GENERAL ASPECTS OF CONSTRUCTING AN AUTONOMOUS ADAPTIVE AGENT

Arthur Pchelkin

Abstract: There are a great deal of approaches in artificial intelligence, some of them also coming from biology and neirophysiology. In this paper we are making a review, discussing many of them, and arranging our discussion around the autonomous agent research. We highlight three aspect in our classification: type of abstraction applied for representing agent knowledge, the implementation of hypothesis processing mechanism, allowed degree of freedom in behaviour and self-organizing. Using this classification many approaches in artificial intelligence are evaluated. Then we summarize all discussed ideas and propose a series of general principles for building an autonomous adaptive agent.

Keywords: reinforcement learning, neural networks, functional systems theory, inductive automaton.

Introduction

One of the directions in artificial intelligent (AI) is adaptive autonomous agents research (AAAR). This research direction started actively growing since 1985 [Maes95,Wil85], however, there were proposed to make researches in similar directions also before it [Bong75].

In this paper we analyse general problems that appear developing autonomous adaptive agents learning algorithm. The goal was to analyse different approaches, directly or indirectly connected with autonomous agent research, and to develop constructive principles for autonomous adaptive agent architecture.

We highlight three aspect in our classification: (1) type of abstraction applied for representing agent knowledge, (2) the implementation of hypothesis processing mechanism, (3) allowed degree of freedom in behaviour and self-organizing, and the structure of the paper is arranged in corresponding order.

Type of Abstraction

There are many different learning algorithms [Hunt75,Mit99,Fau94] well known in AI that model different aspects of intelligence. In the general case the agent needs ability to model practically each aspect of intelligence, because there is no possibility to specialize on some specific aspect of cerebration. In this case, a question appears: how to integrate so many approaches into single architecture?

Analyzing the question about the integration of different approaches, modeling different aspects of thinking, we concluded that there is a need to discuss what is common and what is different in these approaches. The common is that each of them is based on some type of abstraction, but the different is that in each specific case it could be applied specific kind of abstraction. To analyze this question, we selected several typical approaches (see tab. 1).

Table 1. The concrete and the abstract in different approaches

An approach	The concrete	The abstract
Neural networks with supervisor, e.g. error backpropagation algorithm [Fau94]	A training set of vector pairs	Neural network itself, i.e. synaptic weights
Neural network without supervisor, Kohonen feature map [Fau94]	A training set of vectors	Synaptic weights vectors – cluster centers
Clustering algorithms [Ryz77]	A set of vectors	Vectors – cluster centers
Decision tree learning, e.g. algorithm ID3 [Mit99]	A training set of object with known attribute values and classification	Decision tree
Finite automata learning [Ang87, RS93, She95]	An observation table or a sequence of pairs “action-perception”	Finite automata itself – a graph of states and transitions
Hidden Markov models learning [Rab89,Chr92]	A sequence of perceptions	A graph of probabilistic transitions between states
Suffix trees (for hidden Markov models)	A sequence of pairs “action-	Suffix tree

[Ron94, McCallum95]	perception"	
Statistical properties discovery, e.g. correlation	A set of vectors	Correlation matrix (between parameters)
Statistical parameters	A set of numbers	Average, variance and so on

Each approach has two levels: abstract and concrete information. Each approach could be described this way: there is some input information, i.e. the concrete, and it is made some assumption about the structure of this information, - the assumption is used for extracting the abstract from the concrete. Then the abstract is used for decision-making automatically or manually.

Autonomous adaptive agent needs the abstract for the similar purpose, i.e. for decision-making using analogies. Inserting specific kind of abstraction into the architecture of the agent could speed up and simplify the learning process, and in some cases it could be simplified even till several parameters adjustment. However, it must bound adaptability and flexibility of architecture that are very important properties of autonomous adaptive agent.

Inductive automaton: Several researchers have come through another way, e.g. Yemelianov-Yaroslavsky proposed a new neural network model, called "inductive automaton" [Yem90]. According to the author idea, specific aspects of intelligence, i.e. specific types of abstraction, must not be mechanically included into the inductive automaton architecture, but different types of abstraction must appear in the inductive automaton as a by-product of solving by it one single task – minimization on energy consumption by its neurons. Our analysis showed that this neural network is able to work as Kohonen [Fau94] map or as finite automata learning algorithms [Ang87] (of course, as some analogy).

The analysis of the last work brings us at the conclusion that it is more perspective not to integrate specific preprogrammed kinds of abstraction into the agent architecture, but to try to discover the main reason for appearing of abstraction in intelligent system, and inductive automaton [Yem90] is an interesting solution of the discussed problem.

Hypotheses Processing

As it was mentioned, intelligent system needs the abstract for decision-making. However, any abstraction is an assumption about properties of external environment, i.e. hypothesis. Let us analyze hypotheses processing in different approaches (see tab. 1).

Finite automata learning: Finite deterministic automata learning algorithms [Ang87, RS93, She95] employ a hypothesis about each of states known to algorithm. Such hypothesis is an assumption that the corresponding state is unique, and at each time moment learning algorithm work only with some approximation of learning automaton. This approximation is represented by graph where nodes are states and edges are transitions between the states. If the number of states in the learning automaton is not known then it is not possible to check: does the state in the model correspond to one state in the automaton, or it really corresponds to several states. That's why, in practice these algorithms work using such principle.

A model, containing such hypotheses, is used for solving different tasks in the environment, e.g. for calculating of optimal policy. However, if fault occurs, i.e. the response of the automaton conflicts with the model, then the learning algorithm automatically identify fallacious hypothesis and splits the corresponding state in the model. So, after the conflict identification the approximation is made corrected. For decision-making it is enough to have an abstract model (in this case the model of automaton), but to identify and correct a fallacious hypothesis, there is necessary to have links between the abstract and the concrete, used to extract the abstract. For example, in the paper [Ang87] an observation table is used, that contains all the concrete information, that is used for identification and correction of fallacious hypotheses.

Hidden Markov models: The similar principle is applied for hidden Markov models (HHM) learning [Rab89, Chr92]. It is made the same assumption that each state in the model is unique in HHM, and concrete information (a sequence of perceptions) is used for calculating of matrix that stores transition probabilities and for splitting of one state in order to improve likehood of the model.

Suffix trees: Some approaches [McCallum95] efficiently uses suffix trees [Ron94] in reinforcement learning with hidden state. In this case, the states of the environment are represented not by graph nodes but by suffix tree leaves. The algorithm uses hypotheses about each of leaves that it is a unique state in the environment.

However, if using a statistical test it is discovered the one of hypothesis is fallacious then the algorithm splits the corresponding leaf, adding new leaves below it. The concrete also is used to calculate transitions' probabilities.

Decision trees: Decision trees [Mit99] could be applied for pattern recognition. Decision trees learning algorithms, e.g. ID3, could be interpreted this way. At each time moment, a decision tree could be considered sufficient for making correct classification of objects from some known set. However, if a new object appears, that could not be classified correctly, the tree can be improved by adding new leaves. It should be noted, that the abstract, i.e. the tree itself, was sufficient for classification, but, to make a correction in this abstract, the concrete is needed, i.e. a training set.

Inductive automaton: Self-organizing of inductive automaton [Yem90] could be interpreted from the hypotheses processing point of view. Neurons, that frequently activates together, could be joined by excitatory links into assemblies. Each of neurons in such a group can have individual links to and from another neurons inside or outside of the assembly. The circumstance, that al the neurons are connected only by excitatory links, could be considered as a hypothesis, i.e. an assumption, that all the neurons in this group performs the same function, but maybe in different contexts.

The hypothesis about neurons functions similarity could be discovered to be wrong. In this case, there is a possibility to split assembly by inhibitory links. It occurs through a transformation of weak excitatory link into strong inhibitory link. As in the previous cases, to correct a wrong hypothesis, i.e. to split an assembly, it is necessary to have the concrete information, i.e. individual links between neurons, because the individual links decides how and where the assembly must be split. Neural assemblies in inductive automaton have also an interesting semantic interpretation, e.g. an assembly can correspond to some concept, and each neuron can represent some aspect of it. In this case, the splitting of an assembly could mean the formation of subconcepts or subclasses. Interesting ideas connected to this topic could be found in paper [Kus2000], however, in contrast with inductive automaton there is used fixed unchangeable structure of assemblies.

To sum up this part, we need to conclude that the abstract model building usually is made through integration of concrete information into the abstract model. During such integration, there is a need to make assumptions about correspondences between units of concrete information and units of the abstract representation. Therefore, the learning algorithm must be provided with the ability of efficient wrong hypotheses processing. Usually, decision-making needs only an abstract model, but in the case of fault it is necessary back to the concrete information for identifying and correcting of a wrong hypothesis in the abstract.

Degree of Freedom

In the phrase "autonomous adaptive agent" [Maes95] the word "autonomous" means the agent has high degree of freedom in decision-making, and word "adaptive" - high degree of freedom in self-organizing. Building an autonomous agent a question arises: where, when and how much freedom to give an autonomous agent? It is difficult to decide because the deficiency of freedom may dramatically bound abilities to adapt the custom environment, but redundant degree of freedom can produce chaos and instability. To illustrate this dilemma, let us consider traditional conflict between classifier systems and connectionist approaches.

Classifier systems: The idea of classifier systems [Hol86] was to provide an intelligent systems with a maximal degree of freedom in self-organizing in order to improve its adaptation abilities. Really, classifier systems (in original Holland framework) don't have potential boundaries in self-organizing possibilities, but at the same time there is not any paper presenting the full exploitation of these possibilities. Using classifier systems, it is possible to obtain good results only in specially simplified environments, e.g. the paper [Wil85] presented good results for a purely reactive classifier system, i.e. without a temporal memory. However, after improving this system by adding a temporal memory a series of negative results has been obtained [CR94]. The improved system was not able to guarantee stable behavior and self-organizing. Then a new improved version was developed [Wil95], but it also was able to show good results only in very simple environments.

Connectionist approaches: Connectionist approaches, employing artificial neural networks [Fau94] as an engine for information processing, allow much more limited degree of freedom for self-organizing in order to provide more stable behavior and self-organizing. In many cases the self-organizing of such a system could be considered as a parameter adjustment. At the same time, classifier system can be self-organizing through

different variations in behavior and successful constructions inventing. Therefore, it is possible to say that classifier systems have too much freedom but connectionist approaches employ too limited degree of freedom.

Inductive automaton: Inductive automaton [Yem90] can be considered as an compromises between two previous approaches. On the one hand, inductive automaton processes information as a neural network, but on the other hand, self-organizing is performed through successful construction fixing. Each neuron in this network has its own degree of freedom that depends on the age of the neuron. If a neuron frequently activates and takes active participation in the network processes then its age decreases, and, simultaneously, its degree of freedom also decreases. In opposite case, if a neuron is inactive for a very long time its degree of freedom increases. In this example, degree of freedom is hardly preprogrammed for a component of system, but depends on its functioning. Similar approaches, employing different degrees of freedom for different neurons, could be found in works [Amo73, Wick99].

Functional systems theory: Inductive automaton has a very serious disadvantage – it is not able to control degrees of freedom of its components, i.e. neurons. For example, redundant degrees of freedom of elements could produce only chaos and instability in system work in the case when the agent is performing certain sequence of actions using a well-tested plan. Therefore, from this point view we highlight functional systems theory [Ano74] as the most perspective approach that could be described this way.

Anokhin' s opinion is that the interaction between elements, taken by its own, is not able to form a system from a set of these elements, therefore, according to his opinion, the system-formation factor is only useful for the system adaptive result. So, instead of concept "interaction" must be exploited concepts "collaboration" or "co-operation". Additionally, the system should make the problem statement, including the criterion of a solution and the program of actions, before any acting performed by it. In the case of insufficiency of the obtained result the system should stimulate its activating mechanisms, performing active selecting of new components, changing degree of freedom of acting components, and at the end after several "trials and errors" the system should obtain fully sufficient useful result. In such a way, tasks solving is performed through efficient managing of systems components degree of freedom, and in case of success exploited degrees of freedom should be fixed.

Ordering in interacting between elements should be set using their degrees of contribution in collaboration performed in order to obtain by the system a preprogrammed useful result. If a degree of freedom does not help to obtain the useful result, it should be eliminated from the use. The system should efficiently manage degree of freedom of its elements, and simultaneously only a small subset of all elements could be active.

The described conflict between principal possibilities of self-organizing and stability in system functioning could be solved as follows: hardness in internal problem statement by a system – the guarantee of its stability, but flexibility in problem solution obtained through testing different degrees of freedom – guarantee of principally unlimited possibilities in adaptation and self-organizing. Therefore, we propose to use functional systems theory as framework for building an autonomous adaptive agent.

Internal Tasks

Functional systems theory considers a useful preprogrammed result as a single system-forming factor. It means that self-organizing can occur only through tasks solving. So, a question arises "how to produce sufficient amount of tasks?" because achieving the global goal defined by external environment could not occur sufficiently often, especially, in the beginning of learning. For example, searching for a certain object could take increasable much time if the agent exploits random walk strategy [Whit91]. Besides, usually the external environment provides the agent with a very small set of hard-to-reach goals that need the completion of self-organizing. Therefore, a set of externally defined goals could not server as a sufficient source tasks for self-organizing, so, the agent needs to exploit an additional internally defined set of tasks as a sufficient source for self-organizing. Solving internal tasks the system can obtain skills sufficient for achieving the global externally defined goal [Pch2003a].

We propose three main sources for internal tasks:

- Faults identification and correcting;
- Achieving hard-to-reach states;
- Memorizing the environment behavior.

Faults identification and correcting: Faults identifying and then correcting is an important task for an intelligent autonomous self-organizing system. During learning, the agent has autonomously to make many decisions in the lack of human control, and many of these decisions, producing a serious impact on self-organizing, can appear to be erroneous. Faults and breakdowns may occur very often in such self-organizing system, so, the system components should have ability to "feel", identify and correct the place of a fault. Inductive automaton is a good example of such property implementation [Yem90].

In inductive automaton each neuron has not two as usual, but thee stable states: inactive, half active and fully active. The intermediate half active state could be considered as an indicator of a breakdown of a neuron assemblies functioning consistency [Pch2003c]. Therefore, correcting performed by a system of its own structure should be considered as a normal response produced by the system against the appeared breakdown. However, this response, according to Anokhin's theory, should emerge not as an automatic predefined reaction, but only as an automatic problem statement. Inductive automaton has a homeostatic architecture, and, unfortunately, it has a huge list of automatic internal reactions, employing for self-regulation, that take place in pre-programmed way, and the system has no ability to review their utility.

Besides of examples in AI, it is possible to find the confirmation of faults identification relevancy in distant examples, e.g. modern programming languages usually have specially defined abstract class – an exception for faults and breakdowns processing. Thereby, a fault situation is made normal, because of specially designed mechanism to handle it, and the system keeps ability to function normally also in case of faults and breakdowns.

Achieving hard-to-reach states: Usually to achieve the global external goal, the agent must have a set of skills for it. How to obtain these skills before reaching the global goal? – It is possible to employ such a heuristic: *skills and knowledges, needed to reach the global goal, could be obtained through reaching usual hard-to-reach states of the environment*. Really, the goal state is the same state as all other states, therefore, the agent can use any states to obtain needed skills. For example, in the paper [Pch2003a] the agent through training in reaching of local goals, i.e. usual hard-to-reach states temporally defined as internal goals, obtains knowledge that helps to achieve the global goal.

In real situation the agent doesn't have direct access to a global state of the environment, instead, it has only partial information about it. Besides, the agent can form its model of the environment in order to identify its current position in it. Therefore, all recognizable by the agent states could be represented by its internal states, e.g. a set of currently activated neurons. So, reaching of an internal state could be defined as a reaching of certain states of its components.

Therefore, it is proposed to employ the task of achieving hard-to-reach states for self-organizing. For example, the agent can try to make active certain group of neurons by performing different sequences of actions in the environment. The process of solving such tasks could be considered as an exploration activity in the environment performed in order to gain new knowledge.

Memorizing the environment behavior: The third proposed kind of tasks is memorizing behavior of the environment, i.e. memorizing observed sequences of events in the external environment through formation of corresponding skills that help to reproduce them internally.

In the first and second parts there were analyzed a series of learning algorithms, modeling different aspects of intelligence. Each of them employs the concrete to obtain the abstract. Such functioning could be interpreted as iterative process, where through each iteration the learning algorithm "tries to memorize" the current portion of the concrete, integrating it into the abstract. For shortness, we will refer this process "memorizing".

Practically all the artificial neural networks are constructed for tabular static data processing, e.g. a set of vectors. However, in natural neural network, the memorization unit is not a vector, but it is a dynamical stereotype [Pavl49]. Therefore, we also make an accent on memorizing of logical sequence of observed events, as an elementary unit of the environment behavior.

Obviously, each component of the system could be participated in memorizing of a great deal of sequences. Therefore, it is possible to obtain a conflict situation when new information is not integrating with the old one, damaging memory about it. Moreover, mechanically recorded new information may damage the consistency of the information stored in system components. Therefore, memorizing of information must be focused not on

recording of new information, but on searching for a way to generate this information by the system. The system of component should to "born" needed dynamical stereotype.

We propose to solve the task of memorizing of the environment behavior following way. The agent directly records an observing sequence of events into short-term memory. Then the agent states a problem – to reproduce this sequence in the correct order employing its components. To achieve the planned result, the agent should find and setup degrees of freedom of its components sufficient for memorizing of the previously recorded sequence. This process can occur in correspondence to functional systems theory proposed by Anokhin. The agent can stimulate different mechanisms for searching for a sufficient set of components and then applying different degrees of freedom for them. Thus, the agent through trials and errors should to reproduce by its components a previously recorded sequence.

To protect consistency of the system of components, it is proposed to following criterion: *the memorization of a dynamical stereotype must be done through minimal increasing of degrees of freedom of components, and, additionally, the agent must test its ability to reproduce a memorized dynamical stereotype*.

Discussion

In this part we will analyze additional problems that must be taken into account building autonomous agent.

Multitasking: The total number of tasks could be very large. Therefore, these tasks solution may conflict between each other. So, there is a need for mechanism that selects the most important tasks from the set of all tasks. This mechanism must choose one task, control its solving time and compare obtained result with the planned one. In the case of failure, it must manage degrees of freedom of components or choose another task. Functional systems theory [Ano74] proposes a principle of dominating of an active functional system under alternative systems, so, this theory proposes an adequate solution.

Physical and logical layers: Many approaches, *e.g.* [Amo79, Wick99], employ an assumption that a simple relation between physical and logical layer could be employed. In contrast, our position is that the physical and logical layers must be principally separated. For example, the man is able to operate simultaneously with 5-7 objects. However, much more thousands of neurons could be activated simultaneously. It means that the brain work at much more primitive level, *i.e.* complex behavior is obtained not thought complex semantic meaning of components, but through the huge number of possible way of collaboration in the set of simple elements.

Elements must have a primitive problem-oriented sense, but at the same time they can have advanced system forming semantic. For example, neurons in inductive automaton have additional intermediate half-active state that makes a neuron able to recognize its faults and inconsistent work. It is necessary to focus not on problem domain oriented semantic meaning but on calculating abilities, *i.e.* must have a principal possibility (there must be a principal possibility to obtain it during self-organizing) to generate practically any behavior.

Formation of abstraction: As it was mentioned, preprogrammed types of abstraction must not be embedded in the architecture of the agent, because it may limit freedom and principal of system self-organizing. However, how to obtain different abstractions, if they are not embedded into the system? We propose to formulate the main reason as follows: *the formation of different types of abstraction must be obtained as a by-product of dynamical stereotype memorizing by distributed architecture*.

Solving different tasks, a system will learn to reproduce the huge number of dynamical stereotypes, and each of components could be reused in many stereotypes. It means that the function of a component will be abstract regarding the stereotype. So, it is a reason for abstraction formation. Growth of the logical layer, consisting of different stereotypes, based on fixed physical layer, *i.e.* a fixed set of components, should mean the formation of more and more abstract knowledge representations. This principle applied to [Yem90] has been analyzed in [Pch2003b].

Problem of duplicates: The problem of information duplicates is important enough. Real self-organizing has a serious drawback – in different places of the component system similar substructure may be formed. Additionally, formed duplicates are able to speed up explosion of duplicates formation. Interesting, that the same phenomena could be observed in far areas, e.g. large software products developing, and the development of abstract classes and methods is a widely employed approach against duplicates in the code of programs. Therefore, it is important answer a question: how the similar parts will be able to find each other? We distinguish to base approaches:

- *Simultaneous counteraction.* For example, in Kohonen feature map [Fau94] only one neuron could be activated, or fixed-size group of neurons could be activated in associative-projective neural networks [Kus2000] and M-networks [Amo73]. Limiting a number of simultaneously active neurons, it is possible to stanch the growth of duplicates, because the neurons of duplicated substructures must be activated simultaneously.
- *Delayed counteraction.* To our mind, the most efficient method against duplicates is a formation of neural assemblies. The neurons in duplicated substructures with analogous functions must be activated simultaneously, so, applying Hebb's learning rule [Fau94], corresponding neurons must be joined into assemblies. These ideas could be find in inductive automaton [Yem90] that has been analyzed in paper [Pch2003c].

Conclusion

Basing on this investigation as the most perspective approaches have been selected works [Ano74, Yem90], and the discussed ideas could be summarized as a series of following principle:

- The architecture of the autonomous adaptive agent must be based on functional systems theory [Ano74]. Before any action the agent must state the task and choose a program of actions, and the system must have ability to identify an erroneous decision in the case of fault.
- The single self-organizing and system-forming factor is obtaining of useful preprogrammed result, *i.e.* a solution of some problem. For this purpose, it has been proposed to employ following sources of tasks: faults identification and correcting, achieving hard-to-reach states and memorizing the environment behavior.
- Preprogrammed types of abstraction must not be embedded into the agent architecture. However, the composition of components must have a principal ability to be sufficient for any specific behavior implementation. Abstraction formation must be derived as a by-product of memorizing of huge number of dynamical stereotypes by distributed architecture. Logical layer growth based on fixed-size physical layer must imply the formation of more and more abstract forms of knowledge representation.
- The principal ability to efficiently process hypotheses must be integrated into the architecture. The abstract model usually is sufficient for decision-making, but in the case of fault it is necessary to have ability for backing to concrete original information in order to identify an erroneous hypothesis and for making correction in the abstract knowledge representation. In the most perspective way this idea is implemented inductive automaton [Yem90].

To conclude, it is necessary to remember works occurring in the similar direction, that have made an impact on the current investigation, for example the paper [Ku2003].

Acknowledgement. This research was supported in part by the Latvian Science Foundation under grant No.02-86d.

Bibliography

[Shannon, 1949] C.E.Shannon. The Mathematical theory of communication. In: The Mathematical Theory of Communication. Ed. C.E.Shannon and W.Weaver. University of Illinois Press, Urbana, 1949.

[Amo73] Amosov, N.M., Kasatkin, A.M., Kasatkina, L.M., Talayev, S.A. Automata and the mindfull behaviour. Kiev: Naukova Dumka, 1973. (in Russian)

[Amo79] Amosov, N. M. Algorithms of the Mind. Kiev: Naukova Dumka, 1979. (in Russian)

[Ang87] Angluin D. Learning regular sets from queries and counterexamples, Information and Computation, 1987. Nr. 75, pp.87.-106.

[Ano73] Анохин П.К. Принципиальные вопросы общей теории функциональных систем. Принципы системной организации функций. – М.: Наука, 1973.

[Ano74] Анохин П.К. "Системный анализ интегративной деятельности нейрона.", «Успехи физиол. наук», 1974, т.5, №2, с.5-92,.

[Bong75] Бонгард М.М., Лосев И.С., Смирнов М.С. Проект модели организации поведения – Животное // Моделирование обучения и поведения. – М.: Наука, 1975.

[Chr92] Chrisman, L. Reinforcement Learning with Perceptual Aliasing: The Perceptual Distinctions Approach, In Tenth National Conference on Artificial Intelligence, 1992, pp.183-188.

[CR94] Cliff D., Ross S. Adding temporary memory to ZCS. Adaptive Behavior, Nr. 3(2), 1994, pp.101-150.

[Fau94] Fausett, L. V. Fundamentals of neural networks: architectures, algorithms and applications, Prentice-Hall, Inc. New Jersey, 1994, 461 p.

[Hol86] Holland J.H., Escaping Brittleness: the Possibilities of General-Purpose Learning Algorithms applied to Parallel Rule-Based Systems, In: Machine Learning, an Artificial Intellegence Approach, Nr. II., 1986.

[Hunt75] Hunt E.B.: Artificial intelligence, Academic Press, Inc., NewYork, 1975, p.558.

[KLM96] Kaelbling, L.P., Littman, L.M., Moore, A.W. Reinforcement learning: a survey, Journal of Artificial Intelligence Research, Vol. 4, 1996, pp.237-285.

[Ku2003] Kursin, A., Neural Network: Input Anticipation May Lead To Advanced Adaptation Properties, In: Kaynak, O. et al. (eds.): Artificial Neural Networks and Neural Information Processing. Springer-Verlag, 2003, pp.779-785.

[Kus2000] Rachkovskij, D.A., Kussul, E.M. Building large-scale hierarchical models of the world with binary sparse distributed representations, 2000.

[Maes95] Maes P., Modeling Adaptive Autonomous Agents, Artificial Life: An overview, MIT Press, 1995.

[McCallum95] McCallum, A., Reinforcement learning with selective perception and hidden state (Ph.D. dissertation). Department of Computer Science, University of Rochester, Rochester, NY, 1995.

[Mit99] Mitchel, T.H., Machine learning, The McGraw-Hill Companies. Inc., 1999.

[Pavl49] Павлов И.П. Полное собрание трудов. Т. 3. М..Л., 1949, с. 496.499.

[Pch2003a] Pchelkin A. Efficient exploration in reinforcement learning based on Utile Suffix Memory, In: Informatica, Lithuanian Academy of Sciences, Vol.14., 2003.

[Pch2003b] Pchelkin A. Self-organizing in neural networks based on memorizing, In Scientific Proceedings of Riga Technical University: Computer Science, Information Technology and Management Science, RTU, Riga, Vol. 5.

[Pch2003c] Pchelkin A. Inductive automaton: optimal spike frequency, In Scientific Proceedings of Riga Technical University: Computer Science, Information Technology and Management Science, RTU, Riga, Vol. 5.

[Ron94] Ron, D., Singer, Y., Tishby, N. Learning probabilistic automata with variable memory length, In Proceedings of the Seventh Annual ACM Conference on Computational Learning Theory, 1994, pp.35-46.

[Rab89] Rabiner, Lawrence R. 1989. A tutorial on hidden markov models and selected applications in speech recognition. Proceedings of the IEEE 77(2).

[Ryz77] Ryzin J.V.: Classification and clustering, Academic Press, Inc., NewYork, 1977. p.389.

[RS93] Rivest R. L., Schapire R. E. Inference of finite automata using homing sequences, Information and Computation, Nr.103(2), 1993, pp.299.-347.

[She95] Shen W.-M. Learning finite automata using local distinguishing experiments. Computational Learning Theory and Natural Learning Systems, MIT Press, Nr. III: Selecting Good Models, 1995.

[Sut98] Sutton, R.S., Barto, A.G. Reinforcement learning: An introduction. Cambridge, MA: MIT Press, 1998.

[Whit91] Whitehead, S.D. Complexity and cooperation in Q-learning, In Proceedings of the Eighth International Workshop on Machine Learning, 1991, pp.363-367.

[Wick99] Wickelgren, W.A.: Webs, Cell Assemblies, and Chunking in Neural Nets. In Canadian Journal of Experimental psychology. Vol. 53. 1, 1999, pp.118-131.

[Wil85] Wilson S. W. Knowledge growth in an artificial animal, Proceeding of the First International Conference on Genetic Algoritms and Their Applications. Hillsdale, New Jersey: Lawrence Erlbaum Associates, 1985, pp.16-23.

[Wil95] Wilson S. W. Classifier fitness based on accuracy, Evolutionary Computation, 3(2), 1995, pp.149.-175.

[Yem90] Yemelianov-Yaroslavsky L.B. Intelligent quasi-biological system. Inductive automaton. Nauka, Moscow, 1990, 112p. (in Russian)

Author Information

Arthur Pchelkin – Ph.d. student; Decision Support Systems Group, Institute of Information Technology, Technical University of Riga; 1, Kalkyu St., Riga LV-1658, Latvia; e-mail: arturp@inbox.lv