). Uniqueness of the Nash equilibrium value implies a unique auxiliary matrix game, which again suggests that the auxiliary matrices learned by the players converge to the same matrices. Such an assumption may not hold for all real life scenarios. Hence, in Section 7 for numerical experiments, we discuss effects of relaxing this assumption (see Table 1). The above assertion will be further addressed in our convergence analysis.

6.2 Two Time Scale ODE Convergence Analysis

The convergence of R_t and G_t given by (17) and (18) is the consequence of stochastic approximations. The proposed Nash R-learning algorithm, like most learning algorithms, involves fixed-point computation. The existing approaches to convergence analysis include smooth potential function such as the stochastic gradient algorithm Bertsekas and Tsitsiklis (1996), contraction map Tsitsiklis (1994), Szepesvari and Littman (1999), and ODE methods Abounadi et al. (2001), Gosavi (2004), Abounadi et al. (1996), Borkar and Meyn (2000). Our analysis is inspired by a similar analysis by Abounadi et al. (2001) and Gosavi (2004) based on Borkar's approach to stochastic approximation with two time scales Borkar (1997).

The standard form of stochastic approximation with two time scales as in our algorithm has the following structure. We can only obtain observed values that include noise, which we denote by M_t^k. Let $\mathcal{F}(t)$ denote the history of the algorithm up to and including the point at which the step size α_t for the t^{th} iteration is selected, but just before the noise terms M_t^k are generated, that is, $\mathcal{F}(t) = \sigma(R_s^k, M_s^k, s \leq t)$, $t \geq 0$. Define

$$H(R_t^k)(s, \mathbf{a}) = E[r^k(s, \mathbf{a}) + Val[R_t^k(s', \pi_{*t}] \mid \mathcal{F}(t)], and \tag{19}$$

$$H'(R_t^k, G_t^k)(s, \mathbf{a}) = H(R_t^k)(s, \mathbf{a}) - G_t^k(s, \mathbf{a}). \tag{20}$$

Then (17) can be rewritten as

$$R_{t+1}^k(s, \mathbf{a}) = R_t^k(s, \mathbf{a}) + \alpha_t[H'(R_t^k, G_t^k)(s, \mathbf{a}) - R_t^k(s, \mathbf{a}) + M_t^k], \tag{21}$$

where

$$M_t^k = r^k(s, \mathbf{a}) - E(r^k(s, \mathbf{a})) + Val[R_t^k(s', \pi_{*t})] - E[Val[R_t^k(s', \pi_{*t}] \mid \mathcal{F}(t)]. \tag{22}$$

From (22), it is clear that

$$E[M_t^k \mid \mathcal{F}(t)] = 0. \tag{23}$$

Now $E[(M_t^k)^2|\mathcal{F}(t)]$ is the conditional variance of the noise term M_t^k. Using $Var(\cdot|\mathcal{F}(t))$ to denote the conditional variance, we can write from (22) that:

$$E[(M_t^k)^2|\mathcal{F}(t)] = Var(r(s, \mathbf{a})|\mathcal{F}(t)) + Var(Val[R_t^k(s'), \pi_{*t}]|\mathcal{F}(t)). \tag{24}$$

Since the reward $r(s, \mathbf{a})$ is bounded and there are only finitely many states and policies, the conditional variance is bounded. Let C denote the bound of $Var(r(s, \mathbf{a})|\mathcal{F}(t))$. The conditional variance of $Val[R_t^k(s'), \pi_{*t}]$ given $\mathcal{F}(t)$ is bounded above by the largest possible value that this random variable could take. This value is $\max_{s' \in S} \max_{\pi_{*t}(s')} \; [\; R_t^k(s', \pi_{*t}(s'))]^2$. Then we can write (24) as:

$$E[(M_t^k)^2|\mathcal{F}(t)] \leq C + \|(R_t^k)^2\|_\xi, \tag{25}$$

where $\| \cdot \|_\xi$ denote the weighted maximum norm, Tsitsiklis (1994). From Abounadi et al. (2001), Tsitsiklis (1994) it follows that $H(.)$ is "nonexpansive" in the weighted maximum norm sense:

$$\|H(R_t^k) - H(R_t'^k)\|_\xi \leq \|R_t^k - R_t'^k\|_\xi \tag{26}$$

Since G_t^k can be viewed as quasi stationary in the analysis of (17), it follows that $H'(R_t^k, G_t^k)$ also satisfies the property (26).

6.3 Boundedness and Convergence

We now address the boundedness of R_t^k. Our approach is derived from Bertsekas and Tsitsiklis (1996). From (18), boundedness of $r^k(\cdot)$ implies that G_t^k is bounded. Hence G_t^k does not influence the boundedness of R_t^k sequence in (17). The following results require Assumptions A1 through A4.

Theorem 4 *Assume that for all one-stage costs $r^k(s, \mathbf{a})$, $R_0^k \geq 0$. Then, the sequence $\{R_t^k\}$ generated by the algorithm (17) is bounded with probability 1.*

The proof of this result follows exactly similar to Lemma 9 of Tsitsiklis (1994). Also, similar to Lemma 4 in Tsitsiklis (2002), we could write equation (21) as

$$R_{t+1}^k(s, \mathbf{a}) = (1 - \alpha_t) R_t^k(s, \mathbf{a}) + \alpha_t [H'(R_t^k, G_t^k)(s, \mathbf{a})] + \alpha_t M_t^k, \tag{27}$$

and the boundedness proof follows.

Now, using (25), (26), Theorem 4, and the fact that G_t^k is also bounded with probability 1, the time asymptotic part of the two time scale recursions R_t^k, and G_t^k tracks the associated ODEs

$$\dot{R_t^k} = H'(R_t^k, G) - R_t^k, \tag{28}$$

and

$$\dot{G_t^k} = 0, \tag{29}$$

where in (28) G is treated as a fixed parameter, since $\beta_t = o(\alpha_t)$ Borkar and Soumyanath (1997).

Remark: The proof of Theorem 4 can also be argued in the following manner. The boundedness of $\{R_t^k\}$ can be shown by the following observations. In equation (19), by assumption, $r^k(\cdot)$ is bounded. From which it follows that the $Val[R_t^k(\cdot)]$ is also bounded since the elements of the $R_t^k(\cdot)$ matrix are comprised of $r^k(\cdot)$. With finite numbers of players and actions, $\|H(R_t^k)\|_\xi$ is also bounded. Now, $H'(\cdot)$ in equation (20) is bounded in $\|\cdot\|_\xi$, since it is a linear combination of $H(\cdot)$ and $G(\cdot)$, which are already shown to be bounded. The aforementioned facts along with equation (25) imply that $\{R_t^k\}$ of equation (21) is bounded with probability 1.

Now what remains to be shown for the convergence of our RL algorithm is the stabilities of the associated ODEs. The equilibrium point R^{k*} of the ODE (28) is precisely the corresponding fixed points of $H'(\cdot, \cdot)$. Let G^{k*} be the equilibrium associated with the ODE (29). For simplicity of notation, now we write $H'(R_t^k, G)$ as $H(R_t^k)$.

The ODE (28) is independent of ODE (29). Hence we consider ODE (29) first. We present the following lemmas that are based on the results of Abounadi et al. (2001) and Borkar and Soumyanath (1997).

Lemma 5 *The ODE (29) has a unique global asymptotically stable equilibrium point G^*.*

Since $F(G^k)$ is a non-expansive mapping, and the fixed point set is not empty, the solution of the differential equation converges to an asymptotically stable equilibrium point.

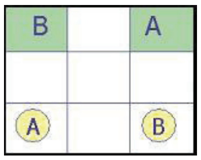

Figure 2: A Grid-World Game

Lemma 6 *The ODE (28) has a unique global asymptotically stable equilibrium point $R^{k*}(G^k)$, and $R^{k*}(G^k)$ is Lipschitz continuous.*

For any fixed G value, fixed point R^{k*} exists. Now since $H(\cdot)$ is a weighted maximum norm nonexpansive, the ODE (28) has a unique global asymptotically stable equilibrium point $R^{k*}(G)$. Since

$$R^{k*}(s, G^k) = r^k(s) - G^k + \sum_{s' \in S} p(s'|s, \pi_*) Val[R^{k*}(s'), \pi_*]$$

is a linear function of G^k, $R^{k*}(G^k)$ is Lipschitz continuous.

Based on the above analysis and the Theorem 1.1 from Borkar (1997), we make the following conclusion.

Theorem 7 *For the Nash R-learning algorithm (17)(18), $(R_t^k, G_t^k) \to (R^{*k}, G^k)$ a.s. for each k.*

7. Numerical Experiments

The grid games have been popular testbeds for evaluating and benchmarking of multiagent learning algorithms since these games possess all the key elements of dynamic games. For example, the MinMax Q-learning algorithm Littman (1994) was implemented on a two-person zero-sum soccer grid game, and the Nash Q-learning algorithm Hu and Wellman (1998) was tested on two-person general-sum grid-world games. We adopt one of the grid games used in Hu and Wellman (1998) to test and benchmark our Nash R-learning algorithm.

7.1 A Grid-World Game

As shown in Figure 2, Player A starts from the lower left cell and tries to reach the upper right cell, her goal state. Player B starts from the lower right cell and tries to reach the upper left cell. The players can only move *up, down, left, or right* to the adjacent cells. After both players select their actions, the two moves are executed at the same time. If they collide with each other, the players bounce back to their previous cells and get punished with reward of -1. The game terminates as soon as any player reaches her goal state, upon which the player gets a reward of 100.

The objective of a player is to reach her goal state through the shortest path, which satisfies the maximizing average (or, discounted) reward criterion. Two noninterfering shortest paths of the players, that are constituted of the best responses, represent a Nash equilibrium. In Figure 3, we identify some of the Nash equilibrium paths for this grid game. The

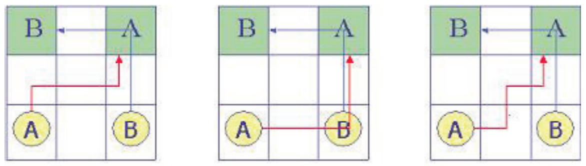

Figure 3: Some Nash equilibrium paths for the grid-world game

Nash equilibrium paths take both players four steps to reach their goals. Clearly there are multiple Nash equilibrium policies, but there is only one unique Nash equilibrium average cost, which is $100/4 = 25$.

In this grid game, the players' joint positions define the system state. Since there are nine cells and the players can not be in the same cell, the total number of states is seventy two including fifteen terminating states (in which at least one player is at her goal state). It is considered that the players do not know their goal states and the payoff functions. They choose their actions independently and simultaneously. They can observe the opponent's previous actions, immediate rewards and current state.

7.2 Benchmarking and Testing of the Nash-R Algorithm

We implemented four different learning schemes on the above grid game, for the purpose of benchmarking and testing of the Nash-R algorithm. The first two learning schemes are based on MDP approximations. The MDP approximations are often used for games due to two common reasons: 1) lack of computationally feasible methods to solve stochastic games, and 2) lack of complete information about the other players. In the first scheme (referred to as MDP-RL), each player ignores the existence of the other player, and uses an independent single agent R learning algorithm for MDPs. The players perform their actions, obtain a reward and update their $R^k(s, a^k)$ values using the $max_{a^k} R^k(s', a^k)$ operator, without regard to the actions performed by the other player. In the second scheme with a modified MDP approach, the players observe the opponents' action (not the reward) and update their $R^k(s, a^1, a^2)$ values using the $max_{(a^1,a^2)} R^k(s', a^1, a^2)$ operator. We refer to this scheme as MMDP-RL.

The remaining two schemes are variants of the Nash-R learning algorithm, shown in Figure 4. In the convergence analysis of Nash-R algorithm, it is assumed that at each stage of learning the players use a unique Nash equilibrium value $Val[R^k_t(s'), \pi_{*t}]$ in updating their matrices of $R^k_{t+1}(s, a^1, a^2)$ values for all k. The purpose of examining the first of these two variants of Nash-R was to assess the impact of deviation from the above assumption by allowing players to use different Nash equilibrium value for updating purposes, when multiple equilibria exist. In our implementation, each player randomly chose any one of the Nash equilibria values of the stage game. In the last of the four schemes, the players always use the same Nash equilibrium value for updating their matrices. The results from the numerical experiments are presented in Table 1.

```
Nash-R Algorithm for a Grid-World Game:

1. Let time step $t = 1$. Initialize matrices $R^k(s, a^1, a^2) = 0$ and average cost
   $G^k(s, a^1, a^2) = 0$ for all $s \in S$, $a^1 \in A^1(s)$, $a^2 \in A^2(s)$ and $k = 1, 2$. Set the
   frequency for the tuple $(s, a^1, a^2)$ being visited $n(s, a^1, a^2) = 0$. Set the number of
   learned episodes $episode = 0$. Start system simulation.

2. Learning Phase: While $episode < 5000$ do

   (a) Randomly generate the initial positions for the players.
   (b) While neither player is in the goal position, do the following:
       i. Choose for each player a random action.
       ii. Simulate the chosen actions. Let the system state at the next decision epoch
           be $s'$, and $r^k(s, s', a^1, a^2)$ be the immediate reward for player $k$ earned as a
           result of actions $(a^1, a^2)$ chosen in state $s$.
           • If player $k$ enters her goal position, set immediate reward
             $r^k(s, s', a^1, a^2) = 100$, for $k = 1, 2$.
           • If the players collide with each other in a cell other than the goal states,
             set $r^k(s, s', a^1, a^2) = -1, k = 1, 2$, and $s' = s$.
           • In all other cases $r^k(s, s', a^1, a^2) = 0, k = 1, 2$.
       iii. Obtain a Nash equilibrium policy $\pi_{*t}$ for the stage matrix game $R^k_t(s'), \forall k$,
            using a suitable mathod (e.g., GAMBIT), and calculate the corresponding
            Nash equilibrium value for each player $Val[R^k_t(s'), \pi_{*t}]$.
       iv. $n(s, a^1, a^2) \leftarrow n(s, a^1, a^2) + 1$. Update the steps sizes as: $\alpha_t = 1/n(s, a^1, a^2)$,
           and $\beta_t = 1/t$.
       v. Update stage matrices $R^k(s, a^1, a^2)$ and the $G^k s, a^1, a^2)$ values for each player
          as follows.

          $R^k_{t+1}(s, a^1, a^2) = (1 - \alpha_t)R^k(s, a^1, a^2) + \alpha_t \left\{ r^k(s, s', a^1, a^2) - G^k_t + Val[R^k_t(s'), \pi_{*t}] \right\}.$
          $G^k_{t+1} = (1 - \beta_t)G^k_t + \beta_t Val[R^k_t(s'), \pi_{*t}].$

       vi. If $s'$ indicates goal state for either player, then $episode \leftarrow episode + 1$, go to
           step 2. Else, set $s \leftarrow s'$, and $t \leftarrow t + 1$, go to step $2(b)i$.

3. Evaluation Phase

   (a) Select the players' starting locations as the bottom corners opposite to their goal
       states.
   (b) Let the players choose their actions based on the Nash equilibrium of the matrix
       game for the current state, and continue to move until both players reache their
       goal states.
   (c) Compare the paths followed by each player to reach goal state and compare them
       with known Nash strategies.
```

Figure 4: A Nash-R reinforcement learning algorithm for computing Nash equilibrium policies for a grid-world game.

Benchmarking schemes	Success rate $\pm 95\% C.I.$
MDP-RL (learning without observing another player's action)	$16\% \pm 5.1\%$
MMDP-RL (learning with observing another player's action)	$30.0\% \pm 6.3\%$
Nash R learning (updated with possible different Nash values)	$29.8\% \pm 5.6\%$
Nash R learning (updated with the same Nash values)	$98.9\% \pm 1.3\%$

Table 1: Testing and benchmarking results for four different learning algorithms

For each of the four schemes, the players were allowed to learn their R-values (for MDP-RL and MMDP-RL schemes) and R-matrices (for the two Nash-R based schemes) over 5000 training episodes. After each training session, the players were allowed to test the strategies calculated based on the learned values/matrices for one episode. Starting positions for both the players were always set to the lower corners as shown in Figure 2. During the one episode testing phase, for mixed strategies, random numbers were generated to select actions. We labeled a testing phase as "success" if both players reached their respective goal states in four steps (which is optimal). For every scheme, two hundred training-testing runs were conducted and the respective "success" rates were obtained. We note that our method of establishing success rate is different from that of Hu and Wellman (2003). It appears that they compared the Nash equilibria obtained from the learned Q-matrices with the theoretical results, and a match was defined as a "success".

We adopted a *non-exploitive exploration* strategy to ensure that all states are visited a large number of times. According to this strategy, the players choose their available actions with equal probability. Let t be the number of steps in the learning phase of the game and $n(s, a^1, a^2)$ denote the visit frequency of the tuple (s, a^1, a^2). The learning rates were obtained as $\alpha(s, a^1, a^2) = \frac{1}{n(s,a^1,a^2)}$ and $\beta = \frac{1}{t}$ (for Nash-R only). We define a training (learning) episode as a process that starts when the players are assigned random positions (except goal positions) and ends when either player reaches her goal position. During the experiments, we found that one run with 5000 episodes usually takes 40,000 steps ($t = 40,000$) (same as in Hu and Wellman (2003)). The total number of state-action tuples in this grid game is 424, and each tuple on average is visited 95 times.

The success rates were calculated for each of the three benchmarking algorithms along with Nash-R. Note that the success rates are binomial proportions obtained from 200 Bernoulli trials consisting of 5000 training episodes followed by one testing episode. We obtained a 95% confidence intervals on the success rates using a normal approximation. Several observations are made from the results.

1. The single player learning scheme (MDP-RL) which ignores the existence of the other player achieved a 16% success rate in attaining equilibrium strategy. This is quite expected, since it is well known that in a game, considering other players as part of the stationary environment yields poor result.

2. In the MMDP-RL scheme, the players observe other's actions and use that information in their single player learning schemes. With more information than MDP-RL, this

algorithm performed much better than MDP-RL and obtained a success rate of around 30%.

3. For the Nash-R algorithm that uses possibly different stage Nash-equilibria for updating, the success rate was significantly lower than the Nash-R scheme that uses the same Nash equilibrium value for updating. When the players, at any stage of learning, use different equilibrium policies to update their matrices, the final form of the matrices learned by the players are unlikely to be the same. As a result, the policies learned by each player may not constitute a Nash equilibrium, which is evident from the result. This validates, for the convergence of Nash-R algorithm, the need for the necessary condition of having a unique Nash equilibrium value.

4. An arbitrary Nash-R scheme that violates the uniqueness condition may not perform significantly better than the MMDP-RL approximation.

8. Concluding Remarks

In this paper, from the perspective of developing solution methodologies for obtaining Nash equilibrium strategies for average reward irreducible stochastic games, we have first discussed the existence of series of equivalent matrix games. To construct these equivalent matrix games, we have developed efficient simulation based two time scale stochastic approximation algorithm using reinforcement learning. An ODE convergence analysis for this two time scale stochastic approximation algorithm is presented. The scheme is benchmarked and tested using a grid game.

The convergence analysis requires a strict assumption for the matrix games (i.e., uniqueness of the Nash equilibrium value for the matrix games). For many problems, it may be difficult to meet the above assumption. But our experimental results suggest that, even when there are multiple Nash equilibrium values for matrix games, optimal policies can be attained almost all of the times by ensuring that the players use identical matrix Nash equilibrium value in updating their R-matrices. The proposed method enjoys excellent generalization capabilities inherent in the reinforcement learning and combines the relative ease in computation of Nash equilibrium in the matrix games.

Acknowledgments

This work was supported in part by a grant # ITR/AP: DMI-0113946 and # ECS 0400268 from the National Science Foundation.

References

J. Abounadi, D.P. Bertsekas, and V. S. Borkar. Ode analysis for q-learning algorithms. LIDS report, MIT, Cambridge, MA, 1996.

J. Abounadi, D.P. Bertsekas, and V. S. Borkar. Learning algorithms for Markov decision processes with average cost. *SIAM Journal on Control and Optimization*, 40(3):681–698, 2001.

J. Abounadi, D. Bertsekas, and V. S. Borkar. Stochastic approximation for nonexpansive maps: Application to q-learning algorithms. *SIAM Journal on Control and Optimization*, 41(1):1–22, 2002.

E. Altman and F.M. Spieksma. Contraction conditions for average and α-discount optimality in countable state Markov games with unbounded rewards. *Journal of Cognitive Systems Research*, 2:55–66, 1997.

Eitan Altman and Odile Pourtallier. Approximating Nash equilibria in nonzero-sum games. *International Game Theory Review*, 2:155–172, 2000.

Dimitri P. Bertsekas and John N. Tsitsiklis. *Neuro-Dynamic Programming.* Athena Scientific, 1996.

V. S. Borkar. Reinforcement learning in Markovian evolutionary games. *Advances in Complex Systems*, 5(1):55–72, 2002.

V. S. Borkar. Stochastic approximation with two-time scales. *System and Control Letters*, 29, 1997.

V. S. Borkar and S.P. Meyn. The o.d.e method for convergence of stochastic approximation and reinforcement learning. *SIAM J. Control Optim.*, 38(2):447–469, 2000.

V. S. Borkar and K. Soumyanath. An analog scheme for fixed point computation, part i: theory. *IEEE Transactions Circuits and Systems I. Fundamental Theory and Application*, 44, 1997.

M. Bowling and M.M. Veloso. An analysis of stochastic game theory for multiagent reinforcement learning. Technical Report CMU-CS-00-165, Computer Science Department, Carnegie Mellon University, 2000.

R.I. Brafman and M. Tennenholtz. A near optimal polynomial time algorithm for learning in certain classes of stochastic games. *Artificial Intelligence*, 121, 2000.

T.K. Das, A. Gosavi, S. Mahadevan, and N. Marchalleck. Solving semi-Markov decision problems using average reward reinforcement learning. *Management Science*, 45(4):560–574, 1999.

J. Filar and K Vrieze. *Competitive Markov Decision Process.* Springer, Verlag New York Berlin Heidelberg, 1997.

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

Z. Ghahramani. Learning dynamic Bayesian networks. In C.L. Giles and M. Gori, editors, *Adaptive Processing of Sequences and Data Structures*, pages 168–197. Springer-Verlag, Berlin, 1998.

A. Gosavi. Reinforcement learning for long-run average cost. *European Journal of Operational Research*, 155:654–674, 2004.

P. Jean-Jacques Herings and Ronald J.A.P Peeters. Stationary equilibria in stochastic games: Structure, selection, and computation. Research memoranda 004, Maastricht : METEOR, Maastricht Research School of Economics of Technology and Organization, 2000.

J. Hu and M. P. Wellman. Nash q-learning for general-sum stochastic games. *Journal of Machine Learning Research*, 4:1039–1069, 2003.

J. Hu and M. P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. In *In Proceedings of the Fifteenth International Conference on Machine Learning*, 1998.

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

J. Li. Learning nash equilibrium in average reward stochastic games. *Unpublished Doctoral Dissertation, University of South Florida, Tampa, Florida*, 2003.

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

M.L. Littman. Markov games as a framework for multi-agent reinforcement learning. In *In Proceedings of the Eleventh International Conference on Machine Learning*, pages 151–163, San Francisco, CA, 1994.

Sridhar Mahadevan. Average reward reinforcement learning: Foundations, algorithms, and empirical results. *Machine Learning, Special Issue on Reinforcement Learning*, 22:159–196, 1996.

R. McKelvey and A. McLennan. Computation of equilibria in finite games. In H. Amman, D. Kendrick, and J. Rust, editors, *Handbook of Computational Economics*, pages 87–142. Elsevier, 1996.

Richard Mckelvey, Andrew Mclennan, and Theodore Turocy. Gambit: Software tools for game theory. Technical report, http://econweb.tamu.edu/gambit/.

R.B. Myerson. *Game Theory– Analysis of Conflict, 3rd Printing.* Harvard University Press, 1997.

V. Nanduri and T. K. Das. A machine learning approach to finding Nash equilibrium in multi-player matrix games. *Working Paper, Dept. of IMSE, University of South Florida, Tampa, Florida 33620*, May 2007.

John Nash. Non-cooperative games. *The Annals of Mathematics*, 54(2):286–295, 1951.

M.L. Puterman. *Markov Decision Processes.* John Wiley and Sons, New York Chichester Brisbane Toronto Singapore, 1994.

J. Rao, K. K. Ravulapati, and T. K. Das. A simulation based approach to study stochastic inventory planning games. *International Journal of Systems Science*, 34(12), 2002.

E. Rasmusen. *Games Information, 3rd Edition*. Blackwell publisher, 2001.

K. K. Ravulapati, J. Rao, and T. K. Das. A reinforcement learning approach to stochastic business games. *IIE Transactions on Scheduling and Logistics*, 36, 2004.

L.S. Shapley. Stochastic game, 1953. In H.W Kuhn, editor, *Classics in Game Theory*. Princeton University Press, 1997.

R.S. Sutton and A.G. Barto. *Reinforcement Learning*. The MIT press, 1998.

C. Szepesvari and M.L. Littman. A unified analysis of value-function-based reinforcement-learning algorithms. *Neural Comput.*, 11:2017–2059, 1999.

John N. Tsitsiklis. On the convergence of optimistic policy iteration. *Journal of Machine Learning Research*, 3:59–72, 2002.

John N. Tsitsiklis. Asynchronous stochastic approximation and q-learning. *Machine Learning*, 6:185–202, 1994.

R. Wilson. Computing equilibria of n-person games. *SIAM Applied Math*, 21:80–87, 1971.