World Academy of Science, Engineering and Technology
International Journal of Computer and Information Engineering
Vol:3, No:3, 2009

Optimizing Dialogue Strategy Learning Using Learning Automata

G. Kumaravelan and R. Sivakumar

Abstract—Modeling the behavior of the dialogue management in the design of a spoken dialogue system using statistical methodologies is currently a growing research area. This paper presents a work on developing an adaptive learning approach to optimize dialogue strategy. At the core of our system is a method formalizing dialogue management as a sequential decision making under uncertainty whose underlying probabilistic structure has a Markov Chain. Researchers have mostly focused on model-free algorithms for automating the design of dialogue management using machine learning techniques such as reinforcement learning. But in model-free algorithms there exist a dilemma in engaging the type of exploration versus exploitation. Hence we present a model-based online policy learning algorithm using interconnected learning automata for optimizing dialogue strategy. The proposed algorithm is capable of deriving an optimal policy that prescribes what action should be taken in various states of conversation so as to maximize the expected total reward to attain the goal and incorporates good exploration and exploitation in its updates to improve the naturalness of human-computer interaction. We test the proposed approach using the most sophisticated evaluation framework PARADISE for accessing to the railway information system.

Keywords—Dialogue management, Learning automata, Reinforcement learning, Spoken dialogue system

I. Introduction

THE Dialogue Management (DM) is the central component of a Spoken Dialogue System (SDS) accepting input from the user, producing messages as output to the user, communicating with external knowledge sources, and generally determining the dialogue flow in an efficient and natural way. In order to complete these tasks, the DM needs a dialogue strategy that defines when to take the initiative in a dialogue, when to confirm receipt of a piece of information, how to identify and recover from recognition and understanding errors, and so forth. Moreover, the dialogue management is the module that defines the interaction with the user and it is the window through which the user perceives the system's capabilities. This means dialogue strategies designed by human are prone to errors, labour-intensive and non portable. These facts motivate the topic of automatic dialogue strategy learning an attractive alternative.

Broadly speaking, three different approaches have been used in DM. First, the finite state-based approach represents the dialogue structure in the form of a network, where every node represents a question and the transitions between nodes represent all the possible dialogue [12] whose primary nature is system initiative. Secondly, the frame-based approach represents the dialogue structure in the form of frames that have to be filled by the user, where each frame contains slots that guide the user through the dialogue. In this approach the user is free to take the initiative in the dialogue [9]. Finally, agent-based approach incorporates a wide variety of approaches that use techniques from Artificial Intelligence to produce more intelligent systems [2] which incorporates mixed initiative interaction in which the user can change the current context and the dialogue history. These approaches typically involve authoring and complicating hand-crafted rules that require considerable deployment of time and cost. However they do not address the issue of how to develop the best possible dialogue strategy.

For these reasons, during the last decade many research groups have been attempting to find a way to automate the design of DM for learning dialogue strategy using machine learning technique such as Reinforcement Learning (RL) [26]. Reinforcement learning addresses the problem faced by an agent that learns behavior using trial and error interaction within a dynamic environment so as to maximize a scalar reward signal. The aforementioned factor influences dialogue management to be modeled as a Markov Decision Process (MDP) in which the state of the dialogue depends only on the previous state & its action and reinforcement learning is applied to find the optimal dialogue policy.

A number of reinforcement learning methods have been proposed in recent years to automate the design of dialogue strategy [7], [20], [21], [23], [25]. However, much research effort is being proposed for improving these techniques and in applying these techniques in various application domains. Generally speaking, RL has already been shown to be a powerful tool for solving single agent MDPs. On the other hand, increasing the size of the state space for RL has the danger of making the learning problem intractable (referred as "the curse of dimensionality"). Recent investigations employ function approximation [10], dialogue simulation [24] and prior knowledge [4] in order to find solutions on reduced state spaces. However, although these kinds of approach presents interesting characteristics in what concerns the learning of dialogue strategies, they suffer from well-known drawbacks in online operation. Especially in the number of interactions needed for convergence, which restricts their application mainly to offline processing. On the contrary, in online dynamic environments, the user interactions are relatively scarce and the SDS must be able to adapt its operation taking advantage of these limited interactions.

In addition, mostly all the popular RL algorithms (e.g.,

G. Kumaravelan is with Department of Computer Science, Bharathidasan University, Tiruchirappalli, 620 024, INDIA,
email: gkumaratcsbdu@gmail.com.
R. Sivakumar is with Department of Computer Science, AVVM Sri Pushpam College, Thanjavur, 613 503, INDIA,
email: rskumar.avvmspc@gmail.com.

Q-Learning) used in learning dialogue strategies are *model free* in nature and require explicit tabular storage of agent Q-functions and possibly of their policies [33]. Hence when the state and action space contain a large number of elements, tabular storage of the Q-function becomes impractical and there exists a dilemma with exploration versus exploitation in choosing an optimal action. Consequently, online model-based policy learning algorithms hold great promise in this regard to overcome the above mentioned factors in learning the dialogue strategy within RL framework [19].

Therefore, in these cases, approximate solutions must be sought out, for example, by extending to multiple agents that works on approximate single-agent RL [6]. Powerful approximate Multi-Agent Reinforcement Learning (MARL) algorithms have been proposed for discrete, large state-action spaces [1], continuous states & discrete state actions [3], [11], [35], and for continuous states and actions [8], [27]. Most of these algorithms work only in a narrow set of problems and are heuristic in nature. In a MARL however, the reinforcement an agent receives may depend on the action taken by the other agents acting in the same environment. Hence, the Markov property no longer holds and as such the convergence policies are lost. Alternatively, Learning Automata (LA) are valuable tools for current MARL research [5], [16], [29]–[31] and their learning scheme (purely *model-based*) updates strictly on the basis of the response of the environment and not on the basis of any knowledge regarding other automata, i.e. neither their strategies, nor their feedback. As such LA agents are simple. Moreover, LA can be treated analytically. Convergence proofs do exist for a variety of settings ranging from a single automaton model acting in a random environment to a distributed automata model interacting in a complex environment. Therefore, in this paper we propose a design for dialogue strategy using a team of learning automaton in the context of decentralized control of MDP. This method offers the following benefits among others: a) faster learning; b) broad convergence guarantees; c) 100% exploitation when all LA converge.

This paper is organized as follows: In the next section, a brief overview of problem area and methodology are given. In section III, the proposed algorithm is presented which could be useful for better convergence along with good exploitation and exploration in modeling dialogue management. The experimental result of the proposed online model-based learning algorithm on train information system is given in section IV along with evaluation consequence and comparison with the state-of-art approach for learning dialogue strategy. Finally the paper concludes in section V.

II. LEARNING AUTOMATA AND SOLUTION APPROACH

A. Learning automata

Adaptive learning is one of the main fields of Artificial Intelligence. LA is one of the most powerful tools in this research area [15], [17], [22], [28]. A learning automaton is a precursor of a policy iteration type of reinforcement learning algorithm and has some roots in psychology and operations research. LA are adaptive decision-making devices operating on unknown random environments. A learning automaton has a finite set of actions and each action has a certain probability (unknown to the automata) of getting rewarded by the environment of the automata as shown in Fig. 1. The objective is to learn to choose the optimal action (i.e. the action with the highest probability of being rewarded) through repeated interaction with the system. If the learning algorithm is chosen properly, then the iterative process of interacting on the environment can be made to result in the selection of the optimal action. We refer the reader to the survey papers [13], [14] for a review of various families and the corresponding properties of learning automata.

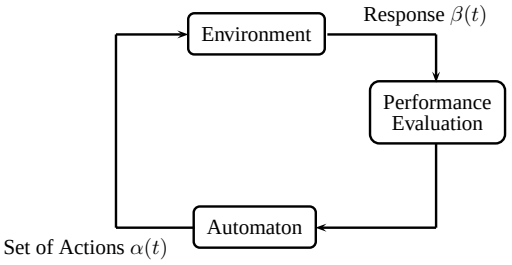

Fig. 1. A Learning Automata that interacts with a stochastic environment.

In a Variable-Structure Learning Automata (VSLA), the probabilities of the various actions are updated on the basis of information the environment provides. A VSLA is a quadruple $\langle \alpha, \beta, p, T(\alpha, \beta, p) \rangle$, where α, β and p constitute an action set with r actions, an environment response set, and the probability set p containing r probabilities, each being the probability of performing every action in the current internal automata state, respectively. The function of T is the reinforcement scheme which modifies the action probability vector p with respect to the performed action and received response. If the response of the environment takes binary values, learning automata model is P-model and if it takes finite output set with more than two elements that take values in the interval [0,1], such a model is referred to as Q-model, and when the output of the environment is a continuous variable in the interval [0,1], it is referred to as S-model. Assuming $\beta \in [0, 1]$, a general linear schema for updating action probabilities can be represented as follows:

$$P_i(t+1) = P_i(t) + \alpha\big(1-\beta(t)\big)\Big(1-\big(P_i(t)\big)\Big) - b\beta(t)P_i(t) \quad (1)$$

if action α was taken at time step t

$$P_j(t+1) = P_j(t) - \alpha\beta(t)P_j(t) + b\big(1-\beta(t)\big)\lceil r-1 \rceil^{-1} - P_j(t) \quad (2)$$

$$\forall j \neq i.$$

The constants a and b in the interval [0,1] are the reward and penalty parameters respectively and r the number of actions of the action set of the automata. When $a = b$ the algorithm is referred to as linear reward-penalty ($L_{R\text{-}P}$), when $b = 0$ it is referred to as linear reward-inaction ($L_{R\text{-}I}$) and when b is small compared to a it is called linear reward-ϵ-penalty ($L_{R\text{-}\epsilon P}$).

B. Interconnected Learning Automata: Decentralized Control of MDPs

The important problem of controlling a Markov chain can be formulated as a network of automata in which control passes from one automata to another. In this set-up, every action state in the Markov chain has an LA that tries to learn the optimal action probabilities in that state with learning update rules (1) and (2). Only one LA is active at each time and the transition to the next state triggers the LA from that state to become active and take some action. The LA active in state i is not informed of the one-step reward $r_j^i(k)$ resulting from its action k, leading to state j. Wheeler and Narendra [34] have proved that this interconnected LA-model is capable of solving the MDP.

C. Casting LA model to learn dialogue strategy

The current state-of-art SDS often has mixed-initiative slot-filling system. This means that both the user and the system take the initiative to provide information or ask to follow up questions in a dialogue session to jointly complete certain tasks. However, automatically designing an efficient dialogue strategy to assist the user to quickly fill in the slots has never been a trivial problem.

The LA model presents DM strategy that has to be optimized and the DM will be the learning agent. At each turn the learning automata has to choose an action according to its interaction strategy with the environment so as to complete the task that it has been designed for. These actions can be greetings, request to constraint and confirm the value of attributes, perform retrieval operations in the database and to close the dialogue session. The response from the environment leads to the updating of the reward and internal state of the learning agent which contains enough information about the history of the dialogue.

At each decision epoch, an action a_i is chosen and the corresponding reward is obtained. The goodness measure GM(a_i) is then updated using the following exponential weighted average:

$$\begin{aligned} &\textit{NewGM}(a_i) \leftarrow \textit{OldGM}(a_i) \\ &\quad + \textit{StepSize}[\textit{CurrentReward}(a_i) - \textit{OldGM}(a_i)] \end{aligned} \tag{3}$$

This formula states that, when each action a_i is performed, its NewGM(a_i) is updated using the subjective difference of CurrentReward(a_i) and OldGM(a_i) to the OldGM(a_i). Here StepSize is a learning parameter, which has been experimentally verified as a good parameter value in nonstationary environments.

The new action is chosen according to the following probabilistic rule with updated Goodness measure NewGM(a_i) as in (3).

$$P[\textit{newaction} = a_i] \frac{e^{\frac{1}{\textit{NewGM}(a_i)}}}{\sum_{a_k \in AS} \left[e^{\frac{1}{\textit{NewGM}(a_k)}} \right]} \tag{4}$$

AS denotes action set $[a_1, a_2, \ldots, a_n]$ for each LA in the Markov chain and the denominator is a normalization term. Since $e^{1/\textit{NewGM}(a_i)}$ in (4) increases as NewGM(a_i) decreases, the action that has the smallest NewGM(a_i) is likely to be chosen with the highest probability.

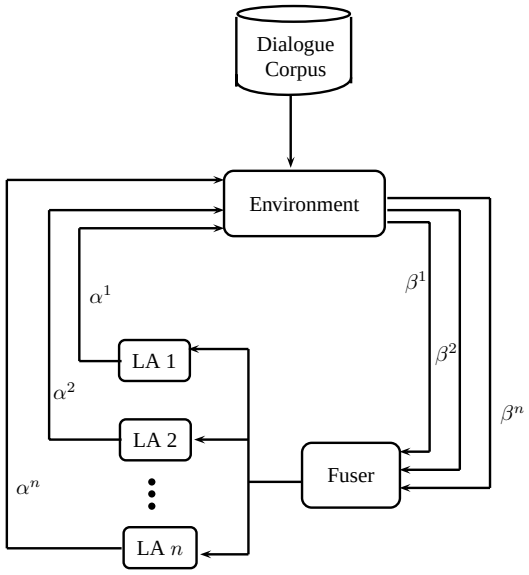

Fig. 2. Mechanism of learning dialogue strategy using interconnected Learning Automata.

III. PROPOSED APPROACH FOR LEARNING DIALOGUE STRATEGY

A central debate in the literature concerns the use of models for learning dialogue strategy. Model-free approaches do not explicitly represent the dynamics of the environment, but instead directly approximate a value function that measures the desirability of each environment state. These approaches offer near-optimal solutions that depend on systematic exploration of all actions in all states [18]. On the other hand, model-based approaches explicitly represent dynamics of the environment to compute an estimate of the expected value of each action. With a model, the agent can reduce the number of steps to learn a policy by simulating the effects of its actions at various states. Perhaps for this reason and for the fact that it is possible to derive a policy that is guaranteed to be optimal with respect to the data. Therefore LA (model-based approach) is guaranteed to find an optimal policy that maximizes the expected total reward with good convergence property.

A. Convergence Property

The ultimate product of utilizing reinforcement learning methods for dialogue management is a policy that is optimal with respect to the data. Fig. 2 shows the interconnected LA model which optimizes the learning strategy through interaction with environment. We put an automata LA_k^i in each action state of the Markov chain s_i with $i : 1 \cdots N$, for each agent $k, k : 1 \cdots N$. Each LA_k^i through its interaction with its environment updates the probabilities until the optimal action(s) has the highest probability. Over time, with trial and error, the system learns the optimal action in each stage of the conversation.

Let A_j denote the set of actions available in LA_k^i. Hence the union of A_j over LA_k^i gives the action space. With every agent-action pair, we associate a probability $P(LA_k^i, a)$ of taking action a in LA_k^i. Obviously $\sum_{a \in A_j} P(LA_k^i, a) = 1$. In the beginning of the learning process, the policy is random and each action is likely to be equal. Hence $P(LA_k^i, a) = 1/r_j$ where $r_i = |A_j|$ is the number of possible actions in LA_k^i. If the performance of an action is good the probability of that action is increased and if the performance is poor, the probability is reduced. However the updating scheme must always ensure that the sum of probabilities of all the actions in a given LA_k^i is 1.

B. Feedback Mechanism

The action chosen by the automata is the input to the environment which responds with stochastic response or reinforcement. The automata LA_k^i active in state s_i is not informed of the one-step reward R_j^i resulting from choosing joint action $a^i = \{a_1^i, \ldots, a_n^i\}$ with $a_k^i \in LA_k^i$ in s_i and leading to state s_j. Instead, when state s_i is visited again, all automata LA_k^i receive two pieces of data: the cumulative reward and the current global time. From these, all LA_k^i compute the incremental reward generated since this last visit and the corresponding elapsed global time. The environment feedback or the input to LA_k^i is taken to be:

$$\beta_i(t_i + 1) = \frac{\rho_i(t_i + 1)}{\eta_i(t_i + 1)} \tag{5}$$

where $\rho_i(t_i+1)$ is cumulative total reward generated for action a_i in state s_j and $\eta_i(t_i + 1)$ the cumulative total time elapsed.

C. Learning Scheme

The learning scheme for updating the probabilities using feedback is a topic of research and several schemes have been suggested in literature. The scheme that gave the best result is known as the Reward-Inaction Scheme [15]. The goal is to identify the optimal action to be achieved through a learning algorithm that updates the action probability at each instant, using the most recent interaction with the environment. For this $\beta_i(t_i+1)$ is calculated and then according to the Reward-inaction scheme, the probabilities are updated via the rule given below:

$$P(LA_k^i, a_i) \leftarrow P(LA_k^i, a_i) + \eta\beta(i)I[D(i) + a_i] - \eta\beta(i)P(LA_k^i, a_i) \tag{6}$$

where $D(i)$ denotes the action taken in LA_k^i in its last visit, η denotes the learning rate, a^i the action whose probability of getting selected in LA_k^i is to be updated and $I[.]$ equals 1 if the condition inside the brackets is satisfied and $I[.] = 0$ otherwise. In this scheme a good action automatically has a high value for β and therefore the scheme increases the probability of that action. Similarly an action that results in a poor reward has a low value for β and therefore the probabilities are not changed significantly. Thus, the objective of the learning scheme is to maximize the expected value of the reinforcement received from the environment. Hence, an equivalent way to characterizing the goal of an automata can be defined as:

$$\max M^k(i) = E[\beta^k(i) \mid P^k] \tag{7}$$

where $\beta^k(i)$ be the feedback received by the automata in the i-th state at the k-th iteration of the algorithm and $M^k(i)$ to be its expected value. The feedback is associated with given values for the vector $P^k = (P^k(i,1)P^k(i,2), \ldots, P^k(i,r_i)$, where $P^k(i,a)$ denote the probability of selecting action a in LA_k^i in the k-th iteration of the algorithm.

Now the algorithm for optimizing dialogue strategy using LA is given as follows:

```
1. Initialize
for all states s, agents k do
  p(s,k) = randomInit()//state action probabilities
  lastTime(s) = 0 // last time step state was visited
  lastReward(s,k) = 0// agent reward when state
                     //was last visited
  totalReward(k) = 0// total reward received by
                     //each agent
  lastAction(s,k) = 0//last action played in each state
end for
state = startState;
2.
for 1 < iteration < maxIter do
  jointAction = ∅
  for all agents do
    if state visited before then
      //calculate feedback and update probabilities
      deltaRew=totalReward (agent)−
                          lastReward(state,agent)
      deltaTime=iteration−lastTime(currentState);
      feedback=deltaRew / deltaTime;
      updateProbabilities(p (state,agent),
            lastAction(state,agent), feedback)
    end if
    //select new action
    action = selectAction(p (state,agent))
    jointAction = jointAction ∪ action
    //store current reward & action taken
    lastReward(state, agent)=totalReward(agent)
    lastAction(state,agent)= action
  end for
  3. Store data as
      lastTime(state)=iteration
      oldState=state
  4. Go to next state by
      state=getNextState(state, jointAction)
  5. Receive immediate reward as
  for all agents do
    totalReward(agent)=totalReward(agent) +
          getReward(oldState, jointAction)
  end for
end for
```

TABLE I
AUTOMATON AND ENVIRONMENT DIALOGUE ACTS

System Acts	User Acts
Greeting	Command (bye)
Request_Info(dep_value)	Provide_ Info(dep_value)
Request_Info(dest_value)	Provide_ Info(dest_value)
Request_Info(date_value)	Provide_ Info(date_value)
Request_Info(time_value)	Provide_ Info(timevalue)
Database Results	

IV. EXPERIMENTAL RESULTS

A. Experimental Design

In this section, we present a slot-filling dialogue system based on train booking domain to verify the effectiveness of dialogue strategy for the proposed model. The goal of the system is to acquire the values for four slots (attributes): departure city, destination city, the date the user wishes to travel and the time of travel. For each slot, we put a learning automata in each state of the Markov chain. Each state representation contains enough information about the history of the dialogue. In slot-filling dialogues, optimal strategy is the one that interacts with user in a satisfactory way through interconnected learning automata in the Markov chain while trying to minimize the length of the dialogue. In our experiments we use the very elemental actions like: i) Greeting ii) Ask to constrain the values of the attribute iii) Ask open ended question iv) Ask to confirm the values of the attribute v) Ask to relax the values of attribute vi) Perform a database query to retrieve information vii) Close the dialogue session. The values of the attributes are preserved in the Markov chain for the purpose of database query. Thus the total state space is restricted to six including the initial state (Greeting) and sink state (Database query) of the controlled finite Markov chain along with the aforementioned 4 attributes with respect to our experimental settings of the dialogue system. However the total possible actions will be 66 on our simulations. The aim is to explore and exploit the learning capability of the automata in an uncertain environment with prior knowledge on using the system and user Dialogue Acts (DAs).

The chosen system and user dialogue acts are summarized in Table I. The system dialogue acts allow the system to request the user for the slot values or to confirm these values, either explicitly or implicitly and to restart or end the dialogue. Finally, the system presents the results of a users database query. The user dialogue acts allow the user to provide slot information and to terminate the dialogue.

B. Results

After several thousand simulated dialogue sessions, the system adopts a learning strategy as shown in Fig. 3. In this simulation, we focused on the convergence of LA algorithm. Since the recognition rate of the speech recognizer was assumed to be 100%, the behavior of the Reward-Inaction learning scheme for the proposed approach with different values of η (learning rate), i.e. $\eta = 0.1$, $\eta = 0.5$ and $\eta = 0.8$ is plotted in Fig. 4 shows that bigger the value of η, the faster convergence. However, the bigger we set η the higher

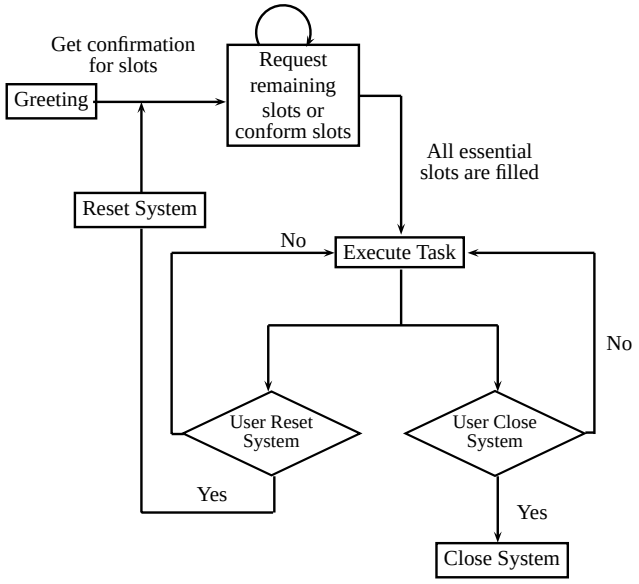

Fig. 3. The dialogue strategy learned by LA learning algorithm

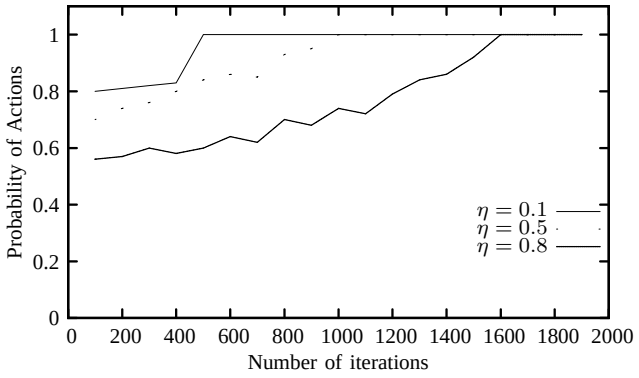

Fig. 4. Convergence of the learning scheme with learning parameter η.

the probability of convergence to a suboptimal action, this is due to the fact that an L_{RI} scheme is only guaranteed to be ϵ-optimal. This is illustrated in Table II, shows the percentage of runs over and its action converges to zero as a function of η. From Table II it is clear that if one were to tolerate a 5% error, a suitable value for the learning rate would be 0.3, which would substantially increase the speed of convergence.

The first few states of the trajectory generated by the simulation along with its updated calculations performed in each states of Markov chain are listed in Table III. The updation process is continued in this fashion for a large number of iterations. The most likely action (the action with the largest probability) in each state is considered to be the action for that state. Table III also shows the cumulative reward obtained for each iteration based on the selected action in each states of Markov chain. Each action is selected based on Reward-inaction scheme and the current reward is assigned with positive or negative value in the range $(+10.0$ to $-5.0)$ for the best & worst action selection respectively.

TABLE II
EFFICIENCY OF LEARNING PARAMETER

η	Percentage of wrong Convergence
0.9	22
0.5	12
0.3	5
0.1	0
0.01	0
0.001	0

TABLE III
ITERATIVE UPDATES DURING THE SIMULATION OF THE SYSTEM

Iteration	{Current state (i) Next state (j)}	Selected Action	Current Reward	Cumulative Reward
1	$\{s_2, s_1\}$	3	0	4.5
2	$\{s_1, s_3\}$	4	4.5	8.0
3	$\{s_3, s_1\}$	4	8.0	7
4	$\{s_1, s_2\}$	3	7.0	9.4

C. Evaluation

In order to find the most significant parameters (i.e., predictors) of the proposed dialogue management performance, the PARADISE framework [32] was used. It maintains that the systems primary objective is to maximize user satisfaction, and it derives a combined performance metric for a dialogue system as a weighted linear combination of task-success measures and dialogue costs. To evaluate the dialogue strategies 60 undergraduate students (32 female, 28 male) with an average age of 21 have tested our system. It is important to mention that since our testers had no previous experience with a dialogue system, our experiments were performed with novice users.

To evaluate user satisfaction, users were given the user-satisfaction survey Table IV used within PARADISE framework, which asks to specify the degree to which one agrees with several questions about the behavior or the performance of the system (TTS Performance, ASR Performance, Task Ease, Interaction Pace, User Expertise, System Response, Expected Behavior, and Future Use). The answers to the questions were based on a five-class ranking scale from 1, indicating strong disagreement, to 5, indicating strong agreement. For our experiment, the mean User Satisfaction value was 33.32 as shown in Table V. In this table we can observe some relevant positive results for the dialogue strategy: Task ease-of-use, user expertise, expected behavior and future use.

For computing the system performance, PARADISE model applies Multiple Linear Regression (MLR) with User Satisfaction as the dependent variable and task-success measures and dialogue costs as the independent variables as:

$$\textit{performance} = \big(\alpha \times N(\kappa)\big) - \sum_{i=1} w_i \times N(c_i) \qquad (8)$$

Here, α is the weight on the Kappa coefficient κ, w_i are weights on the dialogue costs c_i and N is a Z-score normalization function defined as:

$$N(x) = (x - \bar{x})/\sigma_x \qquad (9)$$

TABLE IV
USER-SATISFACTION SURVEY USED WITHIN THE PARADISE FRAMEWORK

TTS quality:	Was the system easy to understand?
ASR quality:	In this conversation, did the system understand what you said?
Task Ease-of-Use:	In this conversation, was it easy to find the message you were looking for?
Interaction Pace:	Was the pace of interaction with the system appropriate in this conversation?
User experience:	Did you know what you could say at each point of the dialogue?
System response:	How often was the system sluggish and slow to reply to you in this conversation?
Expected behavior:	Did the system work the way you expected?
Future use:	Based on your current experience with the system, would you use it regularly when you need information about the train details?

TABLE V
USER SATISFACTION SURVEY RESULTS

Criteria	Scale Values
TTS Quality	4.08
ASR Quality	4.20
Task ease	4.40
Interaction pace	2.68
User expertise	4.20
System response	4.16
Expected behavior	5.0
Future use	4.60
User Satisfaction	**33.32**

where $\bar{x}$ and σ are the mean value and the standard deviation for x, respectively, computed from the sample set of observations. The normalization function N guarantees that the weights directly indicate the relative contributions to the performance function, which can be used to predict User satisfaction. A summary of metrics collected from our systems applying the PARADISE framework is listed in Table VI. This table shows that our approach is more efficient and in fact better in System Turns, User Turns and Elapsed Time.

D. Comparison

In optimizing dialogue strategy using traditional reinforcement learning algorithms such as Q-learning (purely model free) one agent visits all states and keeps a single Q-table for all the states in the system. However in the proposed approach there are non-mobile LA agents which do not move around the

TABLE VI
METRICS PERFORMANCE VALUE FOR THE DIALOGUE STRATEGY

Metrics	Parameter Values
Efficiency Measures	
System Turns	38.96
User Turns	17.56
Elapsed Time (Secs)	346.24
Qualitative Measures	
MRS	0.82
Time Out	1.24
Retry	4.92
Help	0.08
Cancel	0.12
Task Success (κ)	**0.79**
User Satisfaction	**33.32**

state space but stay in their own state to get active and learn to take actions only in their own state of the Markov chain. In Q-leaning approach with multi agent setup, each agent uses an epsilon greedy exploration strategy with a standard single agent Q-leaning rule as:

$$Q_i(s,a) = (1-\alpha)Q_{t-1}(s,a) + \alpha(r = \gamma \max a, Q_i(s',a')) \quad (10)$$

Each Q-learner learns individually the associated Q-values with their own action, rather than joint actions. The Q-learners are completely independent and have no knowledge of the other agents acting in the environment and influencing their reward. Despite the above fact the Q-learning algorithm is known to have problems in such multi-agent setting. Thus the Learning automaton with sufficiently low learning rate is converged into the optimal policy each time. However the Q-learners on the other hand find close but non-optimal policies. Their reward is also lower as they keep a fixed rate of exploration throughout the conversation. Thus when the system has a unique limiting distribution over the state space, both independent Learning automaton based agents and independent Q-learners are able to find a strategy for optimal action. Although the latter needs good exploration settings, it may take a very long time before convergence.

Dialogue designers considering the use of LA methods to learn policies for action selection in dialogue management enjoy several advantages from other meta-heuristics. First, it does not require a starting solution; also it has a natural stopping criterion. Second, since it is much simpler to learn from data and it does no require estimation of the optimal value function. And finally the LA model makes fewer assumptions about the structure of the state space, which is quite appealing from a theoretical standpoint.

V. CONCLUSION

This paper presents a method for the development of model-based learning automata algorithm for dialogue management that can learn from training samples to generate the system actions. This representation allows the system to automatically generate specialized actions that takes into account the current situation of the dialogue depending on the use of expected cumulative reward. This approach is appealing due to the following benefits: a) faster learning, because the state space is being divided among multiple interconnected automata that can work independently to provide better convergence b) reduced computational demands, because breaking the problem into sub problems helps to ignore irrelevant features and adapt to good exploitation strategy c) Knowledge transfer, because solution learnt on previous problems can be re-used in new problems. Some experiments have been performed to test the behavior of the system with respect to PARADISE framework. The results show the satisfactory operations of the developed approach. An important area for future research could be in the area of reinforcement schemes, to analyze the finest configuration parameters for the environment. We hope that such a formulation would lead to a finer learning curve that would mimic more closely the behavior of the learning algorithm. Our tests are geared towards stochastic domain of interest; this may form an interesting avenue for further research.

REFERENCES

[1] O. Abul, F. Polat, and R. Alhajj, "Multiagent reinforcement learning using function approximation," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 4, no. 4, pp. 485–497, Nov 2000.
[2] J. Allen, D. Byron, M. Dzikovska, G. Ferguson, L. Galescu, and A. Stent, "Towards conversational humancomputer interaction," *AI Magazine*, vol. 22, no. 4, pp. 27–38, 2001.
[3] L. Busoniu, B. D. Schutter, and R. Babuska, "Decentralized reinforcementlearning control of a robotic manipulator," in *Proc. 9th Int. Conf. Control Autom. Robot. Vis. (ICARCV-06), Singapore*, 2006, pp. 1347–1352.
[4] H. Cuayahuitl, S. Renals, O. Lemon, and H. Shimodaira, "Reinforcement learning of dialogue strategies using hierarchical abstract machines," in *Proc. of IEEE/ACL SLT*, 2006.
[5] L. V. de Wege, "Learning automata as a framework for multi-agent reinforcement learning," Master's thesis, Vrije Universiteit Brussel, Belgium, 2006.
[6] S. Dzeroski, L. D. Raedt, and K. Driessens, "Relational reinforcement learning," *Machine Learning*, vol. 43, no. 1–2, pp. 7–52, 2001.
[7] R. P. E. Levin and W. Eckert, "A stochastic model of humanmachine interaction for learning dialog strategies," *IEEE Trans. Speech Audio Processing*, vol. 8, no. 1, pp. 11–23, 2000.
[8] F. Fernandez and L. E. Parker, "Learning in large cooperative multi-robot systems," *International Journal of Autonomous Robots*, vol. 16, no. 4, pp. 217–226, 2001.
[9] D. Goddeau, H. Meng, J. Polifroni, S. Seneff, and I. S. Busayapongcha, "A form-based dialogue manager for spoken language applications," in *Proc. of ICSLP, Philadelphia, USA*, 1996, pp. 701–704.
[10] J. Henderson, O. Lemon, and K. Georgila, "Hybrid reinforcement/supervised learning for dialogue policies from communicator data," in *Workshop on Knowledge and Reasoning in Practical Dialogue Systems (IJCAI)*, 2005.
[11] Y. Ishiwaka, T. Sato, and Y. Kakazu, "An approach to the pursuit problem on a heterogeneous multiagent system using reinforcement learning," *Robot. Autonomous System*, vol. 43, no. 4, pp. 245–256, 2003.
[12] M. Mctear, "Modelling spoken dialogues with state transition diagrams: Experiences with the cslu toolkit," in *Proc. of ICSLP, Sidney, Australia*, 1998, pp. 1223–1226.
[13] K. S. Narendra and S. Lakshmivarrahan, "Learning automata—a critique," *Journal of Cybernetics and Information Sciences*, pp. 53–66, 1977.
[14] K. S. Narendra and M. A. L. Thathachar, "Learning automata-a survey," *IEEE Transaction on Systems, Man and Cybernetics-SMC*, vol. 4, no. 8, pp. 323–334, 1974.
[15] ——, *Learning Automata: An Introduction*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
[16] A. Nowé, K. Verbeeck, and M. Peeters, "Learning automata as a basis for multi agent reinforcement learning," *Learning and Adaption in Multi-Agent*, pp. 71–85, 2006, ISSN 0302-9743.
[17] B. J. Oommen and M. Agache, "Continuous and discretized pursuit learning schemes: Various algorithms and their comparison," *IEEE Transactions on Systems, Man and Cybernetics, Part B:*, vol. 32, pp. 77–287, 2002.
[18] T. Peak and D. Chickering, "The markov assumption in spoken dialogue management," in *6th SIGDial Workshop on Discourse and Dialogue*, 2006.
[19] T. Peak and R. Pieraccini, "Automating spoken dialogue management design using machine learning: an industry perspective," *Speech Communication*, vol. 50, pp. 716–729, 2008.
[20] O. Pietquin, *A Framework for Unsupervised Learning of Dialogue Strategies*. Preses Universitaries de Louvain, 2004.
[21] O. Pietquin and T. Dutoit, "A probabilistic framework for dialog simulation and optimal strategy learning," *IEEE Transactions on Audio, Speech and Language Processing*, vol. 14, no. 2, pp. 589–599, 2006.
[22] A. S. Poznyak and K. Najim, *Learning Automata and Stochastic Optimization*. Springer, 1997.
[23] M. K. S. Singh, D. Litman and M. Walker, "Optimizing dialogue management with reinforcement learning: experiments with the njfun system," *Journal of Artificial Intelligence Research*, vol. 16, pp. 105–133, 2002.

[24] J. Schatzmann, K. Weilhammer, M. N. Stuttle, and S. Young, *A Survey of Statistical User Simulation Techniques for Reinforcement-Learning of Dialogue Management Strategies*. Cambridge University Press, 2006, vol. 21, pp. 97–126.
[25] K. Scheffler and S. Young, "Corpus-based dialogue simulation for automatic strategy learning and evaluation," in *Proc. of the NAACL Workshop on Adaptation in Dialogue Systems*, 2001.
[26] R. Sutton and A. Barto, *Reinforcement learning: An introduction*. MIT Press, 1998.
[27] H. Tamakoshi and S. Ishii, "Multiagent reinforcement learning applied to a chase problem in a continuous world," *Artif. Life Robot*, vol. 5, no. 4, pp. 202–206, 2001.
[28] M. A. L. Thathachar and P. S. Sastry, *Networks of Learning Automata: Techniques for Online Stochastic Optimization*. Norwell, MA: Kluwer, 2004.
[29] K. Tuyls and A. Nowé, "Evolutionary game theory and multi-agent reinforcement learning," *The Knowledge Engineering Review*, vol. 20, pp. 63–90, 2005, ISSN 0269-8889.
[30] K. Verbeeck, A. Nowé, P. Vrancx, and P. Maarten, *Multi-Automat Learning, Reinforcement Learning: Theory and Applications*. I-Tech Education and Publishing, 2008.
[31] K. Verbeeck, P. Vrancx, and A. Nowé, "Networks of learning automata and limiting games," in *Proc. of the 7th ALAMAS Symposium*, 2007, pp. 171–182, ISSN 0922-8721.
[32] M. Walker, D. Litman, C. Kamm, and A. Abella, "Paradise: A framework for evaluating spoken dialogue agents," in *Proc. of the 5th annual meeting of the association for computational linguistics(ACL-97)*, 1997, pp. 271–280.
[33] C. J. C. H. Watkins and P. Dayan, "*Q*-learning," *Machine learning*, vol. 8, no. 3, pp. 229–256, 1992.
[34] R. M. Wheeler and K. S. Narendra, "Decentralized learning in finite markov chains," *IEEE Transactions on Automatic Control*, vol. AC-31, pp. 519–526, 1986.
[35] M. Wiering, R. Salustowicz, and J. Schmidhuber, "Reinforcement learning soccer teams with incomplete world models," *Autonomous. Robots*, vol. 7, no. 1, pp. 77–88, 1999.