Experience Based Learning in Policy Control of Multiagent System

Ariuna Damba and Shigeyoshi Watanabe
Graduate School of Electro-Communications
University of Electro-Communications
1-5-1 Chofugaoka, Chofu-shi, Tokyo 182-8585
ariuna@ice.uec.ac.jp, watanabe@ice.uec.ac.jp

***Abstract*—In this paper a method of simulated learning of policy control is proposed for dynamic multiagent system, where agent's decision mechanism is represented as a function of agent's past experience. The system of homogeneous agents with different sensor input and effector output is considered.**

Two extensions to existed works of multiagent system simulation approaches are proposed which enable to capture dynamic behavior of agents. First, an action choosing policy depends on the agent history of chronological order of agent observations and actions. Agent learns not actions directly, but the performance of policy function of what has been experienced in the past. This history dependence in one hand expands the strategy space, but on other hand it makes possible to describe agent policies dynamic over interaction involving wider spectrum of behavior. Second, policy is a function of not just a single agent action but a function of joint action formated by a simulated learning process of multiagent adaptation to dynamic environment.

A simulation of independent episodes and learning from episode output as a powerful investigation tool is used for experimental study.

***Index Terms*—Optimal Control, Multiagent System, Reinforcement Learning, Monte Carlo Approach, Multi-vehicle Simulation**

I. Introduction

Multiagent modeling and solution mechanisms have been successfully applied to social science, economics, organizational management and other scientific work where intelligence is expressed in obtaining good performance using achievable resources and adapting the current situation for future benefit. Researchers have been aiming to provide a bridge between artificial intelligence (AI) research on real-time planning and learning with relevant concepts and algorithms from control theory [1], [2].

Multiagent based approach finds success in many research methodologies due to it feature of segregating the problem into easy solvable subtasks and assigning them to different agents. For complex systems such as air traffic control, the multiagent architecture with appropriate methods of agents coordination is used to solve the decentralized control problem [3], [4], [5]. In order for the agents to be able to take their actions in a distributed fashion, appropriate decision making mechanisms must be additionally developed.

In terms of modeling other agents, there is much room for improvement in the situation when a given agent does not know the internal state or sensory inputs of another agent. If the information is not directly available, it is essential for an agent to learn it [6], [7]. Interacting within unknown and uncertain environment agent learns it dynamic to use this knowledge in own decision making. This mechanism is studied in Reinforcement Learning (RL) research.

Agents learn from simulated experience which are generated by Monte Carlo policy control method from simulated interactions [8], [9]. While acting and receiving some feedback each agent gets experienced adapting and learning to behave [10]. Some advantages of using RL for control problems are that an agent can be trained easily to adapt to environment changes, and trained continuously while the system is online, thus improving performance all the time. On the other hand, the coordination between agents with independent goals and independent strategies is usually established to share information. But in some real world systems establishing communication for such coordination could be partially effective or even impossible and then the behavior of the decision maker is based upon observations of environment during the process.

To deal with these problems we model multiagent adaptation to dynamic environment through a simulation method of multiagent decision making based on computed policy function with no direct communication among agents [11], [12]. Two extensions to existed works of multiagent system simulation approaches are proposed which enable to capture dynamic behavior of agents. First, an action choosing policy depends on the agent history represented by chronological order of agent observation and action as past experience. This history dependence in one hand expands the strategy space, but on other hand it makes possible to describe agent policies dynamic over interaction involving wider spectrum of behavior. Namely, two methods of policy derivation process are considered: policy value update in a risk aware manner and policy update of gaining the best manner. Second, modeling of multiagent adaptation to dynamic environment through a simulated learning of the policy is represented as a function of not just a single agent action but a function of joint action formated by a learning process. Here, simulation of independent episodes and learning from episode output provides a powerful predictive tool for systems with no direct communication among the decision makers.

An internal model of an agent is described in Section 2. Theoretical assumptions of decision making in a multiagent system underlying this paper follow in Section 3. Section 4 de-

978–1–4244–1674–5/08/$25.00 © 2008 IEEE

scribes the proposed algorithm of policy control in a multiagent system. Section 5 involves an experimental part of simulation of multi-vehicle control example. And the concluding remarks are given in the last Section 6.

II. Agent Model.

A rational agent in stage k performs an action $a_k \in \alpha$ (action set), which results in an environment observation (or perception) μ_k, followed by next stage $k+1$. On every stage agent obtains new information μ from the environment and this information is updated to an agent internal state configuring an agent history.

Definition 2.1: Given that the action space α is finite, an agent history h is an internal state of an agent consisting of a chronological sequential order of action/observation records actually updated by an agent on each stage k.

$$h_k = h_0 a_1 \mu_1 a_2 \mu_2 \dots a_{k-1} \mu_{k-1} a_k \mu_k = h_{k-1} a_k \mu_k. \quad (1)$$

An agent performs an action a_k, observes an environment reaction μ_k which is determined by the environment probability distribution ρ, a conditional probability function that the environment responses with $\mu_1 \dots \mu_k, \mu k+1$ under the condition that the agent actions are $a_1 \dots a_k, a_{k+1}$.

$$\rho(\mu_{k+1}) = \rho(\mu_0)\rho(\mu_1|a_1,\mu_0)\dots\rho(\mu_k|a_k,\mu_{k-1})a_{k+1}. \quad (2)$$

Definition 2.2: In stage k the action a_k of the agent is defined by a decision rule called a policy π that is a function of agent history on this stage: $a_k \leftarrow \pi(h_{k-1})$.
By the expressions (1) and (2) and Definition 2.2

$$\begin{aligned} \rho^{\pi}(\mu_{k+1}) &= \rho^{\pi}(\mu_0\mu_1\dots\mu_k|a_1a_2\dots a_k)a_{k+1} \\ &= h_k\pi(h_k). \end{aligned} \quad (3)$$

The above equation implies that the prior probability of environment observation is defined by past history and following policy function.

An agent with history h_k to follow any policy must know some evaluative criteria of the action to be chosen. A rational agent tries to maximize the expected utility function V which is the sum of future total rewards r, the numerical values representing the agent measure of the environment reactions to its actions. Let the number of timing stages be finite value T. As the environment has a probabilistic feature the expected accumulated reward of performing action a_{k+1} and following the policy π thereafter is

$$V^{\pi}(h_k, a_{k+1}) = \sum_{k}^{T} r(\mu_{k+1})\rho^{\pi}(\mu_{k+1}). \quad (4)$$

Utility function V is used to pre-analyze all actions or behaviors of the agent and is called policy value in the environment with distribution ρ given history h. The maximum expected reward is obtained by accumulating over observations μ and maximizing over actions a. During the learning an agent maximizes the future rewards recursively, i.e. agent updates the policy value by sum of immediate reward and a maximum V value of the action to be taken next. The expression (4) can be written for stage k as

$$V^{\pi}(h_{k-1}, a_k) = r(\mu_k) + \max_{a\in\alpha} V^{\pi}(h_k, a_{k+1}). \quad (5)$$

So an agent chooses the action with maximum value over the set of possible actions α,

$$a_{k+1} = arg\max_{a\in\alpha} V^{\pi}(h_{k-1}, a_k). \quad (6)$$

The policy function is updated according to this choice, $\pi(h_k) = \pi(h_k, a_{k+1}, \mu_{k+1})$.

As shown above, the policy function of agent is emanated from history of actually taken actions/observations and the current environment observation. Thus, an agent policy derivation is a process of mapping the past experience and current observation into an action choice.

III. Decision Policy in Multiagent System

A. Multiagent System Design

Multiagent system differs from single-agent system in that several autonomous agents with particular properties exist in an environment the dynamics of which caused by agents behavior. Although this interaction could be viewed by an agent as environmental influence not being separated from the environment dynamics [7].

Work in [13] proposes a form of multiagent reinforcement learning (RL) in which agents do not model each other as agents . Instead agents consider each other as parts of the environment and affect each other's policies only as sensed objects. This research shows that the agents can learn to cooperate without modeling each other.

A Recursive Modeling Method is applied to model the internal state of another agent in order to predict its actions in [14]. Even though the agents know each others' goals and structure, they may not know each other's future actions. The missing pieces of information are the internal states and sensory inputs of the other agents.

Research of [6] describes the Multiagent Markov Decision Process (MMDP) as a general model in n-person cooperative game in which agents share the same utility function . The collection of agents is considered as a single agent with joint actions and optimal joint policies which was determined as a problem of equilibrium selection.

An application in [15] presents a multiagent cooperation where agents must communicate with each other to share the information needed for deciding which actions to take . Here, each agent has its own Markov process involved in global process and global states are known to agents at all time.

The method described in this paper is proposed for dealing with the lack of information between the agents in multiagent uncertain environment. There is no direct communication among the agents and each agent observes the environment from its own local view involving the other acting agents into the environment dynamics. Agent simulates the environment with a large number of different episodes, derives its own decision policy and learns to adapt to environment dynamics.

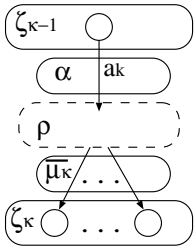

Fig. 1. An Agent State Transition in Multiagent System.

if the number of episodes is large enough then it could be assumed that the agent learns the world behavior involving the other agents' influence and thus would be able to predict and build its own decision policy.

Definition 3.1: The multiagent system is described by the following parameters:

1) The set of N agents $\{agent^1,\ldots,agent^N\}$, the set of possible actions of agent i, $\alpha^i=\{a_1^i,a_2^i,\ldots\}$.
2) The set of local observations of environment state by N agents at stage k, $\{\mu_k^1,\ldots,\mu_k^N\}$.
3) The available information set at stage k, $\{\varsigma_k^1,\ldots,\varsigma_k^N\}$ of each agent i, $\varsigma_k^i=\varsigma_0^i a_1^i\mu_1^i\ldots a_k^i\mu_k^i$.

B. Optimal Policy Considerations

Since the environment in a multiagent system is supposed to be not static but dynamic, its reaction to an agent action is not unique. Figure (1) reflects this point in an agent transition from state with information set ς_{k-1} to state ς_k, where an action a_k from action set α leads to multiple observations $\bar{\mu}_k$ of environment with probability distribution ρ. Consequently, the information set ς_k is not unique.

The function ρ of expression (2) should be edited for multiagent case as $\rho(\varsigma_k)=\rho(\bar{\mu}_k|a_k,\varsigma_{k-1})$.

An agent perceives the other agents as an environment dynamics ρ. Thus an agent observation $\bar{\mu}$, consequently, the information set on a given stage are affected by other agents' policy function. An agent which derives its own policy should estimate the other agents' policies, so that a joint policy of N agents, $\Pi=(\pi^1,\ldots,\pi^N)$, defines the system dynamics. Here, the policy of an agent is optimal in sense of adaptation to other agents' behavior, i.e. in view of optimality of joint policy.

Definition 3.2: In multiagent system with N agents the joint policy $\Pi^*=(\pi^{1*},\ldots,\pi^{N*})$ is optimal if each i agent policy π^{i*} is a best response given that the other agents follow their policies which constitute the joint policy.

The agent i policy on stage k is a best response if no other policies for the consistent information set of an agent has higher value, i.e.

$$V^{\pi^{i*}}(\varsigma_k)\geq V^{\pi^i}(\varsigma_k),\ \pi_k^{i*}\in\Pi_k^*. \tag{7}$$

Now the problem is how to evaluate the policy value if the information set ς is not unique. Regarding the system design issue on agents' goal in system it could be separated two kinds of acting manners of agents. The first is a risk aware manner, i.e. the agent avoids the worst situations. And the second case is where the agent strives to gain the best performance of the actions. The agent goal of avoiding the worst in multiagent setting is obtained by following the policy of a best response defined by (7), where the set of policies of each agent is selected with values minimizing over the observations $\bar{\mu}$ and maximizing over its own action set α. Whereas, striving to gain the best is described as to eliminate the set of policies with maximum value over the environment observations and taken actions, and apply the search of (7). By expression (5) the above considerations are expressed as below.

1) Policy value update in a risk aware manner,

$$V^{\pi}(\varsigma_k)=\sum_k^T \min_{\mu_{k+1}\in\bar{\mu}_{k+1}} r(\mu_{k+1})+\max_{a\in\alpha} V^{\pi}(\varsigma_{k+1}). \tag{8}$$

2) Policy value update of gaining the best manner,

$$V^{\pi}(\varsigma_k)=\sum_k^T \max_{\mu_{k+1}\in\bar{\mu}_{k+1}} r(\mu_{k+1})+\max_{a\in\alpha} V^{\pi}(\varsigma_{k+1}). \tag{9}$$

C. Monte Carlo Approach

A Monte Carlo method of policy iteration involves alternating complete steps of policy evaluation and policy improvement computation, beginning with an arbitrary policy and ending with the optimal policy. For multiagent system case it would be natural to explore joint policies of agents.

The multiagent system is simulated with M independent episodes, $e_1,e_2,\ldots e_M$, each run of T stages. An episode history of an agent consists of information set within this episode, $\varsigma_{e_M}=\varsigma_{1_{e_M}}\varsigma_{2_{e_M}}\cdots\varsigma_{T_{e_M}}$.

The experience samples of observed state space and explored actions determine the policy which is to be optimized. The optimized matching function of information of state observations ς and action set α derives the optimal policy for the given environment episode.

Monte Carlo method estimates the policy value consistent with every appeared episode history. The prior probability of agent history in a simulated environment is the product of probabilities for every episode.

$$\rho(\varsigma)=\prod_{j=1}^{M}\rho(\varsigma_{e_j}). \tag{10}$$

From (3) ρ is the environment reaction distribution to the agent policy π. Thus the agent policy in the system is a product of derived policies on each episode:

$$\pi^*(\varsigma)=\prod_{j=1}^{M}\pi^*(\varsigma_{e_j}). \tag{11}$$

It is supposed to converge to a true value as the number of generated episodes is large.

IV. Policy Control in Multiagent System (PCMA)

In view of a simulation purpose, it is possible to evaluate each policy for the given episode deriving policy distribution in a state environment which represents the optimal behavior in a given episode. On the other hand, evaluating any policy

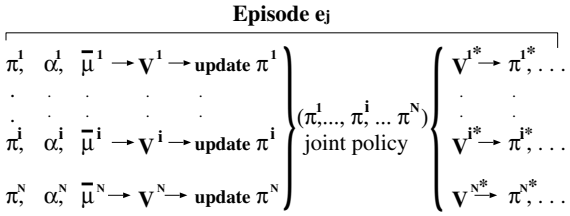

Fig. 2. Diagram of Policy Control Algorithm for N Agents System

for all simulation episodes could result in a policy assessment being made for a particular simulated environment.

The proposed algorithm of Policy Control in Multiagent System (PCMA) outlined in Table 1 and the flow diagram is shown in Figure (2). The large number of episodes are generated with arbitrary policies for each agent. For termination purpose of policy evaluation process the very small number ε is used. Within the episode an agent follows the given policy with action set α. After each action an agent observes the environment state μ and records the information set ς as defined in Definition 3.1. Since the environment response is not unique an agent derives its own policy regarding the two acting manners where the value evaluation for each information set is held by expressions (8) and (9). Policies are updated due to their values and the actions are generated following these policies. On the nest step, from the view of joint policy Π a best response policy π^{i*} is selected by (7). This process is repeated until best response policy functions in sequential runs have about the same values. Then the optimal joint policy for the given episode is determined by final individual policy of each agent. At the end of simulation the agent policy in the system and the joint policy matrix are composed by (11).

1. Generate the episode e_j, $j = 1$ *to* M.
(a) Episode initialization: for $i = 1$ *to* N $\varepsilon \leftarrow$ very small number, $\pi^i \leftarrow$ arbitrary, $\Pi^* \leftarrow$ empty list, $V^i \leftarrow 0$, $\varsigma^i \leftarrow$ empty list. (b) Implement α^i following π^i. (c) Collect $\bar{\mu}^i = \{\mu^i\}$, $\varsigma^i \leftarrow (a^i, \mu^i)$. (d) By (8) and (9) re-evaluate V^i. (e) Update the set of π^i, generate $\alpha^i = \{a^i\}$. (f) Select $\pi^{i*} \in \Pi^*$ by (7). (g) If $\pi^{i*} - \pi^i \geq \varepsilon$ then $\pi^i = \pi^{i*}$, Goto step (b), else Goto step (h). (h) $\Pi^* \leftarrow \pi^{i*}$.
2. Compute $\pi^*(\varsigma)$ by (11) . 3. And compose $\Pi(\pi^*)$ by (12).

TABLE I
ALGORITHM OF POLICY CONTROL IN MULTIAGENT SYSTEM (PCMA)

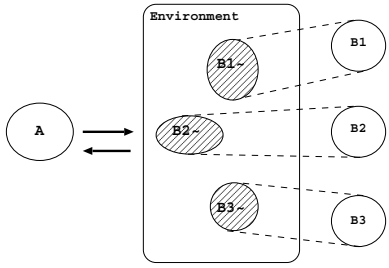

Fig. 3. Multiagent Model

The next section discusses the simulation of the Multi-vehicle Control Problem based on PCMA algorithm described above.

V. SIMULATION OF MULTI-VEHICLE CONTROL PROBLEM

Multiple vehicles domain, where there is no communication among the vehicles and sensing of vehicle is limited, is represented as an application model of simulation. This model could be used for maneuver derivation and evaluation in air traffic control. Air vehicles have limited sensing feature and limited action possibilities. Due to aerodynamic physics, any action or maneuver has some defined properties and takes time to be implemented. In uncertain unpredictable situations it is difficult to quickly react in a proper manner. Thus the strategy or policy of safe actions in a given situation must lead to a safe result as well. The ability to learn the policies to match current and future situations based on past experience, and to avoid non-desirable results is most important.

On Figure (3), agent A models the $B1$, $B2$, $B3$ agents behaviors matching them locally with its own view of optimality. To do that agent A observes or senses the signals from environment episode involving $\tilde{B}1$, $\tilde{B}2$ and $\tilde{B}3$ respectfully and records the obtained observations as a state information:

$$\mu^A_{1\ldots k} \leftarrow (\tilde{B}1, \tilde{B}2, \tilde{B}3)$$
$$\varsigma^A_k = (\mu^A_1, \ldots, \mu^A_k) \tag{13}$$

Agent A simulates a situated environment while generating dynamic episodes, perceives the other agents dynamics based on local interactions, records and computes the necessary values and builds an "own" view of its surrounding world by expression (13).

A. System Design

The following assumptions are taken for simulation conduction.

- The world consists of a cell representing the state of the environment from the view of agent A.
- A senses agents $B1$, $B2$ and $B3$ as moving obstacles with uncertain behavior.
- The agent A sensing function has normal distribution with mean on its own position and fixed deviation. The region is separated into states of perception determining the local view of environment states, where the other agents $B1$,

$B2$ and $B3$ are sensed by A agent as changing state observations value.

- The action set consists of maneuvers taken by an agent. Each action has cost, that is depending upon how many actions an agent carries out then the pay off increases accordingly.
- Agent A avoids obstacles as much as possible, since collision is probably fatal. There are 2 targets also moving in this environment. A perceives targets as attractive goals and tries to reach them.

Assume agent A has n number of sensing states and the sensing parameters are limited to 2, and the local state observation function μ is represented as 2 dimensional state space, where each element represents the observed state transition signal as shown by matrix (14). In order to make the observing (or sensing) ability closer to a real world case, the agent perceives its surrounding with a sensing function of Gaussian distribution. Here, an agent itself has mean value, and other agents or obstacles are sensed according to the Gaussian variance value. In other words, the closer an obstacle or other agent, the higher the sensing value and vice versa. The 2 dimensional sensing matrix μ size of $n \times n$ with signals from the environment to perceive the state information involving other moving agents or obstacles is shown below:

$$\mu = \begin{pmatrix} \mu_{11} & \dots & \mu_{1i} & \dots & \mu_{1n} \\ . & & & & \\ \mu_{j1} & \dots & \mu_{j_i} & \dots & \mu_{jn} \\ . & & & & \\ \mu_{n_1} & \dots & \mu_{ni} & \dots & \mu_{nn} \end{pmatrix} \tag{14}$$

where any object in state i to j sensed as activated element s_{ij} with value according to Gaussian variance in this state.

B. Experimental Study

The action set of five kinds of possible actions are considered: action0 - No Maneuver, action1 - Speed Maneuver, action2 - Horizontal Angle Maneuver, action3 - Vertical Angle Maneuver, and action4 - Random Maneuver. Policy in turns is simulated and learned by a Policy Control in Multiagent System (PCMA) algorithm described above. Here Agent A:

- simulates the episode with initial random policy to be gradually changed into a better one; senses environment states while acting toward target states;
- receives the sensed signals while moving towards policy to be evaluated; since each action has cost, policy has value of accumulated action costs on a simulation run;
- avoids the undesirable sensed objects in future steps of computation, i.e, computes the action values regarding future rewards, where the future obstacle collisions may decrease policy values in the updating process;
- learns its own optimal behavior in the range of two acting manners, risk aware and gaining the best;
- repeats the above steps until updated policies are nearly equal and derives the most optimal policy for the given episode, and finally for the system.

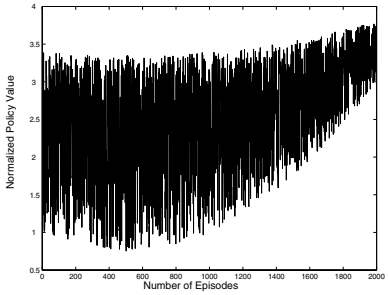

Fig. 4. Optimal Policy Distribution of agent A

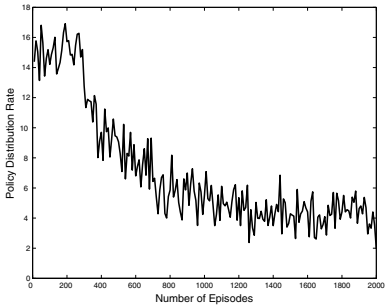

Fig. 5. Randomness Rate of Joint Policy

The above process repeats for 2000 random episodes. Running simulation episodes provide the agent to adapt to the surrounding environment, i.e. agent would be able to predict the system behavior and plan its own future policy.

C. Simulation Result and Discussion

Simulation result of optimal policy distribution of agent A which is derived by PCMA algorithm policy evaluation is shown in Figure(5). The horizontal axis is according to number of simulated episode runs. Vertical axis shows agent A policies evaluation as a normalized reward function which represents the computed optimal policy V^* at the end of episodes. Since agent A learns about its surrounding environment during simulation process, agent A strives to increase this reward function gradually. In other words, if an agent learns a given situation while receiving some reward price, then the next time the agent will try to spend less resources in a similar situation in order to obtain a higher reward.

As depicted on Figure (5) the action value evolution of learning agent A in a multi-vehicle simulated system is increasing with the increasing number of simulated episodes. At the beginning stage, the agent is supposed to be learning the environment which is reflected in more spending on resources as exploration and exploitation rates are high. While the number of simulated episodes increases, i.e. with increasing episodes to be evaluated, the agent obtains more information to be useful for future computations. And the policy evaluation range eventually gets narrower.

Figure (6) shows the distribution rate of joint policy of agents A, $B1$, $B2$, $B3$, computed episode-by-episode on the

basis of a discrete randomness formula

$$Rate(\Pi^*) = -\sum_{\pi^{i*} \in \Pi^*} \pi^{i*} \log(\pi^{i*}) \quad (15)$$

The joint action distribution rate (15) represents the degree of randomness of evaluated policies during the simulated PCMA learning. The larger this rate, the larger is the degree of randomness of the taken policies. The nature of distribution of joint actions in the simulated process is random for uncertain dynamics. During the learning episodes, the degree of randomness decreases due to agents policies' convergence to the optimal value which is defined by the Monte Carlo exploring approach.

As each agent policy derivation process depends on other agents policies, the policy is a function of not just a single agent action but a function of joint action formated by a PCMA learning process. By evaluating and improving the policies the agent decreases the randomness of taken actions which leads to decreased randomness of joint policies.

For a given system with 3 other dynamically moving objects it could be seen that agent A improves its own performance during the simulation progress. This result may lead to the observation that in a non-communicating multi-agent domain, where state and action spaces are discretisized due to domain issues, each situated agent may build its "own" view of the environment by trial-and-error method through Reinforcement Learning. Even if direct communication does not occur among the agents, building their "own" view based on "own" senses provides agent to fill this gap.

VI. Conclusion

This paper proposes a policy control algorithm for multiagent system. An agent learns the optimal policy by interaction with the environment adapting its own action function based on the local view. The Reinforcement learning method with Monte Carlo Control approach with application to multiagent system is used for computation. To cope with missing knowledge, agents perceive each other's current behavior as part of the environment's dynamic and build their policy derived from sensed states of the environment. Coordination among agents may occur through global environment influence, where prediction of environment change based on past experience which involves the influence of other agents behavior as well.

The main points of this method are:

1) Instead of directly communicating with others an agent learns the environment dynamics which involves the behavior influence of other agents and eventually comes up with optimal policy control. Here the optimality is considered by system's joint policy setting.
2) The history dependent agent's internal state makes it possible to involve a wider policy evolution space and to describe agent policies which are dynamic over interaction.
3) Simulation of independent episodes and learning from episode output can provide a powerful predictive tool providing a suitable mechanism for dynamic system control.

For some real world systems establishing communication for coordinated actions could be partially effective or even impossible and then the behavior of the decision maker is based upon observations of environment during the process. In such situations it would be more effective to evaluate the available resources in various situations before system implementation.

However, in order to investigate more general features of decentralized decision making in multiagent system, appropriate interaction problem considerations must be made. Game theory as the study of problems of conflict and cooperation among independent decision makers is assumed to be suitable for this purpose. The next unclear issue is an agent learning which concerns not just only it own history of actions and rewards, but uses observed histories of opponent actions to predict future states, and responds optimally to them. Thus, assumption methods providing approximate prediction of system dynamics should be studied in future work.

References

[1] D. P. Bertsekas, *Dynamic Programming and Optimal Control: 2nd Edition*, 2nd ed. Athena Scientific, Sept 2001, vol. II.

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

[3] C. Tomlin, G. Pappas, and S. Sastry, "Conflict resolution in air traffic management : A study in multi-agent hybrid systems," *IEEE Transactions on Automatic Control*, vol. 43, no. 4, pp. 509–521, 1998.

[4] M. Klein, "Supporting conflict resolution in cooperative design systems," *IEEE Transactions on Systems, Man, and Cybernetics (Special Section on DAI)*, vol. 21, no. 6, 1991.

[5] D. Nicolas and A. Jean-Marc, "Optimal resolution of en route conflicts," Laboratoire d'Optimisation Globale, CENA at ENAC, Toulouse, Tech. Rep., 1997.

[6] C. Boutilier, "Sequential optimality and coordination in multiagent systems," in *IJCAI*, 1999, pp. 478–485.

[7] G. Weiss, Ed., *Multiagent Systems. A Modern Approach to Distributed Artificial Intelligence*. The MIT Press, 2000.

[8] L. Fletcher and S. Dale, "Monte carlo matrix inversion policy evaluation," in *Proceedings of the 19th Annual Conference on Uncertainty in Artificial Intelligence (UAI-03)*. San Francisco, CA: "Morgan Kaufmann Publishers", 2003, pp. 386–393.

[9] A. Barto and M. Duff, "Monte carlo matrix inversion and reinforcement learning," *Neural Information Processing Systems*, vol. 6, pp. 687–694, 1994.

[10] G.Tesauro and G. Galperin, "On-line policy improvement using monte-carlo search," in *Advances in NIPS*, M. M.C.Mozer and T.Petsche, Eds. MIT Press, 1997, vol. 9.

[11] A.Damba and S.Watanabe, "Intelligent control in dynamic system," in *Proceedings of IEEE Conference on CIS and RAM*. Singapore: EI Compendex Database, December 2004.

[12] A. Damba and S. Watanabe, "Policy control in multiagent system," in *Proceedings of Computational Intelligence 2005*, International Association of Science and Technology for Development (IASTED). ACTA Press, July 2005.

[13] J. Schmidhuber, "A general method for multi-agent reinforcement learning in unrestricted environments," AAAI Press. AAAI Technical Report, Menlo Park, CA, Tech. Rep., March 1996.

[14] E.H.Durfee, V.R.Lesser, and D.D.Corkill, *Cooperative Distributed Problem Solving*, ser. Reading, MA. Addison-Wesley, 1989, vol. 4.

[15] P.Xuan, V.Lesser, and S.Zilberstein, "Communication in multi-agent markov decision processes," in *Proceedings of the Fourth International Conference on MultiAgent Systems (ICMAS-00)*, 2000.