Learning to Negotiate Optimally in Non-Stationary Environments

Vidya Narayanan and Nicholas R. Jennings
$\{vn03r, nrj\}$@$ecs.soton.ac.uk$

Intelligence, Agents, Multimedia
School of Electronics and Computer Science
University of Southampton SO17 1BJ, UK.

Abstract. We adopt the Markov chain framework to model bilateral negotiations among agents in dynamic environments and use Bayesian learning to enable them to learn an optimal strategy in incomplete information settings. Specifically, an agent learns the optimal strategy to play against an opponent whose strategy varies with time, assuming no prior information about its negotiation parameters. In so doing, we present a new framework for adaptive negotiation in such non-stationary environments and develop a novel learning algorithm, which is guaranteed to converge, that an agent can use to negotiate optimally over time. We have implemented our algorithm and shown that it converges quickly in a wide range of cases.

1 Introduction

Automated negotiation plays a key role in resolving conflicts in multiagent systems in which individual agents have different stakes in a joint operation. Now, in many such cases, agents have little information about one another, and, in addition, the environment changes as a result of interactions between them [4]. Thus, learning about the other agents in the system and about their common environment becomes essential for effective performance. In particular, while an agent is engaged in negotiations, it has to learn about the *negotiation parameters* and *strategies* [10] of its opponents if it is to bargain optimally in such non-stationary environments.

Generally speaking, reinforcement learning, in particular Q-learning, is often used in multiagent systems since it does not need a model for learning and can be used online [4], [11]. In this vein, several researchers have adopted the stochastic game framework for multiagent reinforcement learning and have developed solution techniques like *Nash Equilibrium* [4], [9] and best response strategies [12]. Others, like [2], have used *fictitious play* techniques to analyse learning in games. However our problem is different. We are not trying to model learning in multiagent systems using game theory, but rather, we are trying to develop negotiation techniques for multiagent systems for which learning is necessary. Therefore, we believe that reinforcement learning, in which agents learn to maximize a reward signal, is not best suited for our purposes. Moreover, reinforcement learning algorithms rely on the assumption that the underlying environment is stationary (i.e., the parameters and strategies of the opponent do not change over time,

they are simply unknown). Now, this is clearly not the case in many realistic negotiation encounters, so we need to look at other learning methods.

Some of the first models to discuss the need for learning in negotiations among agents were [5] and [7]. However, while these papers discuss the concept of reasoning based on experience among negotiating agents, they do not explicitly develop a learning model. In [13] this notion is formally modelled, as a sequential learning mechanism based on a Bayesian belief update process. Specifically, a very general framework is adopted for the negotiation in which multiple agents bargain for multiple items. The *a priori* model that the agents have of their opponents is constantly updated using current information which is received as a signal from the environment. In particular, given the prior knowledge of an agent and the newly incoming information, the posterior distribution of the knowledge of the agents is computed using Bayesian rules. However, this model does not capture the non-stationarity in the environment. By this, we mean that, although the agents are assumed not to know the distribution of players' strategies and their negotiation parameters, it is assumed that this distribution does not change over time. Now, this is a serious shortcoming in the types of open environment in which multiagent systems are often deployed and is something that we wish to rectify in this work. Thus, in our case, the agent has to learn how its opponents change their strategies and then respond optimally to them. We have used Bayesian learning in our model because we believe it is more suitable than other forms of learning, like reinforcement or supervised learning, to represent uncertainties in the negotiation process as a probability function and then update this based on signals received from the environment.

In particular, we consider negotiation between a pair of agents over a single issue (price). We use the non-stationary Markov chain framework to model the negotiation process and prove, for the first time, an important estimation property for these processes (namely that the future distribution of the states can be obtained given their initial distribution and the probabilities of state change during the process). Within this framework, at each stage in the negotiation process, the agent uses Bayesian updating to learn the strategy that its opponent is most likely to use and, based on this, determines what it should adopt to maximise its payoff at that stage of the negotiation process. In so doing, we analytically prove that in repeated negotiations our algorithm converges to the actual optimal strategy at every stage of the negotiation process. We verify this by means of an example problem, and have shown that our algorithm is at least 200% more effective than random estimation. Our empirical results also show that, on average, the algorithm converges within 24 iterations and that each iteration takes 18.92 seconds.

The rest of the paper is organized as follows: Section 2 presents the basics of our negotiation model, Section 3 the learning model, Section 4 our empirical results. Finally, Section 5 concludes.

2 The Negotiation Model

In this section, we detail the basic concepts that we will use in our model. First, we define the notion of a *stochastic process* and the corresponding notion of a *Markov Chain* in order to use it to capture the uncertainty in our domain. Then, to provide a grounding for our learning algorithm, we give the *Bayesian probability rule* and explain how it is

used in Bayesian learning techniques. Finally, we present the concept of mixed strategy profiles in the context of classical game theory which we will use as a framework to describe the strategies that our negotiating agents will use.

Turning first to stochastic processes and Markov Chains, let the real random variable $(r.r.v)$, X, be defined as a function that maps a space of events to a real number. Formally, if Ω represents the space of events and $\Re$ represents the set of all real numbers, then $X : \Omega \rightarrow \Re$. The probability that $X = a$ where $a \in \Re$ is represented as $Pr(X = a)$. A sequence of $r.r.vs$ that is indexed by some parameter, n, where $n \in T$ and T is a suitable index set, is represented as X_n and is called a *Stochastic Process*. A *realization* of a stochastic process $X_n, n \in T$, is an assignment to each $n \in T$, of a possible value of X_n. Now, there are two important sets associated with stochastic processes:

- *State Space* S: This is the space in which possible values of X_n lie. If $S = 0, 1, 2, 3, ..$ then the associated process is called a discrete state process. In our model the state space is discrete.
- *Index Parameter* T: If $T = 0, 1, 2, 3, ...$ (i.e., if T takes only discrete values) then X_n is called a discrete time stochastic process. Again, in our model, we assume that T is discrete.

We also need to define the concept of the *conditional probability* of random variables. Thus, if A and B represent two events, then the *conditional probability* of event A given that event B occurred is defined by:

$$Pr\{A|B\} = \frac{Pr\{A \; and \; B\}}{Pr\{B\}} \tag{1}$$

We now move onto Markov processes.

Definition 1: A *first order* Markov process is a stochastic process that satisfies the following condition:

$$Pr\{X_{n+1} = x|X_n = x_n, X_{n-1} = x_{n-1}, ...\} = Pr\{X_{n+1} = x|X_n = x_n\} \tag{2}$$

Intuitively, this means that the probability of any future behaviour of the process, when its present state is known exactly, is not changed by any additional information about its past states. That is, the Markov Process is *memoryless*. This property of the stochastic process enables elegant mathematical analysis. Now, it is not unreasonable to suppose that the negotiation strategy at a particular stage is dependent only on the immediately preceding stage, since a single offer captures the entire decision making that preceded it. Therefore we model the negotiation process as a Markov process. Here, the probability $P = Pr\{X_{n+1} = x|X_n = x_n\}$ is called the *one-step transition probability function* and it is fundamental to the study of Markov processes and, as such, to our model. This function **P** is usually represented as a matrix where each entry (i, j) is given by $P_{ij} = Pr\{X_{n+1} = j|X_n = i\}$. A Markov process, for which S and T are discrete, is called a *Markov chain* and, so therefore, can our model.

Having introduced Markov chains, we now define the notions on which the learning component of our model is based. Specifically, our agent updates its beliefs using a *Bayesian update rule*. Such Bayesian analysis is often used to estimate the most probable underlying model for a random process, based on some observed data or inference [13], and we choose it here because it enables us to represent the uncertainty in the environment as probability functions and it gives us a formal procedure to update these functions based on our observations. Now, let $A_1, A_2, ..., A_n$, represent n random events. Then, we let $X_n, t = 0, 1, ...$ be the stochastic process we are trying to estimate. Each of these n events can be thought to represent the hypothesis that the parameters of $\{X_n, t = 0, 1, ...\}$ belong to sets $T_1, T_2, ..., T_n$. Finally, we let the event B represent the set of observed data. Now, *Bayes rule* can be stated as:

$$Pr\{A_i|B\} = \frac{Pr\{B|A_i\} \times Pr\{A_i\}}{\sum Pr\{B|A_i\} \times Pr\{A_i\}} \tag{3}$$

where $Pr\{A_i\}$ is the prior probability of model A_i in the absence of any information, $Pr\{\ B|A_i\}$ is the likelihood that observation B was produced given that the model was A_i, and $Pr\{A_i|B\}$ is the posterior probability of the model being A_i given the observation is B. Our learning agent will use these inference rules to estimate the underlying randomness of the negotiation process.

Our final discussion in this section refers to the strategies that the agents will use in the negotiation process. In classical game theory, a *Strategic-Form* game has three elements:

1. the set of players $i \in I$, where $I = 1, 2, ..., n$
2. the *pure-strategy space* S_i, for each player i, and
3. *payoff functions* u_i that give player $i's$ utility $u_i(s)$ for each *profile*, $s = (s_1, s_2, ..., s_n)$, of strategies

Therefore in game theory a *strategy* is perceived as an action choice of a player that has a *utility* associated with it. Often the objective in games is to determine a strategy s_i that will maximise player $i's$ payoff given the strategy set, s_{-i}, that the other players use. Also, notice that the u_i that player i receives depends on the strategies of all the players in the game and is not related to an isolated strategy that i may use. In the definition of a strategic game we called the strategy space a *pure-strategy*, this is because game theorists often refer to an alternate strategy space called the *mixed-strategy space*. The mixed-strategy space is a probability distribution over the space of pure-strategies and for mathematical ease of analysis it is often more convenient to deal with the mixed-strategy space [3]. Thus in our problem the agent will learn the mixed-strategy profile of its opponent and evolve a strategy in response to that strategy that earns it maximum payoff. In this sense, our negotiation problem can be considered as a two-player strategic game.

Having introduced the main concepts that we use in our model, we are now ready to describe the learning component and outline the solution procedure we have developed.

3 The Learning Model

The main objective of our agent is to learn the mixed-strategy profile of its opponent and determine a strategy in response to this profile that maximises its pay-off at each stage of the negotiation process. This learning problem is complicated by the fact that the agent has no information about its opponent and that the strategy that the opponent uses may well change during the course of the negotiation. Now, to model this process of change in the strategies of the opponent, we use a *non-stationary Markov chain*. Formally, a *non-stationary Markov chain* is a Markov chain (see Section 2) whose one-step transition probability function, **P(t)** $= Pr\{X_{n+1} = x | X_n = x_n\}$, varies with time [8]. If we define the state space, S, associated with this non-stationary Markov chain to be the space of all possible strategies that the opponent can employ, and the corresponding time dependent transition probability function, $\mathbf{P^n(t)}$, to represent the probability that the strategy of the opponent changes at each step n of the game and that this probability function itself is a function of time, then this framework gives us a powerful tool to describe and analyse the non-stationary negotiation process that we are trying to model. Therefore, we adopt this mathematical formulation in this work.

Now, if $\mathbf{P^n(t)}$ were specified as a function of time, then we could obtain the strategy profile of the opponent at each stage of the negotiation process using standard stochastic process analysis [6] and then obtain a strategy that maximises our own payoff using maximization algorithms [1]. But since this function is unknown in our problem, the agent has to learn it from interactions with the opponent and the environment. In Bayesian learning, as explained in Section 2, the probability of a hypothesis being true is continuously updated by signals that are received from the environment and, as such, is well suited to modelling uncertainty in the environment. In our problem, in order to learn the function, $\mathbf{P^n(t)}$, we propose that the learning agent initially has a finite number of hypotheses [1] of the possible distributions of $\mathbf{P^n(t)}$ which it updates using Bayesian inference rules. This means that in successive negotiations, by updating the different hypotheses, the agent comes closer to estimating the true value of $\mathbf{P^n(t)}$ and, therefore, to estimating the true optimal strategy in response to its opponent's play.

Formally, we consider two agents say buyer X and seller Y, negotiating over price. We assume that the buyer is learning to respond to the strategy of the seller. Now, we assume that there is a payoff function, $u^n_{s_x}(t)$, associated with X, which depends on the strategy s that X uses in response to Y at each step, n, of the negotiation. $X's$ objective in the negotiation is to find a strategy profile that maximises $u^n_{s_x}(t)$ which we assume is known to X. Therefore, X must learn the strategy profile of Y in order to determine an optimal response strategy. To describe how X learns this strategy within the Markov chain, we need to first define some of its properties. In stationary Markov chains, the probability of moving from state i to state j in n time steps is represented as P^n_{ij} and is given by [6]:

$$P^n_{ij} = \sum_{k=0}^{m} P^r_{ik} \times P^s_{kj} \tag{4}$$

[1] This assumption reduces the search space while allowing us to represent the uncertainty in the problem.

where m is the total number of states, k is some intermediate state and $r + s = n$. Intuitively, this means that the probability of moving from one state to another in a certain number of steps, n, is equivalent to the probability of moving from the first state to an intermediate state, k, in r steps together with the probability of moving from k to the final state in the remaining number of steps. Now, from matrix algebra, we recognise equation 4 as the formula for matrix multiplication, so the n-step transition matrix, represented by P^n, is equal to $P^{(n)}$ or that the entries P_{ij}^n in P^n are equal to the entries in the matrix $P^{(n)}$, which is the n^{th} power of the one-step transition matrix P. It follows that if the probability of the process initially being in state j is p_j, (i.e., $Pr\{\ X_0 = j\}$), then the probability of the process being in state k, at time n is:

$$p_k^n = \sum_{j=0}^{m} p_j P_{jk}^n \tag{5}$$

where m is the total number of states. Thus in our problem if we know the initial distribution of the opponent's strategies we can calculate the probability that the opponent uses a certain strategy after n time steps. This is the main reason for using the Markov chain framework to model the negotiation process. In our model, however, the Markov chain is non-stationary and, therefore, we need to prove, an equivalent result for the non-stationary case (this has not previously been done). Here, since $\mathbf{P^n(t)}$ is a function of time, at each step of the process we have a different transition probability matrix. Here, we propose that to obtain p_k^n as a function of time, we need to multiply n different transition matrices. We now formally state and prove this result.

Theorem 1. *In non-stationary Markov chains, the probability of moving from state i to state j in n time steps, during time instant t, is represented as $P_{ij}^n(t)$ [2] and is given by:*

$$P_{ij}^n(t) = \textstyle\sum_{k=0}^{m} P_{ik}^r(u) \times P_{kj}^s(v)$$

where m is the total number of states, k is some intermediate state, $r + s = n$ and u, v are time instants at which the transitions occur.

Proof. We prove the result when $n = 2$. The event of moving from state i to state j can happen in mutually exclusive ways, of going to some intermediate state $k \in \{0, 1, 2..., m\}$ in the first transition and then moving from state k to state j in the next transition. Now, because of the Markovian assumption that the transition probability is independent of the history of the process, the probability of the second transition is simply $P_{kj}(v)$ and, by definition, the probability of the first transition is $P_{ik}(u)$. Therefore, by the law of total probability: $P_{ij}^2(t) = \sum_{k=0}^{m} P_{ik}^1(u) \times P_{kj}^1(v)$. In the general case, by breaking up the first r steps and then the next s steps into a series of single step transitions and again by using, the law of total probability for each transition, the proof is obtained. □

[2] Here n represents the number of time steps and t represents the fact that the transition probability $P_{ij}^n(t)$ is a function of time.

Thus, in the non-stationary case also, given the probability that initially the process was in, say state j (i.e., $p_j^0(0) = Pr\{X_0(0) = j\}$), then the probability that it is in state k after n time steps and at time t is represented as $p_k^n(t) = Pr\{\ X_n(t) = k\}$ and is given by:

$$p_k^n(t) = \sum_{j=0}^{m} [p_j^0(0)] \times [Q_{jk}^n(t)] \tag{6}$$

where $Q_{jk}^n(t) = [P^0(0)] \times [P^1(1)] ... \times [P^{n-1}(t-1)]$.

Therefore, we now have a means of obtaining the probability distribution of the process and, thereby, the probability distribution of strategies at any stage in the negotiation process given an initial distribution of strategies and the transition probability matrices.

Now, we come to the main issue of learning the transition probability matrices. As already stated, we propose to do this using Bayesian inference rules. However, to do this we must assume that the learning agent, in our case the buyer X, has some knowledge about the negotiation process. Specifically, in order to update its hypothesis about the strategy distribution of Y from the offers that it receives, it has to know how the offer generation process depends on the strategy selection process. To this end, let us assume that S, the set of all possible strategies that Y can use, and as such constitutes the state space of our Markov chain, is given by $S = \{s_0^o, s_1^o, ..., s_m^o\}$. Therefore, Y switches between the strategies in S according to the transition probability matrix $\mathbf{P^n(t)}$, which varies with time. We let $O_n(t)$ represent a sequence of offers made by Y and $O_n^p(t)$ represent the event that the offer at the n^{th} step of the process at time t is p. We also let $H_n(t)$ represent a sequence of finite sets of hypotheses about $\mathbf{P^n(t)}$ during the negotiation process. Therefore $H_n(t) = \{H_n^1(t), ..., H_n^k(t)\}$, where k is some finite positive integer. We assume that the hypothesis representing the true value of the transition probability function also belongs to $H_n(t)$. Then the objective of our learning algorithm is to update each of these hypotheses $\{H_n^i(t) \in H_n(t)\}$ at every step n of the negotiation. The steps of the algorithm are detailed in Algorithm 1. In more detail, using Bayes rule we have for each hypothesis, at step n of the process that:

$$Pr\{H_n^i(t)|O_n^p(t)\} = \frac{Pr\{H_n^i(t)\} \times Pr\{O_n^p(t)|H_n^i(t)\}}{\sum_k^{i=0}[Pr\{H_n^i(t)\}] \times [Pr\{O_n^p(t)|H_n^i(t)\}]} \tag{7}$$

We call $Pr\{H_n^i(t)|O_n^p(t)\}$ the likelihood function, L. Thus each $H_n^i(t)$ is updated in the light of the incoming offer $O_n^p(t)$. Now, B uses the hypothesis $H_n^{new}(t) = \sum_k^{i=0}\{Pr\{H_n^i(t)|O_n^p(t)\} \times H_n^i(t)\}$ to find the strategy used by the opponent. Therefore the learning agent weights the different hypotheses by the probabilities of their occurring in order to form a new hypothesis about $\mathbf{P^n(t)}$. Because of this construction of the new hypothesis we can show that, as $t \to T$ where T is sufficiently large, $H_n^{new}(t)$ approaches the true value of $\mathbf{P^n(t)}$ (see theorem 2 for details). Then, according to $u_{s_x}^n(t)$, the agent determines the strategy $s_{max}^n(t)$ that maximises $u_{s_x}^n(t)$ at each step of the negotiation process. We denote this maximum value of the payoff function by $u_{s_{max}}^n(t)$. This completes the solution procedure for determining the best response

Algorithm 1 The Adaptive Negotiation Algorithm

1. **for** $(t = 0, 1, 2, ..., T)$
2. **for** $(n = 0, 1, 2, ..., t_{terminal})$
3. **initialize** $H_n^i(t) \in H_n(t)$ as an arbitrary distribution.
4. **input** opponent offer $p \in Domain\{O_n(t)\}$
5. $Pr\{H_n^i(t+1)\} \leftarrow Pr\{H_n^i(t)|O_n^p(t)\}$ using equation 7
6. **assign** $H_n^{new}(t) = \sum_{i=0}^{k}\{Pr\{H_n^i(t)|O_n^p(t)\} \times H_n^i(t)\}$
7. **assign** $[P^n(t)] = H_n^{new}(t)$
8. **compute** $[(s_1^o, s_2^o, ..., s_m^o)]^n(t)$ using equation 6
9. **compute** $s_{max}^n(t)$ = max $[(s_1, s_2, ..., s_m)]^n(t) \times u_x^n(t) \times [(s_1^o, s_2^o, ..., s_m^o)]^n(t)]^T$ s.t $\sum_{i=0}^{m} s_i = 1$ and $s_i \geq 0 \; \forall i$
10. **compute** $u_{s_{max}}^n(t)$
11. **next** n
12. **next** t

strategy to the opponent's play and consequently the maximum payoff at each step of the negotiation process.

Here, it is important to note that according to this algorithm, the agent learns across successive negotiations and not within a single negotiation process. Therefore, we claim that in repeated negotiations using our algorithm, the agent learns to use the optimal strategy and earn maximum payoff at each stage of the negotiation process. In order to prove this claim we need the following lemma.

Lemma 1: After a sufficiently large time T, the real probability distribution over the future rational play of a game is ϵ-close to what player i *believes* the distribution is [5].

Here, ϵ-close implies that we can approach arbitrarily close to the actual distribution and rational play means that at each stage of the negotiation the players take the action that maximises their pay-offs. Having stated Lemma 1, we are now ready to prove our main result.

Theorem 2. *In the non-stationary negotiation process, the sequence of n^{th} step strategies $\{s_{max}^n(0), ..., s_{max}^n(t), s_{max}^n(t+1), ..., \}$ and the corresponding sequence of n^{th} step payoff functions, $\{u_{max}^n(0), ..., u_{max}^n(t), u_{max}^n(t+1), ..., \}$, after a sufficiently large*

time T, are ϵ-close to the true optimal strategy and the corresponding maximum payoff function at the n^{th} step of the negotiation process.

Proof. According to Lemma 1, in a systematic belief update process, the learner eventually comes arbitrarily close to the true distribution after a sufficiently large time T. Since the process by which our learning agent estimates the strategy of the opponent is constructed as a Bayesian belief process, the sequence of updated probabilities, $\{Pr\{H_n^0(t)|O_n^p(t)\}, ..., Pr\{H_n^i(t)|O_n^p(t)\}, ..., Pr\{H_n^k(t)|O_n^p(t)\}\}$ comes arbitrarily close to $\{0, ..., 1, ..., 0\}$ for some $i \in \{0, 1, 2, ..., k\}$ as $t \rightarrow T$. This implies that $H_n^i(t)$ is the true hypothesis. Therefore, $H_n^{new}(t) = \sum_k^{i=0}\{Pr\{H_n^i(t)|O_n^p(t)\} \times H_n^i(t)\}$, by construction, and, therefore, $H_n^{new}(t) \rightarrow H_n^i(t)$ as $t \rightarrow T$. Since the opponent determines its strategy at each step using $H_n^{new}(t)$ and since our agent determines its optimal strategy and the maximum payoff at each step in response to this updated opponent strategy, the result is proved. □

Thus we have analytically shown that our algorithm converges. We now present an example to illustrate this operation (and in section 4 we explore the actual speed of convergence). Specifically, in this problem, we make the following assumptions:

1. The strategy space S of the opponent consists of two strategies, $S = \{s_1, s_2\}$.
2. At each time step n of the negotiation, the learning agent has a set of three hypotheses about the possible value of $\mathbf{P^n(t)}$.

We now describe the solution procedure in our problem.

– Here as specified in Step 3 of algorithm 1, we initialize, $H_0^i(0) \in H_0(0)$.

$$H_0^1(0) = \begin{bmatrix} 0.5 & 0.5 \\ 0.5 & 0.5 \end{bmatrix}$$

$$H_0^2(0) = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$H_0^3(0) = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

Now the agent is unaware of the true value of $\mathbf{P^n(t)}$, but has arbitrary probabilities assigned to each of these hypotheses about $\mathbf{P^n(t)}$.

– Let $\{Pr(H_0^1(0)) = 0.5\}$,$\{Pr(H_0^2(0)) = 0.25\}$, $\{Pr(H_0^3(0)) = 0.25\}$ and offer of opponent, $O_0(0) = 100$.
– Then according to Step 4 in algorithm 1, we observe the offer of the opponent and assume that $Pr\{O_0(0)|H_0^1(0)\} = 0.6$, $Pr\{O_0(0)|H_0^2(0)\} = 0.2$ and $Pr\{O_0(0)|H_0^3(0)\} = 0.2$ (here we assume arbitrary values for $Pr\{O|H\}$, but we are in the process of studying the offer patterns of traders in different domains in order to get an accurate representation for these values).
– Using Step 5, we update probabilities as $Pr\{H_0^1(0)|O_0(0)\} = 0.75$, $Pr\{H_0^2(0)|O_0(0)\} = 0.125$ and $Pr\{H_0^3(0)|O_0(0)\} = 0.125$.
– From Step 6, we determine $H_0^{new}(0) = (0.75) \times H_0^1(0) + (0.125) \times H_0^2(0) + (0.125) \times H_0^3(0)$.

– Step 7 specifies the strategy profile of the opponent. The initial profile is assumed to be given as $[0.2, 0.8]$ (i.e., $Pr\{S = s_1^o\} = 0.2, Pr\{S = s_2^o\} = 0.8$) and the payoff function of the agent is:

$$u_x^0(0) = \begin{bmatrix} 1 & 0 \\ -2 & 3 \end{bmatrix}$$

– From Step 8, to determine its strategy profile, $[s_1, s_2]$ and the corresponding payoff function, the agent solves the linear program:
max $[s_1, s_2] \times u_x^0(0) \times [0.2, 0.8]^T$ *s.t* $s_1 + s_2 = 1$ *and* $s_1, s_2 \geq 0$.
The strategy profile thus obtained is denoted as $s_{max}^0(0)$ and the payoff function is $u_{max}^n(0)$.
– We do this for every time step n of the negotiation process and obtain $\{s_{max}^0(0), s_{max}^1(0), ...\}$ and $\{u_{max}^0(0), u_{max}^1(0), u_{max}^2(0), ...\}$
– We then repeat this for every negotiation during time instants $t = 0, 1, 2, 3, ...$ and obtain the sequences $\{s_{max}^0(0), s_{max}^0(1), s_{max}^0(2), ...\}$, $\{s_{max}^1(0), s_{max}^1(1), s_{max}^1(2), ...\}$, $\{s_{max}^3(0), s_{max}^3(1), ...\}$ and the corresponding payoff sequences.

In this case the two sequences converge within 4 iterations to the optimal values at each step of the negotiation process. Having proved that our algorithm converges, we need to determine the rate of this convergence and the factors that influence this. This can only be done empirically and our results are presented in the next section.

4 Empirical Results

We have run experiments by varying the number of hypotheses for $\mathbf{P^n(t)}$. Specifically, we have experimented with both 2×2 and 3×3 matrices representing the strategy profiles. We have found that using our algorithm, the agent on an average (computed by varying the number of hypotheses ie., the number of 2×2 and 3×3 matrices) learns the maximum payoff within 13.6 iterations for 2×2 matrices and on an average within 23.6 iterations for 3×3 matrices. On an average, each iteration takes 7.44 seconds to complete for 2×2 matrices and 18.92 seconds for 3×3 matrices. We have also experimented by changing the elements of the transition matrices. Our results indicate that the rate of convergence is independent of variations in the patterns of the opponent's offers. But it depends on the number of hypotheses for $\mathbf{P^n(t)}$ at each step of the negotiation. However, since the computation of updated probabilities is a simple arithmetic operation and since this computation alone is affected by the number of matrices, the time for convergence does not drastically increase with the number of matrices. This is shown in tables 1 and 2 and figure 1 shows the actual convergence of the algorithm during successive iterations at a *single step* of the negotiation process. In figure 1 the optimal value line for the payoffs is computed by assuming that the opponent's strategy profile is known to the agent. The other line corresponds to the agent learning the opponent's strategy profile and therefore the optimal payoff value, using our algorithm, as illustrated in the example problem. In figure 1 we used 2×2 matrices to represent the opponent's strategy profiles and assumed 2 hypotheses for $\mathbf{P^n(t)}$ at each step of the negotiation.

Now, in general, if we assume that there are k hypotheses for $\mathbf{P^n(t)}$, then in the random case the probability of finding the true hypothesis is always $1/k$ (i.e., this probability does not improve with time). However, the convergence of our algorithm is guaranteed by theorem 2 and therefore our estimation of the true $\mathbf{P^n(t)}$ improves with each iteration and eventually converges. With this and the fact that the algorithm converges on an average within 1315 seconds even with 10 hypotheses and 3 strategies, we claim that our algorithm is k times more effective than random estimation. Therefore even in the case when there are only 2 hypotheses at each step n for $\mathbf{P^n(t)}$ our algorithm is 200% more effective than random estimation. Obviously, as we increase the number of hypotheses, which allows for a more general representation of $\mathbf{P^n(t)}$ and therefore of the uncertainty in the problem, the effectiveness of our algorithm over random estimation increases proportionally.

No of Hypotheses	Offer Distribution	Average Iterations	Average Time for Iteration in secs
2	Arbitrary	2	2.7
4	Arbitrary	4	4.6
6	Arbitrary	11	7
8	Arbitrary	21	10.1
10	Arbitrary	30	12.8

Table 1. Dependence of rate of convergence on number of hypotheses for 2×2 matrices

No of Hypotheses	Offer Distribution	Average Iterations	Average Time for Iteration in secs
2	Arbitrary	7	6.2
4	Arbitrary	15	12
6	Arbitrary	25	18.4
8	Arbitrary	29	26.7
10	Arbitrary	42	31.3

Table 2. Dependence of rate of convergence on number of hypotheses for 3×3 matrices

5 Conclusions and Future Work

In this work we have developed a new framework, using Markov chains, for studying negotiation in non-stationary environments. This is a general framework which can be used to study decision making in many stochastic systems like market places and auctions. Within this framework, we have derived, for the first time, an important result for non-stationary Markov chains that computes the distribution of the random variable, which defines it, at any future step of the process given its initial probability distribution and the transition probability matrices at each step of the process. Then, using this framework we have developed an algorithm to learn a strategy in response to a

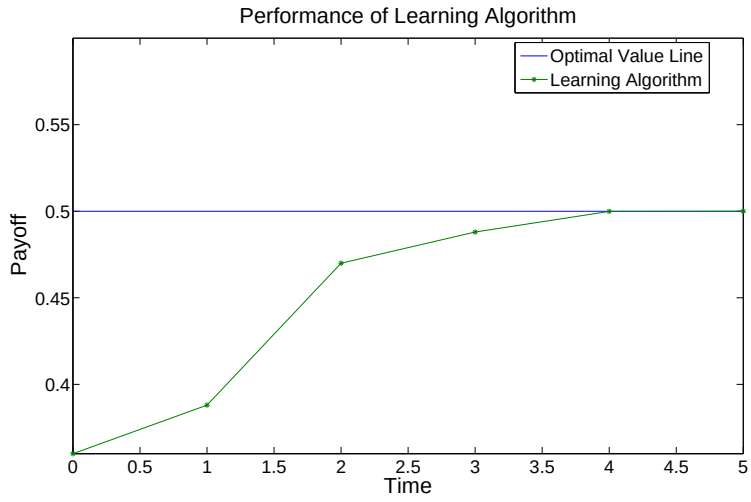

Fig. 1. Convergence to Optimal Strategy

non-stationary opponent's play and proved that it converges to the optimal strategy in repeated negotiations. Unlike previous work in this area, our algorithm does not assume knowledge of the opponent's strategy profile and, as such, is a powerful tool to analyse negotiations in real world environments where such uncertainty is common. Our algorithm is also explicitly designed to deal with cases in which the strategy profile is not only unknown, but changes with time during the course of the negotiation process itself. This significantly extends the state of the art in the field of automated negotiations in non-stationary, real world environments. For such cases, we have proved, analytically, that our algorithm converges. Our empirical results indicate that the algorithm converges within reasonable timeframes to the true hypothesis even when the number of hypotheses is large. Also, in our algorithm, the states of the Markov chain represent the strategies that are available to the opponent and the actual uncertainty in the problem is represented by the number of hypotheses. Therefore we need not increase the number of states to represent greater uncertainty in the domain. The algorithm is also vastly more accurate than random estimation.

In our future work, we intend to use the structure of the Markov chain to develop a more formal model for the likelihood function in the Bayesian belief update process which would help us to reduce the computation effort involved in updating the agent's knowledge even in complicated real world scenarios. We are also in the process of doing statistical analysis to estimate the number of hypotheses that are required to get an accurate representation of the dynamism in exemplar real world domains like mobile communications and we also intend to extend the algorithm to learn the opponent's negotiation parameters along with its strategy profile and to study negotiation when the opponent is also changing its strategies in response to our agent's adaptivity. Finally, we intend to do a detailed comparative study between other machine learning techniques, like Reinforcement learning and Bayesian methods in automated negotiations, for non-

stationary environments. This study would also provide a more effective benchmark than random estimation for our algorithm.

References

1. J. Filar and K. Vrieze. *Competitive Markov Decision Processes*. Springer, 1996.
2. D. Fudenberg and D.K Levine. *The Theory of Learning in Games*. The MIT Press, 1998.
3. D. Fudenberg and J. Tirole. *Game Theory*. MIT Press, 1991.
4. J. Hu and M.P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. *Proceedings of the 11th International Conference on Machine Learning: 242-250*, 1998.
5. E. Kalai and E. Lehrer. Rational learning leads to nash equilibrium. *Econometrica, 61(5):1019–1045*, 1993.
6. S. Karlin and H. Taylor. *First Course in Stochastic Processes*. Academic Press, 1974.
7. S. Kraus and V.S. Subrahmanian. Multiagent reasoning with probability, time, and beliefs. *International Journal of Intelligent Systems 10(5): 459-499*, 1995.
8. V. Kulkarni. *Modelling and Analysis of Stochastic Systems*. Chapman Hall/CRC, 1996.
9. M.L. Littman. Markov games as a framework for multi-agent reinforcement learning. *Proceedings of the 11th International Conference on Machine Learning: 157-163*, 1994.
10. V. Narayanan and N. R. Jennings. An adaptive bilateral negotiation model for e-commerce settings. *Proc. 7th Int. IEEE Conf on E-Commerce Technology, Munich, Germany: 34-39*, 2005.
11. R.S. Sutton and A.G. Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.
12. M. Weinberg and J. Rosenschein. Best-response multiagent learning in non-stationary environments. *The Third International Joint Conference on Autonomous Agents and Mutli-Agent Systems:506-513*, 2004.
13. D. Zeng and K. Sycara. Bayesian learning in negotiation. *Int. J Human-Computer Studies(1998), 48:125-141*, 1998.