New Way How to Create an Autonomous Creature

Pavel Nahodil and Jaroslav Vítků

Department of Cybernetics, FEE, CTU in Prague
Technicka 2, 16627, Prague 6, Czech Republic
{pavel.nahodil, vitkujar}@fel.cvut.cz

Abstract. We introduce new approaches for creating of autonomous agents. The life of agent created by us is very similar to the animal's life in the Nature, which learns autonomously from the simple tasks towards the more complex ones and is inspired in AI, Biology and Ethology. We present our proved design of artificial creature, capable of learning from experience in order to fulfil more complex tasks, which is based mainly on ethology. It integrates several types of action selection mechanisms and learning into one system. The main advantages are in its autonomy, ability to gain all information from the environment and decomposition of the decision space into the hierarchy of abstract actions, which dramatically reduces the total size of decision space. The agent learns how to exploit the environment continuously, where the learning of new abilities is driven by his physiology, autonomously created intentions, planner and neural network.

Keywords: agent, creature, behaviour, architecture, intentions, planning.

1 Introduction

One of the ultimate goals of nowadays research in the field of technology, to create an autonomous robotic system capable of flawless operation in the real environment full of disturbances and unpredicted events, still has not reached. The complexity and observability of the real environment is simply too high to be exactly described by anything less complex than the environment itself. As the given problem cannot be precisely described by the mathematics, some other approaches which are working with uncertainty and incomplete models must be used. We think that the best chance of fulfilling the given aim is in constructing something relatively simple, but capable of autonomous learning and using this new knowledge to improve itself. This paper tries to describe the possible method of solving this problem by presenting a new architecture of autonomous agent partially capable of things described above, called "An Artificial Creature Capable of Learning from Experience in Order to Fulfil More Complex Tasks" [8].

The main feature of this architecture is in its total autonomy, ability to gain all information from the surrounding environment and effective information filtering and classification. Agent can operate based only on the sensory input and by its actuator system, so the resulting architecture is almost fully independent on the concrete area and form of use. Consequently it is unimportant whether the agent is embodied in some robotic system, intelligent house, or just operates in some virtual environment. Thanks to the fact that all the designer has to specify is just the sensory layer, actuator layer and agent's needs, the architecture should be convenient especially in unknown environments, where some complex task has to be fulfilled.

Agent architecture is inspired by the layered model, combining various approaches on different layers. The life of agent is similar to a newly born animal, which explores new and unknown environment, learns from experiences and links the newly learned abilities to its needs in order to survive and increase effectiveness of its behaviour. New knowledge is learned simultaneously on various levels of abstraction using different learning approaches.

One of the main features is an alternative implementation of system similar to reactive and hierarchical planning. The system combines hierarchical reinforcement learning and planning engine into a domain independent hierarchical planner.

2 Main Parts of the Control System

The architecture is an attempt to create an agent capable of autonomous intelligent operation in nearly so complex environments as is the real world. The agent could be used in the real or virtual environments where the designer has no a priori knowledge about its fundamental regularities and is able to specify only the agent's needs and some high-level goals. The agent autonomously explores the environment and tries to "understand" the principles from simple to the complex ones in order to gain the ability to survive and provide various services to the user.

The single kind of problem representation or approach to solving the problem is almost never sufficient. More than just exploring the capabilities of one concrete kind of decision making, we have decided to use different approach for creating this agent. Each action made by living animals is consequence of superposition of many different motivations, needs, emotions, intensions etc. We simulate this by connecting several of decision and control blocks. Each block consists of some kind of well-known and widely used system, such as for example neural network or planner. We want these blocks to be interconnected in such manner, that some kind of intelligent behaviour emerges from their interaction. The resulting system also suppresses the weaknesses of particular subsystems and exploits their benefits more efficiently. Three main subsystems used in this architecture are from the bottom: artificial neural network, reinforcement learning and planning. We will describe their purpose in more details here.

The core of entire architecture is created by the block providing the hierarchical reinforcement learning. This implementation of *Reinforcement Learning* (RL) is mainly inspired by the Dissertation Thesis [1], where Kadleček presented his idea called *"Hierarchy, Abstraction, Reinforcements, Motivations Agent Architecture"* (**HARM**) [2],[3]. It is the hierarchical reinforcement learning with autonomous creation of hierarchy based on the interaction with the surrounding environment. According to the HARM, the agent has some physiological state space described by the dynamic system; this space contains several variables, where each variable represents some of the agent's needs.

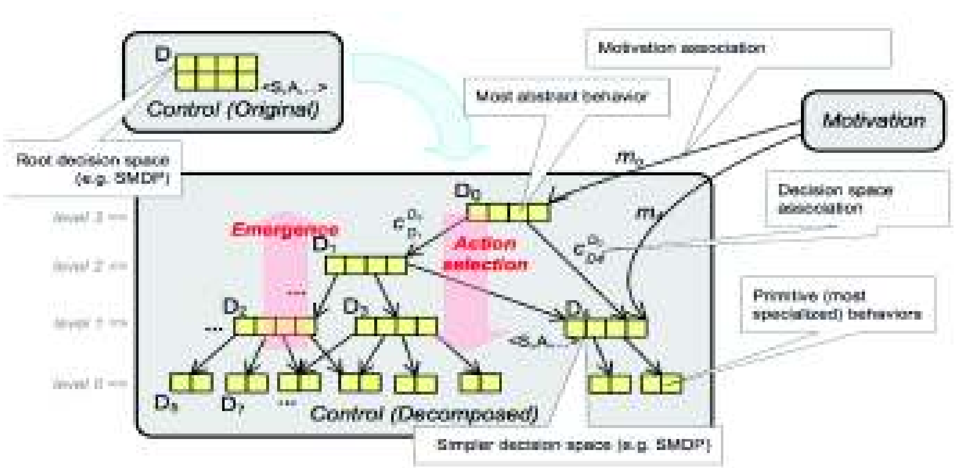

Fig. 1. Hierarchically decomposed decision space by the HARM system. The root node corresponds to the most abstract behaviors, whereas the leaf nodes correspond to the lowest behaviors-primitive actions.

At the beginning of the simulation, the hierarchy of abstract actions is empty the agent acts randomly and observes whether he caused some active change of some of his physiological variables towards the limbo area. If so, the new abstract action in the hierarchy is created and connected to the motivation. The abstract action corresponds to the new decision space used by the RL engine. Each decision space contains some subset of environment variables and agent's actions, which is defined online during the agent's life. This system dramatically reduces the total size of search space used by the RL engine. The Q-Learning algorithm is used in each decision space provides online trade-off between the exploration and knowledge exploitation. The resulting primitive action executed by the agent is then composition of decisions made by all RL engines created so far. The main benefit of this system is the fact that the HARM is capable of autonomous creation of action hierarchy based on the interaction with surrounding environment; this reduces the decision space that have to be considered.

This system was further improved for online continuous learning and augmented with the **Intentional State Space**, it has the similar purpose as the physiological state space, and the difference is that agent's intentions are created autonomously during his life. If the agent discovers that is able to actively change some environment variable (e.g. turn on the lights), the new intentional variable is created. Intentional state variables have predefined its own dynamics and motivate the agent to learn and "train" this newly discovered behaviour. Because of this part, the agent can autonomously discover new potentialities of the environment and learns how to exploit it, this new knowledge can be exploited later. This approach corresponds to learning of newly born animal by play.

The other main subsystem of the architecture is the hierarchical **Planning Engine**. It is composed of classical "flat" planner operating over the hierarchy of decision spaces. One of the most important ideas of this architecture is in the ability to represent the abstract action from the RL action hierarchy as a set of primitive actions in the *Stanford Research Institute Problem Solver* (STRIPS) language [4]. This gives the agent ability to deliberatively "think" about the actions previously learned during the interaction with the environment and to fulfil complex tasks. The advantages of hierarchical decomposition of plans are already well known, and referred for example in [5] as *Abstraction Space Hierarchy from STRIPS* (ABSTRIPS) or *Hierarchical-Task Network* (HTN) [6]. Our planner can operate over the abstract actions on arbitrary level of action abstraction. The main benefit of this solution is in the fact that the complicated outer world is pre-processed by the RL-action hierarchy, so this hierarchical planner can operate in very complex domains, while still maintaining its domain independence, which is advantage against the well-known hierarchical planners. More precisely, our planner could be referred as domain self-configurable.

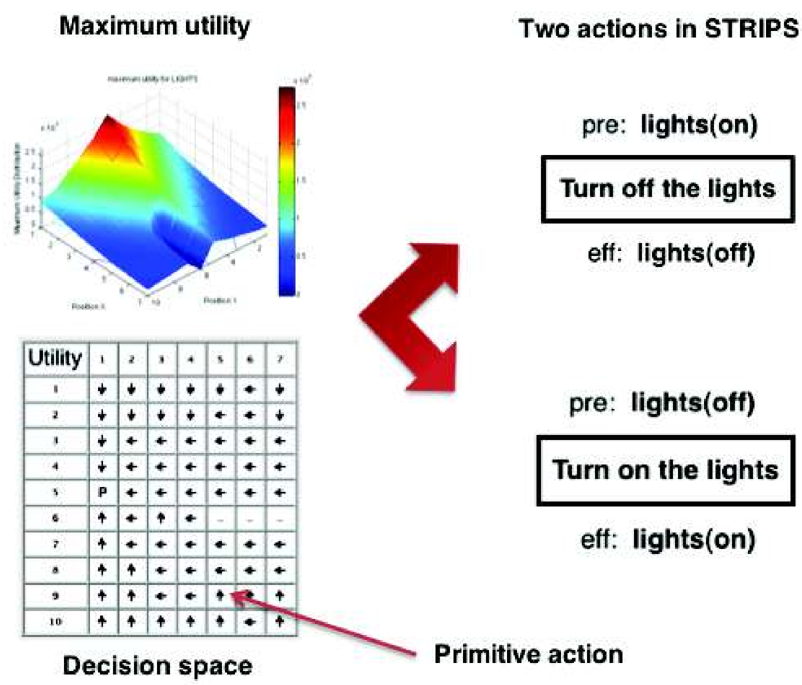

Fig. 2. Example of representing the decision space (abstract RL action) as two primitive actions in the format of STRIPS language.

The HARM action hierarchy serves as some "interface" between planning engine and outer world, this interface is created autonomously and adapts to the given problem online during the agents life.

The last part of architecture serves for learning of reflexive behaviour and gives the agent ability to react in selected situations with necessary speed of response. This subsystem is implemented by **Artificial Neural Network** and learns patterns situation-action. The *Artificial Neural Network* (ANN) learns appropriate behaviour from the HARM system. When the situation is considered as critical, the HARM system generates adequate action and the ANN learns this one pattern "actual situation"-"generated action". After some time, when the ANN shows learning error small enough, the agent can act reflexively. From this moment, the ANN can take the control over the agent and generate one primitive action in the critical situations. This system provides small reaction time where it is important and in fact increases the precision of action selection; this is caused by generalization ability of neural network.

3 Selected Experiments and Conclusion

We have conducted numerous experiments in order to test implementation of our architecture and to compare against other known approaches. The main focus was in testing the ability to effectively reduce the size of decision space (Fig.3), to act in dynamic environments (predator-prey simulation) and to test the ability to create and execute plans based on the knowledge gained autonomously.

In the Fig.3 we can see selected experiment, where the agent learned how (and when) to eat and drink. The total size of the decision space was increased by other moving agents (cows) to approximately 64×10^{12} states, the agent successfully learned that the positions of particular cows can be ignored, and thus was able to reduce the number of states to 2×100 states by identifying two important behaviours ("eat" and "drink") and creating corresponding RL decision spaces.

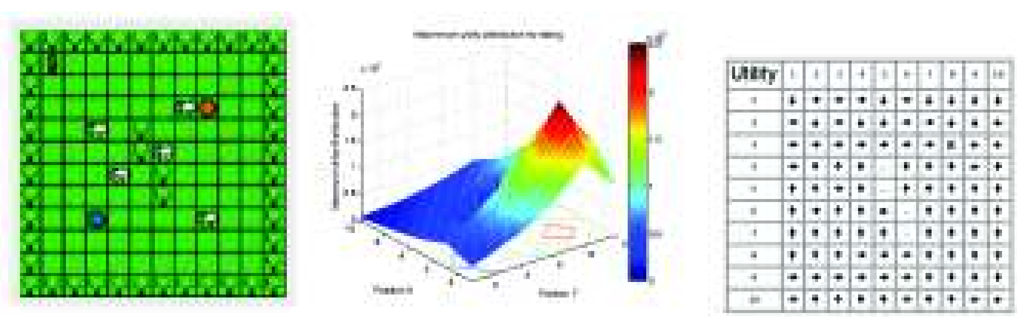

Fig. 3. Experiment testing the reduction of decision space size. The map (left) contains food (brown object) and water (blue) sources and the size of the decision space was increased by other moving agents (white cows). The goal is to learn how to survive - how to maintain the water and food levels in agent's body in bounds. During his life, the agent was able to successfully decompose his behaviour into two abstract actions, "eat" and "drink". For the "eat" behaviour, we can see the graph of maximum utility (and the table describing the best action) corresponding to the agents $\langle X, Y \rangle$ position, which means dramatic reduction of decision state space size compared to the flat RL approach.

None of the mentioned approaches could handle the all of the comparable problems alone, so the main contribution of this architecture is in combination of the advantages of particular subsystems and in their interconnection in such manner, that more complex behaviour can emerge from their mutual interaction.

We believe that the main advantage of our architecture is in combination of hierarchical reinforcement learning subsystem and the planning engine in such manner, so their collaboration provides ability of hierarchical planning while still maintaining the domain independency. This is because the fact that architecture configures itself based only on the knowledge autonomously gained from the particular domain. This means that, compared e.g. to the planner in widely used *Belief-Desire-Intention* architecture (BDI) [7], our agent does not need the predefined plan library - that is a knowledge a priory. Recently, some similar architectures, which try to combine planning and reinforcement learning (that is subset of our work) were found [9], [10], where the first one has the similar disadvantage as the BDI architecture mentioned above, and the second one uses slightly different approach. Based on our experiments concluded in order to compare it with other approaches, we believe that our architecture is superior to several widely-used principles in nowadays field of AI.

References

1. Kadleček, D.: Motivation Driven Reinforcement Learning and Automatic Creation of Behavior Hierarchies, CTU in Prague, FEE, PhD thesis, (2008)
2. Kadleček, D., Nahodil, P.: New Hybrid Architecture in Artificial Life Simulation. In: *Proc. of 6th European Conf. of Artificial Life: Advances in Artificial Life.* ECAL, Prague, LNAI Nr. 2159, vol. 1, Printed by Springer, Berlin, vol.1, 143-146 (2001)
3. Nahodil, P., Kadleček, D.: Adopting Animal Concepts in Hierarchical Reinforcement Learning and Control of Intelligent Agents, In: *Proc. of 2nd IEEE/RAS-EMBS International Conference on Biomedical Robotics and Biomechatronics, BioRob* 2008, Scottsdale, U.S.A, 122-131 (2008)
4. Fikes R., Nilsson N.: STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Artificial Intelligence*, 2, 189-208, (1971)
5. Sacerdoti E. D.: Planning in a Hierarchy of Abstraction Spaces. *Artificial Intelligence* 5(2), 115-135 (1974)
6. Erol, K., Nau, D., Hendler, J.: HTN Planning: Complexity and Expressivity. In *AAAI-94*, Seattle, (1994)
7. Sardina, S., de Silva, L., Padgham, L.: Hierarchical Planning in BDI Agent Programming Languages: a Formal Approach. *AAMAS '06 Proceedings of the fifth*

international joint conference on Autonomous agents and multiagent systems, 1001-1008. ACM New York, USA (2006)

8. Vítků, J.: An Artificial Creature Capable of Learning from Experience in Order to Fulfill More Complex Tasks, CTU in Prague, FEE, Diploma thesis, (2011)
9. Ryan, M., Pendrith, M.: RL-TOPs: An Architecture for Modularity and Re-Use in Reinforcement Learning, In *Proc. of the 15th International Conference on Machine Learning*, 481-487, (1998)
10. Ross, M.: Hierarchical Reinforcement Learning: A Hybrid Approach, PhD Thesis, The UNI of New South Wales, School of Computer Science and Engineering, (2002)