A NEW APPROACH USED FOR ANTICIPATION TO IMPROVE THE SYSTEM'S BEHAVIOUR

Ahmed M. Elmahalawy and Pavel Nahodil
Department of Cybernetics,
Czech Technical University,
Karlovo námestí 13, 121 35 Prague 2, Czech.
E-mail: elmahal@labe.felk.cvut.cz
nahodil @fel.cvut.cz

KEYWORDS
Anticipatory Behaviour, Artificial Life, World of Artificial Life Simulator, Markov Chain, Similarity Distance.

ABSTRACT
Anticipation occurs in all spheres of life. Nature evolves in a continuous anticipatory fashion targeted at survival. Conscious reaction takes too long to process. Motivation mechanisms in learning, arts, and all types of research are dominated by the principle that an expected future state controls present action. The study of anticipatory behaviour refers to behaviour that is dependent on predictions, expectations, or beliefs about future states. Anticipation is an important behaviour in life. Recently, it is used in different system to improve its performance. In our work, we begin by making a modification in a simulator WAL – (World of Artificial Life) to add the bases of anticipation. Also, we try to introduce a new approach for anticipation based on the Markov chain; we will see that with anticipation the performance of the system will be better. We apply our approach on a well known problem - a maze problem.

INTRODUCTION

Basic definition of anticipatory systems was published in 1985 by cyberneticist R. Rosen in his book "Anticipatory systems". He defined an anticipatory system as follows: "*A system containing a predictive model of itself and/or its environment, which allows it to change state at an instant in accord with the model's predictions pertaining to a latter instant*". [Kohout and Nahodil 2007, Dubois 2003, Butz et. al. 2003a]

A tentative definition of anticipation could be: *an anticipatory system is a system for which the present behaviour is based on past and/or present events but also on future events built from these past, present and future events*. [Dubois 2003]

In this paper, we make a modification in WAL to have good performance of the system and then suggest a new approach that is used in anticipation. This approach is based on the principals of Markov Chain. We try to verify that using anticipatory behaviour in this system will improve its performance.

This paper begins in section 2 by explaining the anticipatory behaviour. In section 3, a short description of the simulator WAL and architecture of agent is shown. Agent Architecture for Anticipation has been described in section 4. In section 5, a suggested modification of WAL and Anticipatory Agent with Markov Chain is presented. The description of the maze environment is shown in section 6. In section 7, the evaluation of our new approach for anticipatory behaviour is presented. Finally, a conclusion and future work are briefly described in the last section 8.

ANTICIPATION BEHAVIOUR

In artificial intelligence, anticipation is the concept of an agent making decisions based on predictions, expectations, or beliefs about the future states. It is widely recognized that anticipation is a vital component of complex natural cognitive systems. Anticipation seems to be suitable for key role in design and realization of anticipatory behaviour. [Kohout and Nahodil 2007]

Several recent attempts have been made in artificial intelligence to integrate anticipatory mechanisms into artificial learning systems in the framework of reinforcement learning, learning classifier systems (as online generalizing reinforcement learners) and related systems, as well as neural networks. So far, research in artificial intelligence has included anticipatory mechanisms wrapped in model learning systems such as the model-based reinforcement learning approach. Anticipatory processes were never analyzed on their own. [Butz et. al. 2003a]

There are different types of anticipatory behaviour as [Butz et. al. 2003b]

(1) Implicit anticipatory mechanisms where no actual predictions are made but the behavioural structure is constructed in anticipatory fashion.
(2) Payoff anticipatory mechanisms where the influence of future predictions on behaviour is restricted to payoff predictions.
(3) Sensory anticipatory mechanisms where the future predictions influence sensory (pre-) processing.

Proceedings 21st European Conference on Modelling and Simulation
Ivan Zelinka, Zuzana Oplatková, Alessandra Orsoni ©ECMS 2007
ISBN 978-0-9553018-2-7 / ISBN 978-0-9553018-3-4 (CD)

(4) State anticipatory mechanisms where predictions about future states directly influence current behavioural decision making.

Anticipatory behaviour appears useful in many situations allowing for previously impossible behavioural patterns: [Butz et. al. 2003a]

1- Anticipatory processes can stabilize behavioural execution.
2- Anticipations may guide, or canalize, behavioural flow.
3- Anticipatory mechanisms can bias attention processes enabling goal-directed focus and faster reactivity.
4- Anticipatory behaviour may result in advantages in hunting and other competitive scenarios.
5- Anticipatory behaviour may result in faster adaptively in dynamic environments by the means of internal reflection and planning.
6- Cooperative behaviour may be improved and suboptimal behaviour may be overcome by preventive state anticipatory behaviour.
7- Anticipatory behaviour appears to be an important prerequisite for social interaction.

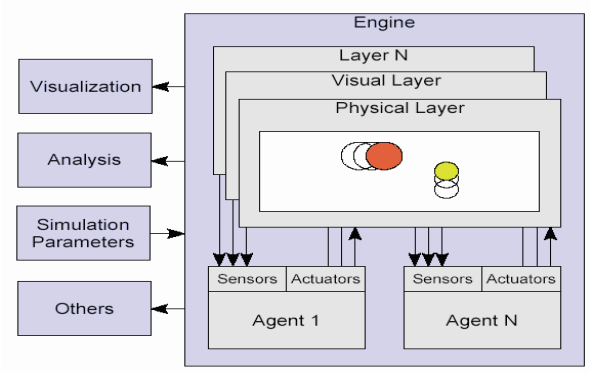
In particular, future research on anticipatory behaviour may lead to [Butz et. al. 2003a]

(1) Significant improvement of the behaviour of adaptive learning systems;
(2) Further understanding of the function of anticipatory mechanisms in animals and humans;
(3) The creation of social interactive systems with human-like anticipatory features;
(4) The discovery of the processes underlying motivations and emotions;
(5) The development of truly cognitive systems that do not only reactively move through the world but learn about important resemblances, contiguities, and causes of effects, and efficiently exploit this knowledge by anticipatory behaviour mechanisms

WORLD OF ARTIFICIAL LIFE SIMULATOR

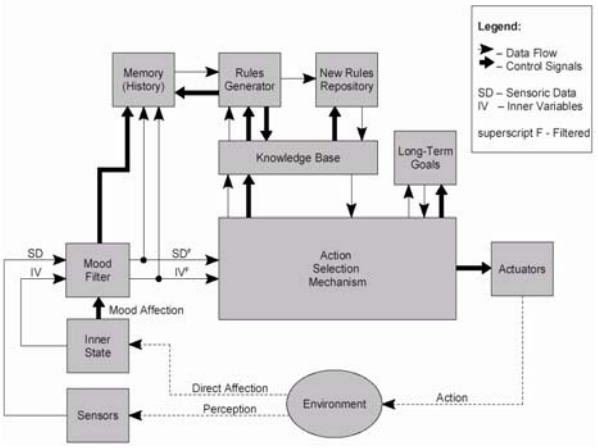
One of the most recent architectures of Multi-Agent Systems (MAS) for ALife domain is World of Artificial Life (WAL) simulator. WAL architecture is based on hybrid architecture for ALife agent that would be emergent, proactive, sociable, autonomous and adaptive. Agent's learning is based on algorithms from Artificial Intelligence (AI) and movement planning agent uses concept of artificial potential field known from mobile robotics. Algorithms from AI were chosen as an alternative to evolutionary algorithms and Artificial Neural Networks, which are commonly used in ALife domain. This simulator has been developed by the research group *Mobile Robotic Group* "MRG" (leaded by P. Nahodil) in previous years [Foltýn 2005].

We can see the basic WAL architecture in Figure 1.



Figures 1: WAL abstract architecture

In Figure 2 we can see the whole architecture of agent. The architecture of agent is based on processing data from sensors and inner states. If we follow data flow, we start in the block Environment. Environment supplies agent through Sensors with sensory data and through Inner States block with information about inner variables. Data are filtered by Mood Filter and passed to Action Selection Mechanism and Memory. Action Selection Mechanism compiles available data and using Knowledge Base and Long-Term Goals agent makes decision about future actions which are performed by Actuators. History, Rules Generator and New Rules Repository are used for generalization [Foltýn 2005].



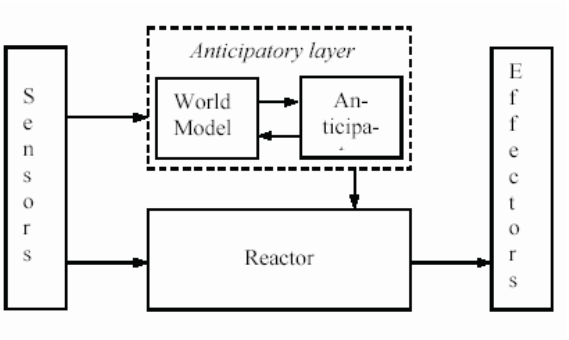
Figures 2: Block scheme of architecture of agent

AGENT ARCHITECTURE FOR ANTICIPATION

Agent is a computational system which is long-lived, has goals, sensors and effectors and decides autonomously which action to take in which situation to maximize progress towards its (time varying) goals. Agents have been around for a number of years. It is not easy to find a common definition of an agent because of its diverse mainstream characteristics: level of autonomist, reactivity, proactively, collaboration, social abilities etc. [Rehor et. al. 2003].

In the suggested framework, an anticipatory agent consists mainly of three entities: an object system (*S)*, a world model (*M*), and a meta-level component (*Anticipator*). The object system is an ordinary (i.e., non-anticipatory) dynamic system. *M* is a description of the environment *including S*, but excluding the *Anticipator*. The *Anticipator* should be able to make predictions using *M* and to use these predictions to change the dynamic properties of *S*. Although the different parts of an anticipatory agent are certainly causal systems, the agent taken as a whole will nevertheless *behave* in an anticipatory fashion. [Seger and Törnqvist, 2002, Davidsson, 2004]

When implementing an anticipatory agent, what should the three different components correspond to? And what demands should be made upon these components? To begin with, it seems natural that *S* should correspond to some kind of reactive system similar to the ones mentioned above. We will therefore refer to this component as the *Reactor*. It must be a fast system in the sense that it should be able to handle routine tasks on a reactive basis and, moreover, it should have an architecture that is both easy to model and to change. The *Anticipator* would then correspond to a more deliberative meta-level component that is able to run in the world model faster than real time. When doing this it must be able to reason about the current situation compared to the predicted situations and its goals in order to decide whether (and how) to change the *Reactor*. The resulting architecture is illustrated in Figure 3. [Seger and Törnqvist 2002, Davidsson 2004]

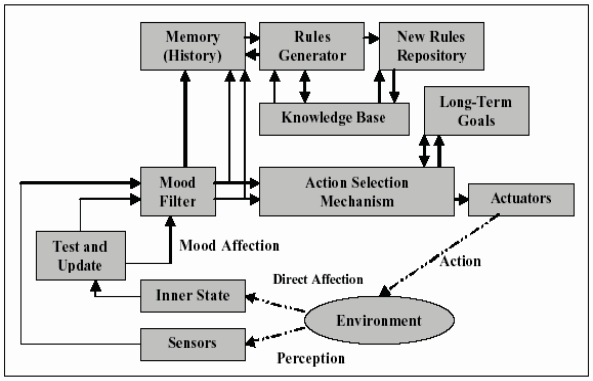


Figures 3: The basic architecture of an anticipatory agent.

To summarize: The sensors receive input from the environment. This data is then used in two different ways: (1) to update the World Model and (2) to serve as stimuli for the Reactor. The Reactor reacts to these stimuli and provides a response that is forwarded to the effectors, which then carry out the desired action(s) in the environment. Moreover, the Anticipator uses the World Model to make predictions and on the basis of these predictions the Anticipator decides if, and what, changes of the dynamical properties of the Reactor are necessary. Every time the Reactor is modified, the Anticipator should, of course, also update the part of the World Model describing the agent accordingly. Thus, the working of an anticipatory agent can be viewed as two competing processes, one reactive at the object-level and one more deliberative at the meta-level. [Davidsson, 2004]

SUGGESTED MODIFICATION OF WAL

The drawback that was found in the previous architecture of agent in WAL is missing the real values of inner state values. It was assumed that all inner state values have ideal in the beginning of the experiment. So, our modification is adding a new block to this architecture to test the inner state values at the beginning of the experiment. If they have not its ideal values so do action to adjust them to its ideal values. This modification is based on the principles of anticipation. That is shown in Figure 4.



Figures 4: The proposed modification in architecture of agent

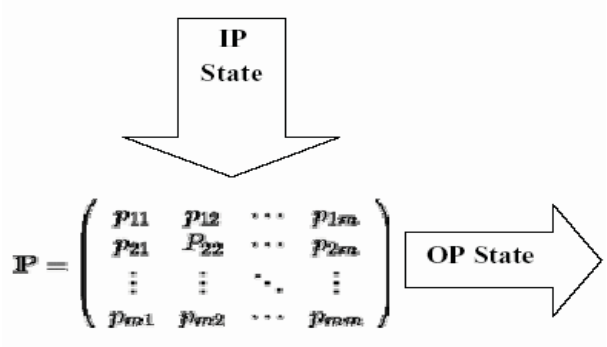
ANTICIPATORY AGENT WITH MARKOV CHAIN

The Markov chain is an important mathematical principle that will be useful not only in the study of statistics in college, but also in real-life problems or situations. A systems engineer will find this mathematical principle quite useful in his career along with the study and analysis of discrete dynamical systems which model real-life situations. Also, it can be used to solve a very useful class of problems in a rather remarkable way.
[http://en.wikipedia.org/wiki/Markov_chain]

A Markov chain is a sequence of random values whose probabilities at a time interval depend upon the value of the number at the previous time. The controlling factor in a Markov chain is the transition probability; it is a conditional probability for the system to go to a particular new state, given the current state of the system. For many problems, such as simulated annealing, the Markov chain obtains the much desired importance sampling. This means that we get fairly efficient estimates if we can determine the proper transition probabilities. [Carter Jr. 1996]

Markov chains have many applications in biological modelling, particularly population processes, which are useful in modelling processes that are (at least) analogous to biological populations. [MLpedia. 2006]

The construction of a Markov chain requires two basic ingredients, namely a transition matrix and an initial distribution. A finite number of states: consider a population distributed among n states (state 1, state 2, . . state n). The tendency of the population to move among the n states can be described using an n X n matrix, called the transition matrix: [Aldous 1999]

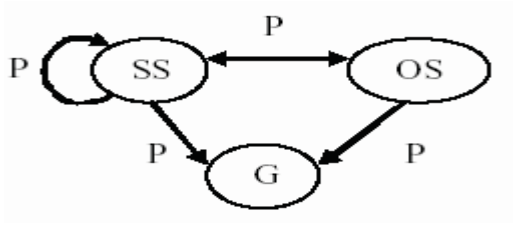


P_{ij} is the probability of moving from state i to state j.

Based on the previous description of Markov Chain's, we can state our suggested method for anticipation behavior. Our method has the following steps:

1- Start with initial state.
2- Calculate the probability of moving from this state to the same state and other state, depending on the similarity distance.
3- The more similarity distance, the more probability. We take the state with higher similarity distance as the next state.
4- Test this state and see if it is desired or undesired, according to some rules about the environment.
5- If it is undesired the goto step 2 and repeat the algorithm.
6- If it is desired then see if it is not our goal, goto step 2 and repeat the algorithm.
7- If it is our goal then stop the algorithm.

The following Figure 5 illustrates our idea about moving from one state to other. Where P is the probability, SS is the same state, OS is the other state and G is the goal.



Figures 5.Algorithm state transition

Now, it is important to know how to calculate the transition probabilities. In our method, we consider the similarity distance to compare the two states and according to the results of comparison, the transition probability is calculated. There are three different distance measures provided as Euclidean Distance, Manhattan Distance and Hamming Distance. In any of the given measures, the lower the distance between two elements, the more similar the elements are deemed to be the measures are as the following laws [Black 2006]:

In our method, we will use the hamming distance to calculate the similarity between states. This measure gives a value of 0 or 1 to distance between two elements in any one dimension. The distance is 0 if the assigned values are the same and 1 otherwise. The hamming distance over *n* dimensions is therefore the number of attributes which have different values:

$$S = \sum_{i=1}^{n} F(X_i, Y_i)$$

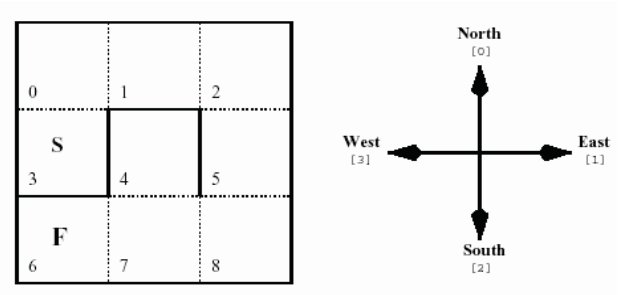
where $F(X_i, Y_i) = 0$ if $X_i = Y_i$ else $F(X_i, Y_i) = 1$

This similarity distance is very important for that algorithm, where it depends on it.

THE ENVIRONMENT

Maze problems have been widely used in the learning classifier system literature to evaluate performance. The agent is randomly placed in the maze. The task is to reach to the closest food position. [Gérard and Sigaud 2001, Butz 2002, Ramos et. Al. 2005]

We use the maze problem described in Figure 6 in order to evaluate our algorithm. This maze illustrates a state transition diagram with nine states and four possible transitions starting from each state [Gérard and Sigaud 2001]



Figures 6: A simple maze problem

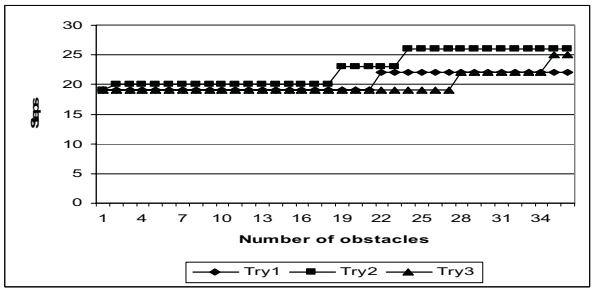
The environment is a small two-dimensional grid (25x25 units) in which a number of unit-sized square obstacles form a maze. The goal of an agent is to pick up a number of targets by finding paths from its current position to the positions of the targets. The agent is able to move in four directions (north, south, east, and west), unless there is an obstacle that blocks the way. The agent is always able to change its direction to the target if there is any obstacle in its way to target.

EVALUATIONS

We will present many experiments using the simple maze problem. In each case, we run the system once without anticipatory module and another time with anticipatory module. In each experiment, we use different maze configuration that is, we change the obstacles position.

System without Anticipatory Module

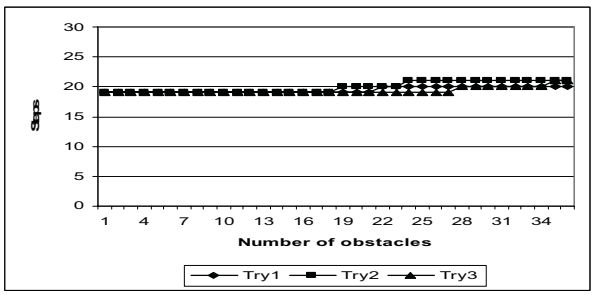
We use single agent without anticipatory module and single goal (SASG) in this experiment. The numbers of obstacles are changed from 0 to 35. The number of steps that agent will take to reach its goal is calculated. Figure 7 shows our results.



Figures 7: Number of steps for system without anticipatory module (SASG).

System with Anticipatory Module

We add anticipatory module to the system and use single goal in this experiment. The number of obstacles are changing from 0 to 35 and calculate the number of steps that agent will take to reach its goal. Figure 8 shows our results.



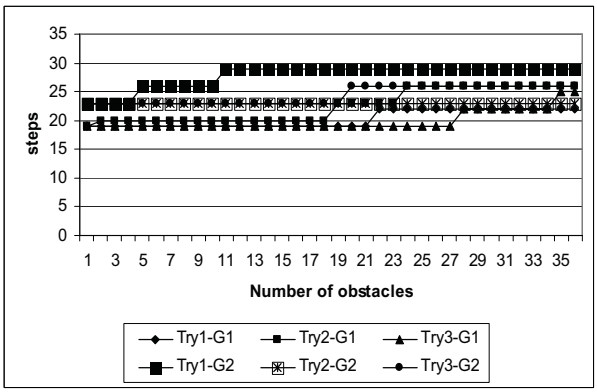
Figures 8: Number of steps for system with anticipatory module (SASG).

As we saw in figures 8 and 9, that the performance of the system with anticipatory module is better than the performance without anticipatory module. For example, when we have 35 obstacles in the environment, it takes 26 steps to reach to the goal when we don't use anticipation module. In the case of using anticipation module, it will take 21 steps.

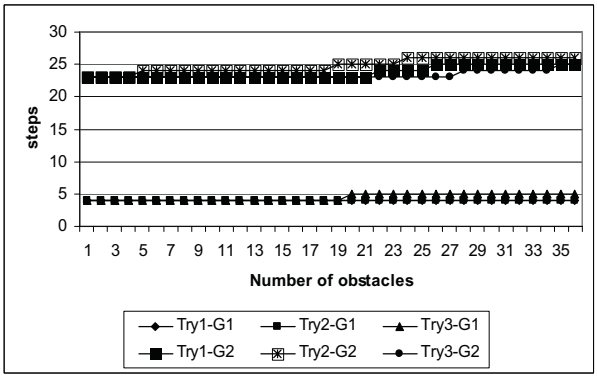
System has Multi Goal

We will repeat the previous experiments with and without anticipatory module in the system using multi goal (SAMG).

Figure 9 and 10 resent the system results without and with using anticipatory module respectively.



Figures 9: Number of steps for system without anticipatory module (SAMG).



Figures 10: Number of steps for system with anticipatory module (SAMG).

Also, in the case of multi goal as in figures 9 and 10, we can see that the max number of steps to reach the goal is 29 steps for system without anticipatory module and 26 for the system with anticipatory module.

Also we can compare the performance of the system with and without anticipator module by calculating the overload percentage of steps that agent must walk to reach to its goal. This overload can be calculated as follows:

Overload=

[(max number of steps / min number of steps) -1]*100

The following table 1.shows the overload for the system in case of don't use anticipatory module and with using anticipatory module. We use the previous equation to calculate the overload when the system has 35 obstacles.

Table 1: The overload percentage of steps

	Without anticipator	With Anticipator
SASG	37%	11 %
SAMG	36 %	25 %

CONCLUSION AND FUTURE WORK

From the previous figures and table, it shows us the importance of anticipatory module in our system. We can see that the agent takes more steps to reach its goal without anticipatory module. This increases the overload percentage of steps. When the agent uses the anticipatory module, it can predicate the position of obstacles and then change its direction before collision with it. In this case, it takes less steps and the overload percentage of steps will be lower.

The anticipatory behaviour is the most important in ALife domain. Our future work will focus on two branches. The first one is to find other methods that can be used in cooperation with Genetic Algorithms or Reinforcement Learning to receive better results. The second is to use many agents in our system with and without anticipatory module and see the performance of the system.

REFERENCES

Aldous D. 1999. Reversible Markov Chains and Random Walks on Graphs. [WWW] Available from: http://www.stat.berkeley.edu/users/aldous/RWG/book.html. [Accessed: 31 January 2007]

Black P.E. 2006. Dictionary of Algorithms and Data Structures. [WWW] Available from: http://www.nist.gov/dads/. [Accessed: 31 January 2007]

Butz M. V. 2002. State value learning with an anticipatory learning classifier system in a Markov decision process. IlliGAL Report 2002018, Illinois Genetic Algorithms Laboratory, University of Illinois at Urbana-Champaign. [WWW] Available from: http://wwwilligal.ge.uiuc.edu

Butz M. V. Sigaud O. and Gerarad P. 2003. Anticipatory Behaviour: Exploiting Knowledge about the Future to Improve Current Behaviour. In: Butz, M., Sigaud, O., and Gerard, P. (eds.) Anticipatory Behaviour in Adaptive Learning Systems: Foundations, Theories, and Systems. LNAI. Vol. 2684, 1-10, Springer Berlin / Heidelberg

Butz M. V. Sigaud O. and Gerarad P. 2003. Internal Models and Anticipations in Adaptive Learning Systems. In: Butz, M., Sigaud, O., and Gerard, P. (eds.) Anticipatory Behaviour in Adaptive Learning Systems: Foundations, Theories, and Systems. LNAI Vol. 2684, 86-109, Springer Berlin / Heidelberg

Carter Jr. E.F. 1996. Random Walks, Markov Chains and the Monte Carlo Method. [WWW] Available from: http://www.taygeta.com/rwalks/rwalks.html. [Accessed: 31 January 2007O

Davidsson P. 2004. A Framework for Preventive State Anticipation. In: Butz, M., Sigaud, O., and Gerard, P. (eds.) Anticipatory Behaviour in Adaptive Learning Systems: Foundations, Theories, and Systems. LNAI. Vol. 2684, 151-166, Springer Berlin / Heidelberg.

Dubois D. M. 2003. Mathematical Foundations of Discrete and Functional Systems with Strong and Weak Anticipations. In: Butz, M., Sigaud, O., and Gerard, P. (eds.) Anticipatory Behaviour in Adaptive Learning Systems: Foundations, Theories, and Systems. LNAI Vol. 2684, 110-132, Springer Berlin / Heidelberg.

Foltýn L. 2005. Realization of Intelligent Agents Architecture for Artificial Life Domain. Diploma Thesis on Dept. of Cybernetics, supervised by Nahodil, P., Faculty of Electrical Engineering, CTU in Prague, Prague 2005

Gérard P. and Sigaud O. 2001.YACS: Combining Dynamic Programming with Generalization in Classifier Systems. In P.L. Lanzi, W. Stolzmann, and S.W. Wilson (Eds.): Advances in Learning Classifier Systems: Third International Workshop, IWLCS 2000, Paris, France. LNAI, 52-69, .Springer-Verlag Berlin Heidelberg.

Kohout, K. - Nahodil, P. 2007. Simulation of Anticipatory Behavior in the Artificial Life Domain. WSEAS Transactions on Information Science and Applications. March 2007, issue 3, Vol. 4, 568 - 575.

MLpedia.2006. [WWW] Available from: http://www.mlpedia.org/index.php?title=Markov_chain# References. [Accessed: 31 January 2007]

Ramos M. Berro A. and Duthen Y. 2005. Distributed Anticipatory System", F.F. Ramos et al. (Eds.): ISSADS 2005, LNCS, Vol. 3563, 443–451, Springer-Verlag Berlin Heidelberg.

Rehor D. Kadlecek D. Slavík P. and Nahodil P. 2003. VAT – A New Approach to Multi-Agent Systems Visualization. In Proc.: Visualization, Imaging, and Image Processing (VIIP 2003) Benalmádena, Spain, October 8-10

Seger C. and Törnqvist B. 2002. Linear Quasi Anticipation: An evaluation in real-time domains. Master Thesis, MSE-2002:10, Blekinge Institute of Technology, Ronneby Sweden

Eng. Ahmed M. Elmahalawy was born in Benha, Egypt and went to the Menoufia University, where he studies Computer Science and Engineering and obtained his degree in 1995. He worked as a demonstrator in Faculty of Electronic Engineering, Menoufia University. He had his MSc. in 2001and then worked as assistance lecture in Faculty of Electronic Engineering, Menoufia University. Now he is a PhD student in the Czech Technical University, department of Cybernetics. His supervisor is Pavel Nahodil. (see below).

Professor Assoc., Dr. Pavel Nahodil obtained his second - Scientific Degree at the Czech Technical University in Prague, Czech Republic, in 1979. Since 1986 he has taught Applied Robotics and Artificial Life at the Faculty of Electrical Engineering in Prague. He was a Visiting Professor at the Dept. of Artificial Intelligence TU in Edinburgh and at the Dept. of Informatics in Hanover. Currently he is still active at the Dept. of Cybernetics, FEE, CTU in Prague.
He is author or co-author more then 140 papers and publications, journal papers and invited lessons. His research interests include behavioural based robotics and artificial life approaches in general.