An Adaptive Bilateral Negotiation Model for E-Commerce Settings

Vidya Narayanan and Nicholas R. Jennings
Intelligence, Agents, Multimedia
School of Electronics and Computer Science
University of Southampton SO17 1BJ, UK.
$\{vn03r, nrj\}$@$ecs.soton.ac.uk$

Abstract

This paper studies adaptive bilateral negotiation between software agents in e-commerce environments. Specifically, we assume that the agents are self-interested, the environment is dynamic, and both agents have deadlines. Such dynamism means that the agents' negotiation parameters (such as deadlines and reservation prices) are functions of both the state of the encounter and the environment. Given this, we develop an algorithm that the negotiating agents can use to adapt their strategies to changes in the environment in order to reach an agreement within their specific deadlines and before the resources available for negotiation are exhausted. In more detail, we formally define an adaptive negotiation model and cast it as a Markov Decision Process. Using a value iteration algorithm, we then indicate a novel solution technique for determining optimal policies for the negotiation problem without explicit knowledge of the dynamics of the system. We also solve a representative negotiation decision problem using this technique and show that it is a promising approach for analyzing negotiations in dynamic settings. Finally, through empirical evaluation, we show that the agents using our algorithm learn a negotiation strategy that adapts to the environment and enables them to reach agreements in a timely manner.

1. Introduction

Automated negotiation is a key issue for e-commerce because it provides the de facto means of interaction between stakeholders with different aims and objectives [4]. Given this, many different models with many different properties have been developed (covering a wide variety of auction types and the direct negotiation and bargaining mechanisms we focus on in this paper). However, a key challenge that appears in many real world applications, but that is often neglected in such work, is that of negotiating effectively in dynamic environments. Here, by dynamism, we mean that the very structure of these systems is subject to change with new agents being added and removed constantly, that the computational and monetary resources available for carrying out the negotiation are limited and can fluctuate, and that the deadline by which the negotiations must be completed might change. To rectify this, we develop a new model for automated negotiation (between pairs of agents) in which agents can adapt their negotiation strategy as such changes occur. In particular, we have developed a novel negotiation algorithm using which the agents can learn to react appropriately to such situations.

By means of an illustration, consider, a typical business-to-consumer (B2C) scenario, in which an agent (which forms a part of a large network of agents), acting on behalf of a retailer, negotiates online with a specific consumer for the price of a certain product. As the retailer has many such agents acting on its behalf, they share resources like bandwidth for communication and computational power with one another and they reside in the same computational space. Now, in such situations an agent might suddenly find that its messages take longer to reach its opponent because of increased network activity or that it has to wait longer for CPU time since one of the other agents is using a large number of CPU cycles. This means the resources available to carry out the negotiation change, either increasing or decreasing, depending on the nature of the environmental changes. In yet other cases, an agent's allocated budget for acquiring the desired service might be cut. For instance, in a large supply chain (implemented as a multi-agent system), where several agents pursue independent goals, the budget spent in one part of the supply chain may well impact the budget further down the line (because the agent is suddenly given more money as a result of savings or because it is given a lower budget as a result of overspending). In this case, in negotiation terms, the agent's reservation price changes. Finally, in view of the goal of the entire supply chain system, the agents may need to shorten or lengthen the time available for them to complete their negotiation because earlier activities take a shorter or longer

time to procure than was initially expected. In this case the agent's deadline changes. Now in all these cases, the agents need to adapt their negotiation strategies if they are to be effective in their changed environment. Failure to adapt, may lead to poor outcomes and may leave the owner less satisfied.

Against this background, in this paper we concentrate on single issue negotiation, between a buyer and a seller, each of which is trying to maximize their return and each of which have their own private deadlines. In this setting, we need a mechanism by which the agent can learn to negotiate without significant prior knowledge of the system or its opponent (because this is the nature of many e-commerce negotiations). To this end, reinforcement learning enables an agent to learn the preferences of its opponent and the state of the environment without the aid of a model [9]. In this vein, [5] uses the framework of two player zero sum Markov games to describe interactions between adaptive agents with opposing goals that share an environment. They also develop a Q-learning algorithm to determine optimal policies in this context. Then, in [3], an extension of this approach is given to general sum games. Here the agents first determine a mixed-strategy Nash equilibrium profile for the game and then use this profile in the Q-learning algorithm to determine an optimal policy. Moreover, speaking more generally, both these approaches share the following key assumptions: (i) both agents share the same environment, (ii) both agents can observe the entire state space and the payoffs received, the state space, action space, reward function and the optimal policy are all stationary and (iii) the agents optimize for an infinite time horizon. However, in our case, the agents need to make decisions based on deadlines and reservation prices that are private information, therefore these parameters cannot be part of a common state space. Again, in [2], a finite horizon Q-learning algorithm is described for non-stationary processes which is suitable for the negotiation process, but this algorithm assumes that only a single agent has to adapt to changes in the environment (which is clearly not the case in our scenario). Moreover, in general, these game theoretic models assume that the agent's decision is solely based on its opponent's action and ignores changes in the environment which is unreasonable for the e-commerce settings we wish to tackle.

Therefore we need a model in which the agents choose strategies at each step of the negotiation based on the current state of the environment and that will result in achieving its long term objectives (e.g. reaching an agreement before its deadline). Given this background, we have modelled the negotiation as a set of two non-stationary Markov Decision Processes (MDPs). There are two processes because each agent has its own view of the state space and the dynamics of the environment. The process is non-stationary because the probabilities of transition from one state to another vary over time as a consequence of the environmental dynamics. In typical e-commerce domains, however, not only does the state of the system change (e.g. resource availability), but also the agents in these domains are unaware of the pattern of variability. We model this by assuming that the agents have probabilistic knowledge of the transition function. Specifically, we view the negotiation as Markovian since we believe that effective strategies can be chosen based only on the current state of the system and independently of the history of the negotiation process [1]. Given this, we have then developed a negotiation mechanism using a value iteration algorithm we have devised for this process where the response of the agent depends on factors like resource availability, time availability and the attitude (conceding or stubborn) that it adopts during the negotiation. Our work advances the state of the art in the following ways:

1. it develops a mathematical framework for adaptive negotiations in the e-commerce domain using Markov Decision theory.
2. it develops a novel automated mechanism to negotiate adaptively in these domains.
3. it also shows, by means of empirical evaluation, that the algorithm performs better in dynamic environments than a non-adaptive algorithm.

The remainder of the paper is organized as follows. Section 2 describes the requirements, assumptions and the components of our negotiation model. Section 3 describes the solution procedure and the algorithm used. Section 4 presents our empirical results and Section 5 concludes.

2. Modelling Adaptive Negotiation

Our overarching aim in this work is to design a negotiation mechanism for agents operating in dynamic e-commerce environments. To do so, we first characterize the environment in which the agents function:

1. The agents negotiate in an environment whose dynamics are unknown. That is, the resource and the time available for negotiations can change.
2. The negotiation outcome depends on the resources available for negotiation, the negotiation parameters (e.g. deadlines and reservation prices), and the negotiation strategies of the agents. These are all subject to change (as exemplified in Section 1).
3. The agents are unaware of their opponent's parameters (e.g. deadlines or utility functions).

1 This is clearly an assumption that needs to be evaluated in future work.

4. The agents cannot directly observe changes in their opponent's parameters or their payoffs. They can only observe the changes indirectly through the negotiation actions of their opponent.

We can now turn to the underlying assumptions of our negotiation model:

- We consider two agents (designated as buyer b and seller s), bargaining over a single issue (i.e., the price of a service.)
- Agents are aware of their own negotiation parameters; namely, their own deadline, $T^a_{deadline}$, and their reservation price RP^a which is the maximum (minimum) price that the buyer b (seller s) can offer. But they are unaware of their opponents' parameters (i.e., the agents have incomplete information).
- The interval $[RP^s - RP^b]$ is called the *the zone of agreement*. For an agreement to be reached this zone must be non-empty. In our case, the agents do not know this zone (or even whether it is non-empty) and moreover it can change during the course of the encounter.
- The agents alternate in making offers and these offers are made at discrete time points in the set $\{T = 0, 1, ..., T^a_{deadline}\}$.
- The agents seek to reach an agreement before their deadline is reached. Failure to conclude the negotiation before this time is the worst possible outcome.
- The agents cannot opt out of the negotiation process and so the negotiation terminates either when an agreement is reached or when one of the deadlines passes.

Having studied the assumptions, we can now characterize the main components of our model [6]:

- *The Negotiation Protocol*: Formally specifies the rules of the negotiation process — who can participate, the states of the negotiation process and some of the events that change the state of the negotiation process. In our case, the agents alternate in making offers until an agreement is reached, (hence the use of the *alternating offers protocol* [1]).
- *The Negotiation Objects*: Represent the issues over which the agents are negotiating. In our model the agents negotiate over a single issue (i.e., the price of a good or service).
- *The Participants' Negotiation Preferences*: These represent the objectives of the agents participating in the negotiation process. In our model, the agents' broad objectives are to reach an agreement on the price of the service that maximizes their return before their deadlines are reached or before their resources are exhausted.
- *The Participants' Negotiation Strategies*: These specify how an agent should respond to a given situation. In our case these strategies enable the agent to negotiate effectively by adapting to their environment.
- *The Participants' Negotiation Parameters*: These represent the deadlines of the agents and their reservation prices.

In our setting of incomplete information and variable parameters, the agents have to devise strategies for reaching an agreement. As argued previously, the agent has to decide on the best course of action given the current state of the system. Now, since the state space is discrete and has the memoryless property we cast the negotiation as a MDP and use a value iteration algorithm for determining a strategy for dynamic negotiations.

3. The Adaptive Negotiation Model

This section outlines our adaptive negotiation model. We first recap some basic definitions of MDPs and value based iteration methods for solving MDPs that form the foundation of our model (section 3.1). We then describe the structure of our model (section 3.2), before going onto the negotiation algorithm itself (section 3.3).

3.1. Basic Definitions

3.1.1. The Markov Property. The negotiation process analyzed in this paper is assumed to be a *Markov* process. Intuitively, a process is Markovian if and only if the state transitions depend only on the current state of the system and are independent of all preceding states. Formally, the sequence of random variables $\{X_n, n = 0, 1, 2, ...\}$ is defined to be a Markov process iff their conditional probability density function, P, satisfies the following relationship [8]:

$$P\{X_n|X_1, X_2, ..., X_{n-1}\} = P\{X_n|X_{n-1}\} \qquad (1)$$

Then a process that uses this property of the state space to analyze all decisions, based on a reward scheme, that need to be made within this space is called a Markov Decision Process. Now, the specific problem that we wish to consider is set in non-stationary environments where the dynamics of the system vary with time (as argued in section 1). Thus the associated decision process is also non-stationary and we are in the realm of non-stationary MDPs.

3.1.2. Non-Stationary MDPs. A non-stationary MDP for each time-step, n, is defined as [2]:

- a discrete state space S_n
- a set of discrete actions A_n

- a reward function $R_n : S_n \times A_n \rightarrow \Re$
- a probabilistic state transition function, $T_n : S_n \times A_n \rightarrow [0,1]$, $T_n(s,\ a,\ s')$ is defined as the probability of making a transition from state s_n to state s_{n+1} using action a_n.

In a standard MDP, an agent tries to find a policy $\pi : S \rightarrow A$ that maps an action, a, to a state, s, and maximizes its expected sum of discounted rewards over an infinite period of time. However, in our negotiation context, the agents have finite deadlines and, therefore, we define the corresponding notion of maximizing expected rewards for a finite time horizon. In this case, the policy π can be decomposed into a set $\pi_1, \pi_2, ..., \pi_N$ where $\pi_n : S_n \rightarrow A_n$. As the first step towards determining the optimal policy we introduce the notion of the value of a state. Formally, a *value* of a state $s \in S_n$, under a policy π_n, during the n^{th} time-step, is defined as:

$$V_n^{\pi}(s) = \sum_{t=n}^{N} E(R_t(s_t, \pi_t(s_t))|s_n = s) \tag{2}$$

where s_n is the state of the system at time-step n, R_n is the reward obtained at time step n, and $E(R_t(s_t, \pi_t(s_t))|s_n = s)$ is the expected value of the reward under policy π_n and state $s_n = s$. The optimal policy is denoted by π^* and the associated value function is given by:

$$V_n^*(s) = max_a[R_n(s,a) + \sum_{s' \in S_{n+1}} T_n(s', a, s) \times V_{n+1}^*(s')] \tag{3}$$

for all $s \in S_n$, $n \in 1, ..., N$ and $V_{N+1}^* = 0$. The optimal policy is specified by $\pi_n^*(s) = a$ where a is the action at which a maximum is attained in equation 3. Now, when the dynamics of the system are known, the optimal value function can be solved by standard dynamic programming techniques [9]. However, in the our negotiation problem, the probability of state transitions (changes in the dynamics of the system) are not known exactly but are themselves specified by another non-stationary probability function P_n called the estimate function. We define this function as:

$$P_n(s,\ a,\ s') : T_n(s,\ a,\ s') \rightarrow [0,1] \tag{4}$$

Given this, we have developed value iteration algorithm based on the average or expected $T_n(s,\ a,\ s')$ values given by:

$$E_n(T_n(s, a, s')) = \sum_{s'} (P_n(s, a, s')) \times (T_n(s, a, s')) \tag{5}$$

3.1.3. Average Value Iteration. Here we describe the key notions used in developing an adaptive negotiation model based on an average value iteration method. Towards this end, we first define the average value function and the Q-values for the states of the system [9]:

$$V_n^*(s) = max_a[R_n(s,a) + \sum_{s' \in S_{n+1}} E_n(T_n(s', a, s)) \times V_{n+1}^*(s')] \tag{6}$$

$$Q_n^{\pi}(s,a) = \{R_n(s,a) + \sum_{s' \in S_{n+1}} E_n(T_n(s, a, s')) \times V_{n+1}^{\pi}(s')\} \tag{7}$$

The corresponding optimal Q-function is given by:

$$Q_n^*(s,a) = \{R_n(s,a) + \sum_{s' \in S_{n+1}} E_n(T_n(s, a, s')) \times V_{n+1}^*(s')\} \tag{8}$$

for all $s \in S_n$ and $a \in A_n$.

Now from equations (6) and (8) we have:

$$V_n^*(s) = max_a[Q^*(s,a)] \tag{9}$$

Then once the Q-value for each state, s, is determined using the average value iteration algorithm, the agent will deterministically choose the action, a, that maximizes the Q-value (i.e., assign $\pi^*(s,a) = 1$).

3.2. The Structure of the Dynamic Negotiation Model

For reasons outlined earlier, we base our negotiation model on MDPs. We formally define the Markov negotiation set as composed of two Markov decision processes: $(S_n^1, A_n^1, P_n^1, R_n^1)$ and $(S_n^2, A_n^2, P_n^2, R_n^2)$ where S_n^a is the non-stationary discrete finite state space for agent a, A_n^a is the discrete finite action space for agent a, P_n^a is the estimate function for agent a, and R_n^a is the reward function for agent a, at time instant n. Now the state space must include all the factors that have an impact on the decision-making of the agents. In our case this includes:

1. environmental considerations like resource availability

2. the agent's reservation price (RP) and deadline ($T_{deadline}$)
3. the opponent's offer.

The agent makes an offer based on its state space and using the opponent's action as an input to make decisions.

In more detail, the agents use negotiation decision functions (NDFs) [7] to generate offers (since these have been developed specifically for negotiations in incomplete and time constrained environments). Formally, these are mathematical functions that generate values between the initial offer and the RP of the agent. These functions were chosen in our model because they enable us to control the rate at which the agent's offers approach its RP depending on the currently available resources, the current reservation price and the other identified factors. Using this model, the offer of the agent a to its opponent $\hat{a}$, $p^t_{a\rightarrow\hat{a}}$, is defined in terms of the NDF as:

$$\begin{aligned} p^t_{a\rightarrow\hat{a}} &= IP^b + f^b(t)(RP^b - IP^b) for\ buyer\ b \\ &= RP^s + (1 - f^s(t))(IP^s - RP^s) for\ seller\ s. \end{aligned}$$

Here IP^b, RP^b, IP^s and RP^s are the initial and reservation prices of the buyer and seller respectively and $f^a(t)$ represents the NDF of agent a. These functions are such that $0 \leq f^a(t) \leq 1$, $f^a(0) = k^a$ (a pre-defined constant which determines the initial offer of agent, a), and $f^a(T^a) = 1$.

Keeping these requirements in mind, the NDF for agent a is defined as:

$$f^a(t) = k^a + (1 - k^a)(min(t, T^a)/T^a)^{1/\psi} \qquad (10)$$

In fact, this represents a family of NDFs defined by the parameter ψ. From this, it can be seen that $p^t_{a\rightarrow\hat{a}}$ tends to RP^b as t tends to T and that $p^t_{a\rightarrow\hat{a}}$ tends to RP^s as t tends to T. In both cases, the parameter ψ controls the rate at which the NDF approaches the value 1, which, in effect, controls the rate at which the offer of the buyer and seller reach their respective reservation prices. A more detailed description of the variation of the behaviour of agents with the parameter ψ is given below:

1. *Conceder*: When $\psi > 1$: The agent quickly reaches its reservation value. The agent employs this strategy when time or resources for negotiation are limited.
2. *Boulware (Stubborn)*: When $\psi < 1$: The agent maintains its initial offer until the deadline is almost reached and then concedes quickly.

Thus the agent has two broad classes of strategy that it can adopt during the negotiation process. The key strategic decision is which of them has to be adopted and at what time. Using our algorithm (detailed in section 3.3), the agent will endeavour to appropriately map its negotiation actions to situations so that an agreement is reached before the deadline and before the resources available for negotiation are exhausted. Intuitively this is achieved by rewarding the agent when it adopts a stubborn approach when there are adequate resources for the negotiation process and, correspondingly, rewarding the agent for adopting a conceding approach when the available resources are low. The selection of the appropriate action is based not simply on the immediate reward that the agent will obtain, but takes into consideration all possible future rewards.

3.3. The Adaptive Negotiation Algorithm

In this section we outline the steps of the average value based iteration algorithm based on observations of the state space, actions of the agents and the reward signals (see Algorithm 1). In more detail, the agent must decide what policy to adopt depending on the state of the system. To do so, it first observes its opponent's offer. The agent then specifies $V^*(s)$ for the final states (i.e., states representing the fact that the deadline is reached, that resources are exhausted or that an agreement is reached). Then at each time instant it iteratively computes $V^*_n(s)$ using equation (6) for all other states. It determines the current state s of the system and chooses the action a (i.e., Conceder or Boulware) that maximizes equation (9) and appropriately chooses the parameter ψ. Using this value of ψ in equation (10), it determines an offer according to definition in section 3.2. The opponent observes this offer and determines a counter-offer using the same algorithm and based on its state space and reward definitions. The process terminates when an agreement is reached or when either deadline is reached.

4. Solving the Negotiation

In this section we will illustrate this algorithm using an example negotiation scenario. To do so, however, we first describe the state and action spaces for the agent.

4.1. State and Action Spaces

The finite discrete state space of each agent in the negotiation process is defined by

1. Resource Availability: This denotes the computational resources that are available for the negotiation. Here we assume this can take two values (high and low). Thus the agent adapts its behaviour according to changes in its resource availability.
2. Deadlines: The agent has a finite discrete set of deadlines. This represents the fact that its deadline can

Algorithm 1 Adaptive Negotiation Algorithm

Observe

1. the offer of the opponent $p^t_{a \to a}$.
2. the current state s_n of the system.

Specify the current estimation function $P_n(T_n(s', a, s))$ based on the partial knowledge of the system for each state s and instant n.
Specify the reward function $R_n(s, a)$ for each (s, a), $s \in S_n$ *and* $a \in A_n$.
Specify $V^*_n(s)$ for terminal states (i.e., agreement reached, deadline reached).
Compute iteratively $V^*_n(s) = max_a[R_n(s, a) + \sum_{s' \in S_{n+1}} E_n(T_n(s', a, s)) \times V^*_{n+1}(s')$ for all other states.
Choose action a that maximizes $V^*_n(s)$
if $a = Boulware$ **then**
  Set parameter $\psi < 1$
else $\{a = Conceder\}$
  Set parameter $\psi > 1$
end if
Use parameter ψ to determine the NDF $f_s(n)$ and generate offer $p(n)$.
Terminate when terminal states are reached.

change. For our example negotiation problem, we assume that state space consists of two deadlines: T^1, T^2 time units.

3. Reservation Price: The agent has a discrete, finite set of reservation prices. This again means that the reservation prices of an agent can vary. In the example problem we will assume that the agent has two reservation prices: R^1, R^2.

Now, depending on the state of the system and the input (offer) that the agent receives from its opponent, the agent chooses between a stubborn and a conceding strategy. It also has the option of doing nothing (i.e, making no response) since the negotiation is a process of alternating offers, at alternate time-steps when it is the opponents's turn to make an offer the agent does nothing.

4.2. Estimation and Reward Functions

The agent should be rewarded for choosing a stubborn strategy when the resources are high and when the agent has sufficient time to continue with the negotiation process (as argued for in section 3.2). Similarly, it should be rewarded when it adopts a conceding approach when the resources for negotiation are low. Also the agent should adapt to changes in its RP.

There are a total of 8 (2 Deadlines $\times$ 2 Reservation Prices $\times$ 2 states of Resource Availability) states and 64 state transitions for each action a. There is an estimation function $P(s', a, s,)$ and a reward function $R^a_{ss'}$ associated with every state transition and action. We have specified estimation probabilities and the reward scheme for a few such transitions in table 1. In a similar fashion they can be described for all the remaining state transitions.

s	$s\prime$	a	$P(s', a, s,)$	$R^a_{ss'}$
(low,T^1,R^1)	(low,T^1,R^1)	C	P^C	R^C
low,T^1,R^1)	(high,T^1,R^1)	C	P^C	R^C
(low,T^2,R^1)	(high,T^2,R^1)	B	P^B	R^B
(low,T^2,R^1)	(high,T^2,R^2)	B	P^B	R^B

Table 1. Reward Scheme

4.3. Results

Having detailed the model's instantiation, we now consider its effectiveness. To do this, for a specific set of negotiation parameters (T^a, RP^a, IP^a) we allowed a buyer and a seller to negotiate and measured the value at which an agreement was reached and the time taken to get there. In particular, we consider negotiation in the face of varying resource availability, two different RPs and two different Ts (since, as outlined in section 1, these two factors are the key drivers in e-commerce settings). Using the adaptive algorithm, the agent, at each state of the negotiation process, autonomously determines the optimal action (the one that yields the maximum future reward). This translates into choosing the parameter, ψ, that governs the offer, p, that the agent makes.

To provide a benchmark for our algorithm we compare it against an agent that uses NDFs to determine the negotiation strategy, but that does not adapt its strategy in response to changing resource availability, deadlines or reservation prices during the course of the encounter. The particular strategy we compare it against involves the agent adopting either a Boulware strategy or a Conceder strategy throughout the negotiation process irrespective of changes in resource availability. We experimented with different values of negotiation parameters and determined that our adaptive strategy consistently performed better than the non-adaptive strategy. On an average the time taken to reach an agreement using the adaptive strategy was 54% less than the time taken by the non-adaptive strategy. This improvement is a direct result of choosing a strategy in response to changes in the environment (Conceding when resources and time availability as specified by the deadlines are low and Stubborn otherwise). To illustrate this, we have plotted the course of the negotiation process in figure 1 for a specific set of negotiation parameters given in table 2. The results of the comparison are

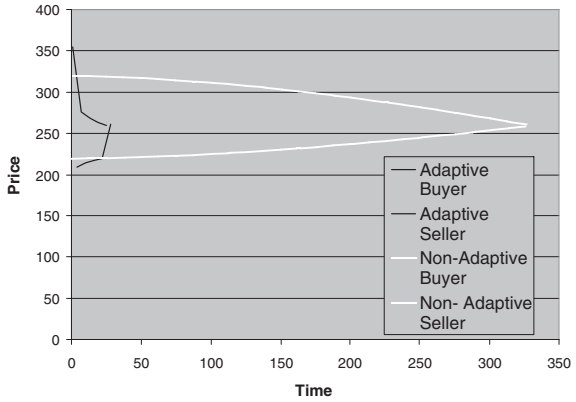

Figure 1. Adaptive vs. Non-Adaptive Strategy

given in table 3. As can be seen, the value of the agreement is about the same in both the adaptive and the non-adaptive cases. This is because while specifying the reward scheme we have assumed that it is more important to reach an agreement before the deadline than the actual value at which the deadline is reached (this could easily be changed to give more importance to the actual outcome attained simply by specifying an alternate reward scheme).

Agent	Buyer	Seller
Initial Price	100	500
Reservation Price	300 or 250	250 or 200
Deadline	600	500

Table 2. Negotiation Parameters

Strategy	Adaptive	Non-Adaptive
Time of Agreement	29	327
Value of Agreement	260	260

Table 3. Comparison Results

5. Conclusions and Future Work

In this paper we have developed a model for adaptive bilateral negotiation that is suitable for dynamic e-commerce environments. Specifically, we have used a non-stationary value iteration algorithm to determine non-stationary negotiation strategies when the dynamics of the system are only probabilistically known. This negotiation model can adapt the agent's strategy in response to resources availability and variation in negotiation parameters (deadlines and reservation prices). We believe that this represents an important step forward in the field of bilateral negotiations in that our mechanism gives the agents a method by which they can negotiate effectively in a variety of situations in which the dynamics of the system are not completely known. In particular, this helps the agent to function in dynamic environments, where it is impossible to know at the start of the negotiations all the states that the agent might encounter. We have illustrated this approach by solving a representative negotiation problem. Initial results show that the time taken to reach an agreement can be significantly reduced compared to a non-adaptive strategy.

In future work we will extend the algorithm to deal with situations in which there is complete ignorance about the system dynamics (as opposed to the present probabilistic assumption) where the agent learns an optimal policy by repeatedly negotiating with its opponent. We will extend the algorithm to cover the convergence aspects of our learning algorithm and will evaluate its suitability and effectiveness for online learning.

6. Acknowledgement

The work reported in this paper has formed part of the PDE area of the Core 3 Research Programme of the Virtual Centre of Excellence in Mobile & Personal Communications, Mobile VCE, www.mobilevce.com, whose funding support, including that of EPSRC, is gratefully acknowledged. Fully detailed technical reports on this research are available to Industrial Members of Mobile VCE.

References

[1] A.Rubinstein. Perfect bargaining in a bargaining model. *Econometrica,50:97–110*, 1982.

[2] F.Garcia and S.M.Ndiaye. A learning rate analysis of reinforcement learning algorithms in finite horizon. *Proceedings of the 15th International Conference on Machine Learning (ML-98)*, 1998.

[3] J.Hu and M.P. Wellman. Multiagent reinforcement learning: Theoretical framework and an algorithm. *Proceedings of the 15th International Conference on Machine Learning (ML-98)*, 1998.

[4] M.He, N.R.Jennings, and H.Leung. On agent-mediated electronic commerce. *IEEE Trans on Knowledge and Data Engineering, 15(4): 985-1003*, 2003.

[5] M.L.Littman. Markov games as a framework for multi-agent reinforcement learning. *Proceedings of the 11th International Conference on Machine Learning (ML-94)*, 1994.

[6] N.R.Jennings, P.Faratin, A.Lomuscio, S.Parsons, C.Sierra, and M.Woolridge. Automated negotiation:prospects, methods and challenges. *International Journal of Group Decision and Negotiation, 10(2): 199-215*, 2001.

[7] P.Faratin, C.Sierra, and N.R Jennings. Negotiation decision functions for autonomous agents. *International Journal for Robotics and Autonomous Systems*, 1998.

[8] R.A.Howard. *Dynamic Programming and Markov Processes*. The MIT Press, 1960.

[9] R.S.Sutton and A.G.Barto. *Reinforcement Learning: An Introduction*. The MIT Press, 1998.