Motivated Learning In Autonomous Systems

Pawel Raif, Student Member, IEEE, and Janusz A. Starzyk, Senior Member, IEEE

***Abstract*— Motivated learning (ML) is a new biologically inspired machine learning method. It is the combination of a reinforcement learning (RL) algorithm and a system that creates hierarchy of goals. The goal creation system is concerned with creating new internal goals, building a hierarchy of them, and controlling the agent's behavior according to this constituted hierarchy of goals. As in case of reinforcement learning method, a motivated learning agent is learning through interaction with the environment. The comparisons of both methods in special type test environment show that the motivated learning method is more efficient in learning complex relations between available resources (concepts). ML has better performance than RL, especially in dynamically changing environments. In the presented experiments we have shown that ML based agent, which has the ability to set its internal goals autonomously, is able to fulfill the designer's goals more effectively than RL based agent. In addition, because the observed concepts are not predefined but emerge during the learning process, this method also addresses problem of merging connectionist and symbolic approaches for intelligent autonomous systems.**

***Index Terms* - autonomous systems, intelligent agents, intrinsic motivation, motivated learning, reinforcement learning, hierarchical problem decomposition.**

I. INTRODUCTION

IN current research of computational reinforcement learning there are two main problems. The first one is the decomposition of complex problems into a hierarchy of simple tasks. To overcome this problem, methods of hierarchical reinforcement learning have been proposed. The second one is an intrinsic motivation mechanism which allows a machine to define its own way of learning in and exploring its environment. In this paper we present the motivated learning system (ML) that combines these two elements. In the proposed system both elements: intrinsic motivation and hierarchical decomposition are inherently linked. Specifically, intrinsic motivation is the mechanism that performs hierarchical decomposition of the problem. The proposed motivated learning system is a combination of a goal creation system (GCS) with reinforcement learning. The goal creation system is concerned with creating new internal goals, creating a hierarchy of such internal goals, and controlling an agent's behavior according to this emerged hierarchy of goals and internal states of the system.

Our aim is to use one of the well-known reinforcement learning algorithms in the proposed learning system to both enhance ML learning capability and to compare with the raw reinforcement learning control in tasks typically used to illustrate the RL approach.

Issues concerning hierarchical reinforcement learning and intrinsic motivation in autonomous systems are discussed in the following sections.

Manuscript received January 25, 2011.

J. A. Starzyk is with the School of Electrical Engineering and Computer Science at Ohio University, Athens, OH, USA (phone: 740-593-1580; fax: 740-593-0007; e-mail: starzykj@gmail.com).

P. Raif is with the Institute of Economics and Computer Science, Silesian University of Technology, Gliwice, Poland (e-mail: pawel.raif@polsl.pl).

II. RELATED WORK

A. Intrinsic Motivation

In typical RL experiments rewards are usually defined by the designer of the machine. An intelligent autonomous system must be able to create and pursue its own internal goals. An intrinsically motivated agent's behaviors are usually aimed at performing its own activities to satisfy its curiosity rather than solving practical problems, but what the agent learns during this activity is useful for solving practical problems as they arise [1].

Intrinsic motivation and autonomous development in autonomous systems has been investigated in psychology, neuroscience and computational intelligence, mainly using a reinforcement learning approach. It is widely believed that intrinsic motivation is integral to the way humans learn and explore their environment [2], [3]. Neuroscience aspects of intrinsic reward systems have been explored in [4] and [5].

A computational approach to intrinsic motivation has been presented in [1]. Authors presented results from their study of an intrinsically motivated learning method based on reinforcement learning. The aim of their method is to allow artificial agents to construct and extend hierarchies of reusable skills. As the authors state, they intend to implement a simple computational analog of intrinsic motivation mechanisms similar to those suggested in the psychological, statistical, and neuroscience literature. Their method is a combination of existing RL algorithms for learning options and option-models with a simple notion of intrinsic reward.

Closely related RL research are the works of Schmidhuber [6], [7], [8] on artificial curiosity in robots for exploratory behavior in unknown environments. His system is based on Watkin's Q-learning algorithm [9]. In [10] the author revisited his former idea in the context of recent results on optimal predictors and optimal RL machines and proposed some variants of the basic principle. In this context he presented such activities like art and creativity as by-products of curiosity generated rewards.

Another approach based on the curiosity principle has been presented in [11], [12]. The authors proposed an intelligent adaptive curiosity (IAC) system, which attempted to direct a robot in continuous, noisy, inhomogeneous, environments, allowing for an autonomous self-organization

of behavior toward increasingly complex behavioral patterns.

Roa, Kruiff and Jacobson explored the concept of curiosity and whether it can be emulated through a combination of active learning and RL using intrinsic and extrinsic rewards [13]. The authors developed their intrinsic motivation system based on Oudeyer's work [11], and then added an extrinsic reward system to guide the robot to its goal. As the authors stated, they didn't use hierarchical mechanisms to abstract simple tasks into complex ones.

B. Hierarchical reinforcement learning

Usually realistic environments are very complex. In the case of reinforcement learning based systems, the computational cost of learning increases significantly with the environmental complexity [14]. This feature of the RL approach, called "the curse of dimensionality", is one of its main disadvantages in applications to real-world problems. Creating hierarchy of subsequent goals is one of the ways to improve the efficiency of RL. This approach, often called, hierarchical reinforcement learning (HRL) tends to exploit the structure of both the environment and the agent's tasks to improve policy learning in complex problems. Among the many approaches to hierarchical RL one can distinguish: Dayan and Hinton's research on feudal reinforcement learning [15], Parr and Russel's study on hierarchical abstract machines (HAM) [16], and development of MAXQ method [17].

Bakker and Schmidhuber [18] proposed a method for hierarchical reinforcement learning based on subgoal discovery and subpolicy specialization. Their algorithm can learn to create both useful subgoals and the corresponding specialized subtask solvers. This approach may yield a complex hierarchy of subgoals, although a large number of parameters and lack of strict convergence guarantees are weak points of this approach.

C. Paper Organization

In this paper we describe a motivated learning (ML) approach to learning in a dynamically changing, complex environment. The idea of ML was recently introduced [19] by the coauthor and was initially explored in simple one-step tasks that could be accomplished without multistep searches. It uses "primitive pain" and "abstract pain" signals to control machine's motivations, goal selection and action. The terms "primitive pain" and "abstract pain" signals are synonyms of all kinds of discomforts like fear, panic, anger, and are similar to the concepts of demands and urges in the Psi theory [20]. A motivated learning machine develops and manages its own motivations creates and selects goals using continuous competition between various levels of pain signals (and possible attention switching signals). In a follow-up article [21] basic neural network structures used to create abstract motivations, higher level goals, and subgoals were presented.

This work extends the application of ML towards tasks that require multistep searches typical for RL applications. Section 3 shows differences between known RL approach and proposed motivated learning (ML) to help understand how this method is related to RL. Section 4 presents a detailed methodology: test environment and internal agent structure. Section 5 presents the obtained results. It shows details of learning processes and comparison of effectiveness of ML and RL approaches. This paper concludes with a summary of proposed approach to machine learning.

III. ML IN COMPARISON TO RL

A typical reinforcement learning task is to generate an action a to be performed depending on the observed state of the environment. After each action, the algorithm is provided with two signals: information s about the current state and reward R associated with this state. This situation is depicted on Fig.1a. It is a typical pattern of interactions between RL agent (Fig1.b.) and its environment, characteristic of the reinforcement learning method. Motivated learning (ML) is a combination of the reinforcement learning method and the goal creation system (GCS). Detailed architecture that implements this combination is depicted on Fig.2. The inner structure of the ML agent is composed of a goal creation module and several reinforcement learning modules.

During the interaction with the environment, the GC module constitutes hierarchy of abstract pain "levels" associated with recognized goals. Every RL module corresponds to each such level, which also means that one module corresponds to one goal. For example if the aim is to learn how to use various environmental resources, then after the learning process, all relationships between resources in the environment will be reflected in the hierarchical structure of the agent's knowledge. During the interaction with the environment the agent learns how to use available resources by learning one value function for each goal. Finding the selected resource becomes the agent's current goal. When the agent wants to learn how to achieve the current goal and, by extension, how to find and obtain a needed resource, the agent has to "switch to" an appropriate goal and use the value function associated with this goal. Then it has to search for its goal on appropriate hierarchy level. Learning how to search, and performing real search of a goal, are accomplished by using a typical reinforcement learning algorithm. We use SARSA algorithm [22] to perform this function.

In order to survive, an agent needs knowledge about all kinds of resources available in the environment. An agent's operation is based on two elements: managing of goals and accomplishing these goals on individual hierarchy levels. In turn, goal management is compound of two tasks:

-- Creation of internal motivation system by discovering new goals and their usefulness to the agent

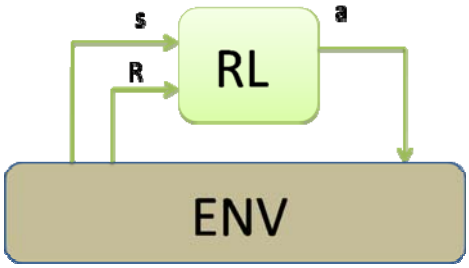
-- Switching agent's activity between discovered goals (levels).

These complex tasks cause that a special type of virtual environment is needed for testing ML agent. In order to show the advantage of using an intrinsic motivation system based on goal creation in comparison to typical reinforcement learning, an appropriate virtual test environment is used as described in the next section. In order to be able to conduct comparative tests of both ML and RL

methods, this test environment is constructed in such way that both kinds of agents may use it. Both agents use the same interface to the environment; they have the same input/output signals, as depicted on Fig.1.b and Fig.2. In both cases the interaction scheme between agent and environment is as illustrated on Fig.1a.

In the case of ML, the reward signal that comes from the environment is replaced with the internal reward signal generated by GC module. Details about GC module of ML agent are described in the next section of this document.

a)



b)

Fig.1.a. Schematic diagram of interaction between RL agent and environment; b. Diagram of input/output signals of RL agent.

Fig.2. Internal structure of motivated learning (ML) agent

IV. Experiments

A. Methodology

This section presents computational experiments performed in an artificial environment with learning agents that use symbolic representations of sensory inputs and motor control. The aim of these experiments was to compare the effectiveness of both methods of learning. We have compared these methods in situations where the task is complex and the agent should learn how to use available resources to accomplish this task. An agent's effort is measured by an aggregated pain signal. In every step of simulation the agent is characterized by certain level of the primitive pain signal that is externally defined and controlled. In an animal, such a primitive pain may be seen as "hunger level" or other basic needs. If the agent learns rules of exploiting and replenishing environmental resources, it can reduce its pain signal by proper action. If it can't learn these rules quickly enough, its pain level will increase because there is lack of some resource (i.e. food) and the agent is not able to replenish them efficiently. Comparison of agents' effectiveness is performed by comparison of their pain signals. The notion of pain is the opposite of the notion of reward, and primitive pain is defined by the designer to control the agent.

B. Environment

We assume that the test environment fulfills several conditions. First of all, it is fully observable and available to both types of agents. Both methods use the same reinforcement learning algorithm in a typical grid-world environment, where the agent moves over a two-dimensional space. Agent's aim is to learn how to find a reward in the smallest possible number of steps. Secondly, we have assumed that the environment will dynamically change and the agent can influence these changes. We have achieved this by implementing a mechanism for exploiting and replenishing natural environmental resources. The environment consists of several kinds of resources arranged all over the grid-world. Resources can be exploited by the agent just as in the natural environment. This feature is not typical for reinforcement learning test environments where the reward is usually at the same place. In this setup if a resource is intensively used it will vanish after some time. However, as in a natural environment, resource can be replenished by the agent's action. Imagine such a situation: if an agent is short of food and is not able to find it in the place where he used to (i.e. in the fridge) it has to learn how to replenish food. It has to find the grocery. If it finds it, it will have access to the food. However, after some time it will run out of money and it will not be able to buy anything. The agent will have to find money before he can shop again, and so on. This kind of mutual dependency of resources can be extended to many levels. This kind of environment is based on a complex, often hierarchical, structure of concepts and resources and ways of using them. In our experimental environment there are four levels of increasingly abstract resources. We assumed that the highest level is inexhaustible.

After learning, an agent's knowledge about the environment (more precisely: about dependencies between environmental resources and his actions) reflects this hierarchical structure and may be presented in several levels of abstraction. The first level of abstraction is connected with searching for a base resource (i.e. food), subsequent levels are connected with more and more abstract resources. So it is that searching for different resources is performed on different levels.

A resource is consumed when an agent reaches this state on the grid, where the specified resource is located. In order to meet the assumption about dynamically changing environment the availability of resources has to be changing over time. Controlling availability of the resources is based on changing probability of finding the resource. An agent

can use the resource for a limited number of times without any consequences. Then the probability of finding the resource is decreasing. The following function describes probability of finding resources:

$$p(x)=\begin{cases} 1 & x \le x_{gr} \\ e^{-\left(\frac{x-x_{gr}}{\tau}\right)} & x > x_{gr} \end{cases} \quad (1)$$

where:
τ – scaling factor that describes a resource declining rate
x – number of times a resource was used
x_{gr} – threshold limit

We have assumed that the agent should be provided with simple information about the availability of a specified resource: 'resource available' or 'resource unavailable', meaning '1' or '0'. This is regulated by a number drawn from a range $\langle 0,1 \rangle$ of uniform distribution according to the value of p(x). The final effect is that at the beginning the agent is able to find the chosen resource (and receive the reward or reduce pain while it reaches the goal location on the grid), but later, more and more often, resource is not there and the agent will not receive his reward.

Replenishing (filling up) of the resource is the consequence of using a more abstract one (e.g. spending money to buy food). Then, probability of getting the reward at the goal location is reset to 1 (with x reset to 0 in (1)) thus a pain can be reduced. Resources are distributed all over 25 x 25 units grid-world. Meaning, there are 625 different states. The distribution of resources on the grid is depicted on figure Fig.3.

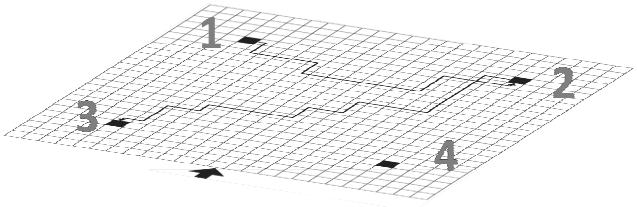


Fig.3. Grid-world type test environment: 25 x 25 grid and four kinds of resources distributed all over it. Sample route 1-2-3.

Fig. 3 also shows a sample route that agent travels to manage its resources after learning.

All limitations in environment size and number of resources in it are consequences of computational time required to perform the test.

C. Agent

Internal structures of ML and RL agents are depicted in Fig.1.b and Fig.2. There are differences in the agents' structures, but the way they interact with environment is the same (Fig.1a): they read environment state *s*, undertake action *a* and check if they get any reward *R*. In the case of both agents we have been using SARSA as the reinforcement learning algorithm. We haven't been using neural networks as value function approximators, only a look-up table.

An agent is able to move over the two dimensional surface of the grid-world. The set of possible actions is defined as follows:

$$A = \{right, left, up, down\} \quad (2)$$

In every step of simulation the agent reads the state value, performs an action and then checks if there is any reward.

Motivated learning based agent. As it was depicted on Fig.2. the agent based on the motivated learning paradigm consists of a few elements: several reinforcement learning modules and one goal creation module. The function of the goal creation module is to constitute the hierarchy of new goals and to manage them (and consequently to manage the agent's behavior according to these goals). Each of the internal goals corresponds to specific reinforcement learning module.

Managing internal goals requires switching the agent's activity between different goals (different abstraction levels) based on their importance. This process is organized as follows. The Goal creation module observes internal pain signals levels. The primitive pain, (which could be interpreted as hunger), is increased by a small constant value every time step of simulation. Higher level (more abstract) pain signals appear only when the agent discovered higher level resources (goals) and has learned their significance. When the agent gets to know the importance of specified resources it gains the ability to estimate the danger of their shortage. The lack of specific resource increases corresponding abstract pain signal. Switching the agent's activity between different goals is based on a WTA mechanism. The dominant pain signal determines the current goal.

The process of creating new goals is also carried out by GC module. The creation of new goals is based on observation of known resources (more precisely: by observing their consumption counters). Through observing resource consumption counters the agent can determine the states where resource counters are reset. Resetting means that the corresponding resource has been replenished. If the replenished resource was the highest one in the hierarchy of resources, it means that some new, yet unknown, higher level resource has been found. It becomes one of agent's internal goals and can be used in the future in order to replenish a lower level resource.

The task of learning how to find a specified goal on the appropriate level is realized by the reinforcement learning algorithm.

Epochs and Episodes. In the case of the reinforcement learning method the process of learning proceeds in repeated cycles called epochs. An epoch covers the period from the start of searching to the moment of finding a reward. It consists of a sequence of simulation steps; an action is performed at every step. We did not limit the maximum number of steps in one epoch. Because the amounts of resources are gradually decreasing, the agent should be provided with enough time to find the solution to this

problem. This is regulated by the scaling factor τ and the threshold limit x_{gr}.

In the case of the motivated learning method the process of learning consists of episodes to be distinguished from epochs in RL. An episode is a period of time between two subsequent switching from one goal to another. It is a time dedicated to current goal. An episode may be aborted when another goal has higher priority (higher pain signal) switching the agent to another episode. However, while an episode also covers the period from the start of searching to the moment of finding a reward there is the difference: in the case of motivated learning, the reward may be intrinsic. This means that an episode may be finished even when the primitive (base) reward hasn't been found. Thus the episode may be finished when one of the internal rewards has been found or in situations when another pain signal starts to dominate the agent's behavior. Tasks carried out during an episode are presented in pseudo code in Fig.4.

```
Repeat until episode_continues:
  1. Initialize SARSA algorithm
  2. Perform action in the environment
  3. Is there change in R (base reward) or G
     (internal reward)
     (if yes then reward = True)
  4. Identify dominant pain signal (winnerP)
  5. IF dominant pain:
     is different than the current pain signal AND
     it is known THEN finish the current episode
```

Fig.4. Pseudo code of single episode (ML).

V. RESULTS

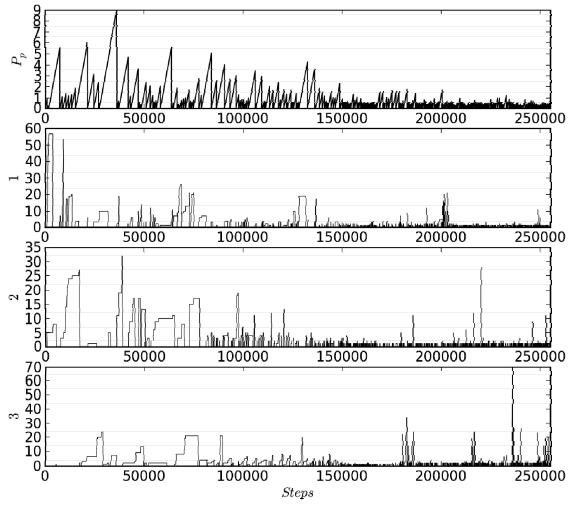
The presented experiments consist of two subsections. In the first one, the characteristics of the learning processes of both kinds of agents has been described. In the second subsection a comparison of the effectiveness of both methods in the test environment has been presented.

A. ML Agent Learning Process

Effects of interaction between the motivated learning agent and the environment are presented visually in the figures shown below. These results are obtained in a single computational experiment, meaning a single simulation. In Fig.5a. four signals are illustrated: primitive pain signal P_p and three resource consumption counters (which correspond to higher level abstract pain signals). Because the highest level resource is inexhaustible, it is not depicted. The higher the consumption of specified resource, the higher the corresponding pain is. The reduction of a lower level pain occurs only as a result of using a higher level resource. For example: reduction of primitive pain signal comes only after using the resource "food".

The simulation shown consists of 1200 episodes, with over 250 thousand simulation steps. In every single step the agent performs a chosen action. Each episode is usually of different length.

a)



b)

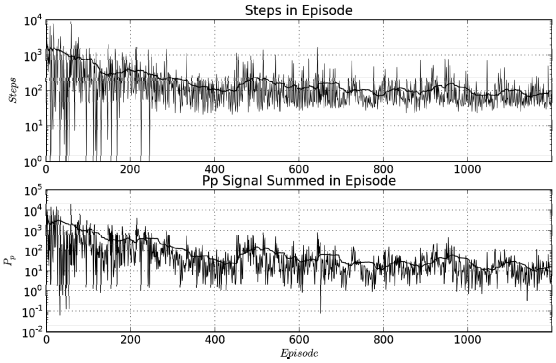


Fig.5. Motivated learning agent learning process: a) primitive pain signal Pp and resource consumption counters, b) simulation steps and primitive pain aggregated by episodes.

At the beginning of the simulation, when the agent is still intensively learning value functions, episodes are usually longer. They are getting shorter with the progress of learning (as time goes on). This situation is depicted in Fig.5b., where total number of simulation steps is divided into episodes. Upper part of this figure shows the number of simulation steps in subsequent episodes. We can observe that after about 350 episodes the average number of steps per episode stabilizes. The number of steps needed to find the reward is correlated with primitive pain signal P_p. The lower part of this figure shows that the primitive pain signal is also stabilizing after about 350 initial episodes. After that it is stable. It is because finding of the reward takes almost a constant number of steps in every episode. This means that the agent has learned all the value functions needed to find all kinds of resources (to accomplish all kinds of goals, including internal).

The agent has to be able to use all kinds of environmental resources in order to manage its pain signals. In order to learn how to use all of the resources, the ML agent acts in

such a way, that for every new (known) resource the state where it is located becomes new internal goal. So, after the learning process every known environmental resource corresponds with one of agent's internal goals. As each new goal appears, the goal creation module creates a new value function. In this experiment, there are four possible goals (resource locations) to learn. The internal motivation system based on goal creation and controlled by pain signals is switching the agent's activity between these goals. We can get more information about this goal switching process by observing the switching diagram depicted in Figures 6.a and 6.b. In Fig.6.a. we can see the early stages of simulation – the first 100 thousands steps. In Fig.6.b we can see the last 10 thousand simulation steps.

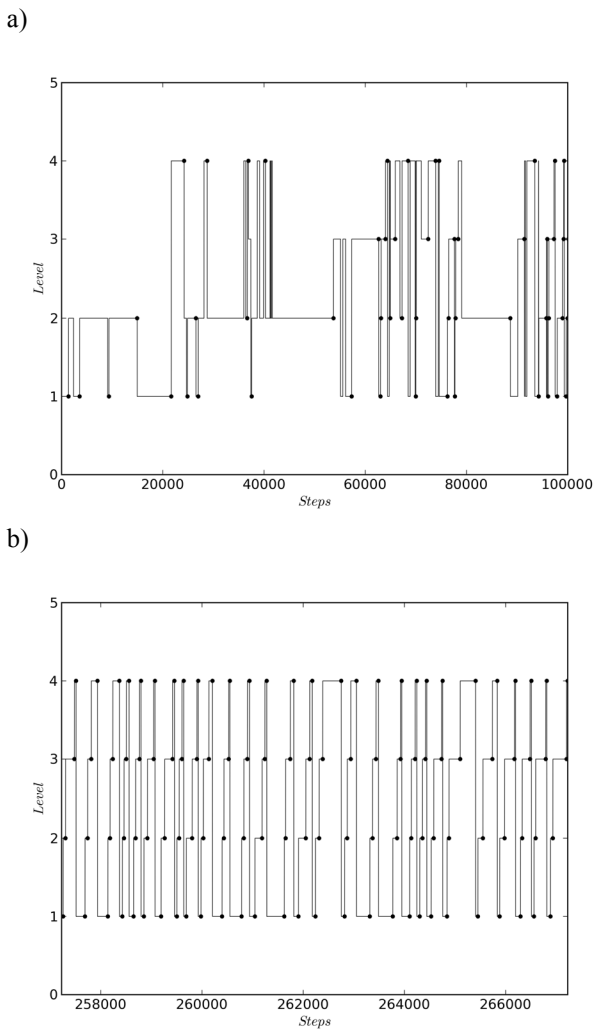


Fig.6. Switching of ML agent's activity between several goals. Dots at the end of selected episodes indicate success; a) the first 100 thousands of steps, b) the last 10 thousands of simulation steps

Switching to another goal (another level) is depicted as a vertical line. The single episode - pursuit of specific goal, is depicted as a horizontal line. The bigger dot located at the end of horizontal line means that pursued goal was found. Comparison of figure 6.a. and 6.b. shows that, as the effect of the learning process, subsequent episodes has been shortened (notice different scales on Fig. 6.a and Fig. 6.b). That means that the agent has learned all the value functions corresponding with all necessary resources during the interaction with test environment. These value functions are shown in Fig.7.

As a result of the learning process, switching between goals has been changed from chaotic in the beginning (Fig.6a.) to organized (Fig.6.b.) at the end of simulation.

Subsequent levels created during the learning process are connected in such a manner that reflects the hierarchical structure of environmental resources, which has been discovered by the agent.

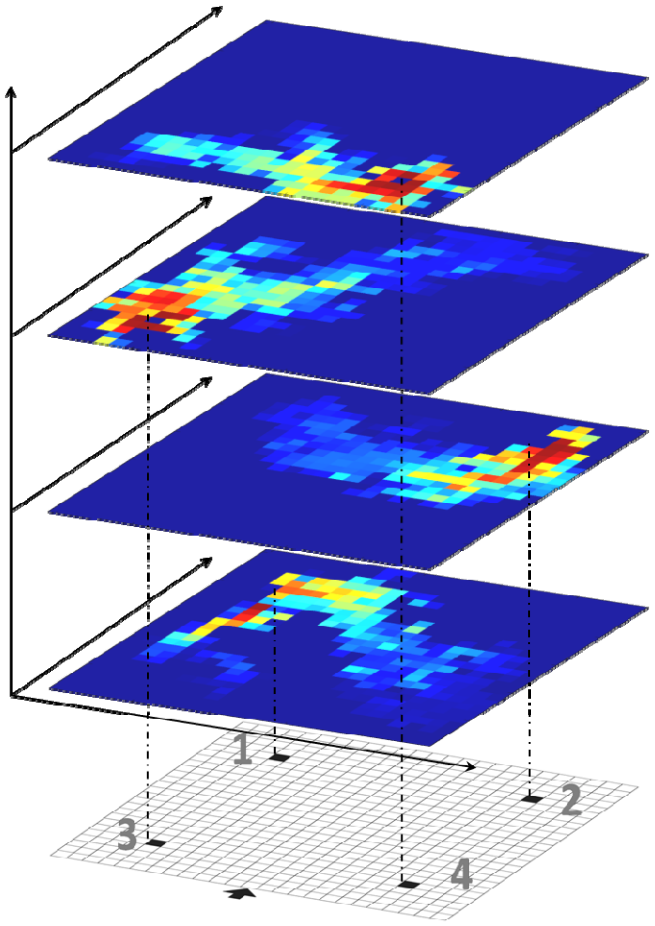


Fig.7. Visualization of all learned value functions (corresponding to all goals). The higher location, to the higher goal it corresponds.

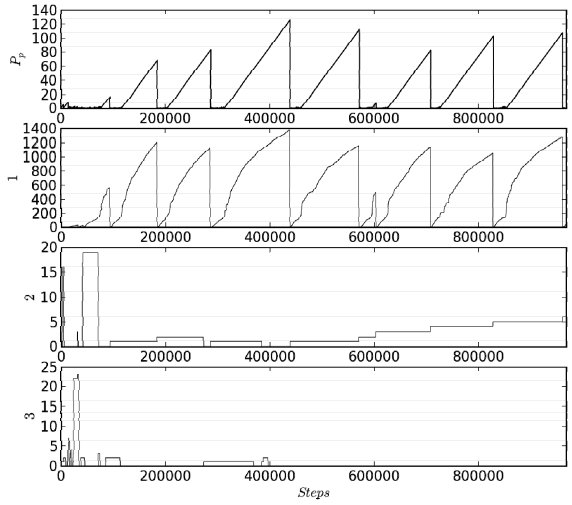
In the performed experiment this hierarchical structure is linear, but more complex relations between abstract goals can be explored as well. An example of such a situation is when specified resource can be restored using more than one way, then the agent would learn a tree type structure of goal dependencies.

B. RL Agent Learning Process

In the case of the RL agent we use the same environment, with four kinds of resources, renewable according to the same rules as in the simulation with the ML agent. By analogy, figure 8.a. shows a primitive pain signal and resource consumption counters. Because the RL based agent doesn't have an intrinsic motivation system, consumption

counters shouldn't be interpreted as abstract pain signals.

a)



b)

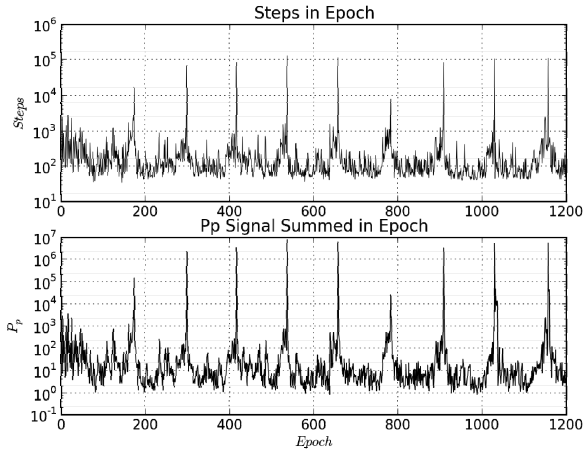


Fig.8. RL agent learning process: a) primitive pain signal P_p and resource consumption counters, b) simulation steps and primitive pain aggregated by epochs.

They can be used to illustrate how the agent uses the available resources. The simulation consists of 1200 epochs, with the total number of steps being above 500 thousands. Long periods between subsequent primitive pain reductions mean that the agent did not learn the location of resources in the environment precisely enough to be able to manage its pain signal (i.e. hunger). This situation is depicted in Fig.8.b. We can observe that every some period of time the agent has to spend a lot of time (simulation steps) searching for resources. This is because basic resources are exhausted and the only way to replenish them is to find a higher level resource. The impossibility of finding a reward causes primitive pain to increase. This is depicted as characteristic peaks.

C. *Effectiveness of ML and RL Approaches*

Results presented in this subsection were obtained in a series of 6 simulations. Thanks to this, it is easier to observe some characteristic features of both methods. The comparison of average values of the primitive pain signal is depicted in Fig.9. These average signals show the agent's effectiveness in reducing primitive pain signal P_p. They present the agent's ability to learn all the environment rules necessary to keep its pain signal low.

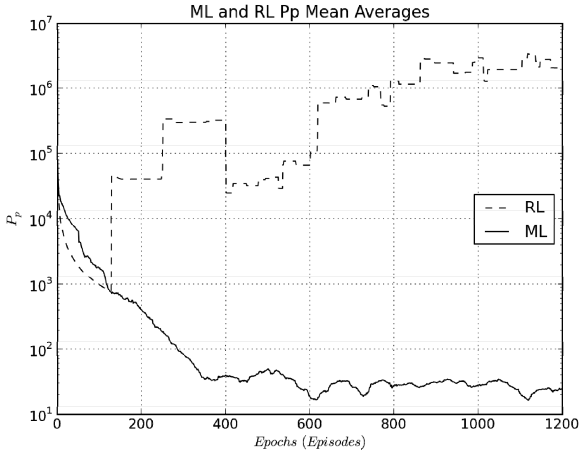


Fig.9. Moving averages of primitive pain Pp values as a function of simulation time (in episodes/epochs)

We can observe that the difference between primitive pain average values for the two types of agents reaches up to 5 orders of magnitude. After about 350 epochs in the case of the ML based agent we can observe that primitive pain signal is stabilized. In the case of RL based agent the signal still goes up. This means that RL agent is not able to replenish basic environment resources.

Minimization of the primitive pain signal is one aspect of comparison. The second one is maximizing total base reward – the sum of all gained base rewards (i.e. food). The efficiency of the ML based agent in comparison to the RL agent is shown on figure 10. We can observe the accumulated value of base rewards gained during the whole simulation. The base reward is the only one kind of reward that is determined by the designer. It is an objective reward (not internal) that RL method use to control agent's behavior.

In the initial phase of the learning process, RL agent seems to have an upper hand as it accumulates more reward. This is because the ML agent is learning several value functions simultaneously. This short period ends before the environment is depleted of resources (after about 350 thousands steps). In this period of time the ML agent gains enough knowledge about the environment to be able to replenish any required resource needed to win the base reward.

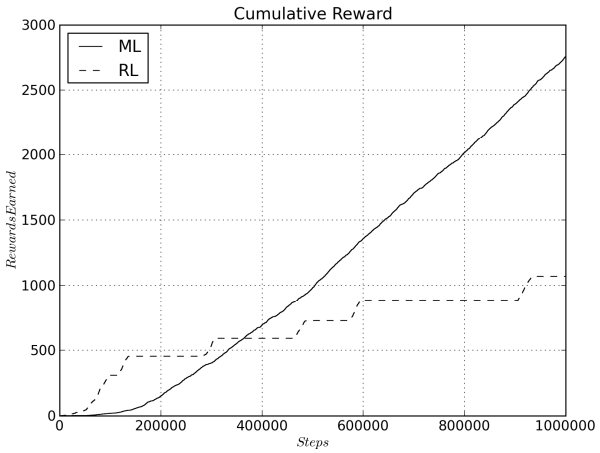


Fig.10. Effectiveness of winning base reward.

In case of the RL agent the periods of fruitless search for reward become increasingly longer.

VI. Conclusion

This paper presents a motivated learning (ML) method [19] in an application where reinforcement learning (RL) is combined with goal creation system (GCS). The agent's internal hierarchy of goals built gradually through interaction with the environment constitutes an internal motivation system, which manages agent's activity. The agent's essential aim is to survive in a hostile, dynamically changing, environment, with a minimal pain. This aim requires the agent to frequently acquire the base reward, determined by the designer.

For these simulations a special kind of test environment has been designed, where availability of the base reward is variable, and depends on the agent's ability to use several kinds of resources.

In the presented experiments we have shown how ML agent learns dependencies between environment resources, and how it can use its knowledge to operate in such environment. In the presented experiments the ML based agent, that has the ability to set its internal goals autonomously, is able to fulfill the designer's goals more effectively than the RL based agent.

Acknowledgment

The authors would like to thank James Graham form Ohio University for his help and suggestions in preparing this research work.

References

[1] A. Barto, S. Singh, and N. Chentanez, "Intrinsically motivated learning of hierarchical collections of skills," *Proc. 3rd Int. Conf. Development Learn.*, San Diego, CA, pp. 112–119, 2004.

[2] R. White, "Motivation reconsidered: The concept of competence," *Psychological review*, 66: pp. 297–333, 1959.

[3] W. Schultz, "Getting Formal with Dopamine and Reward," *Neuron*, Vol. 36, pp. 241-263, 2002.

[4] P. Dayan and T. J. Sejnowski. "Exploration bonuses and dual control. Machine Learning," 25:5–22, 1996.

[5] P. Dayan and B. W. Balleine. "Reward, motivation and reinforcement learning," Neuron, 36:285–298, 2002.

[6] J. Schmidhuber, "Curious model-building control systems," In Proceedings of the International Joint Conference on Neural Networks, Vol. 2. Singapore, IEEE, pp. 1458–1463, 1991.

[7] J. Schmidhuber, "A possibility for implementing curiosity and boredom in model-building neural controllers," In From Animals to Animats: Proceedings of the First International Conference on Simulation of Adaptive Behavior, pp. 222–227, Cambridge, MA, MIT Press, 1991.

[8] J. Schmidhuber and J. Storck, "Reinforcement driven information acquisition in nondeterministic environments," Technical report, Fakultat fur Informatik, Technische Universit at Munchen, 1993.

[9] C. J. C. H. Watkins, "Learning From Delayed Rewards," PhD thesis, Cambridge University, Cambridge, UK, 1989.

[10] J. Schmidhuber, "Developmental robotics, optimal artificial curiosity, creativity, music, and the fine arts," Connection Science, Volume 18, Number 2, June 2006 , pp. 173-187(15) , 2006.

[11] P-Y Oudeyer, F. Kaplan, and V. Hafner, "Intrinsic motivation for autonomous mental development," in *IEEE Transactions on Evolutionary Computation*, 11(2), pp. 265–286, 2007.

[12] P-Y. Oudeyer, A. Baranes, and F. Kaplan, "Intrinsically Motivated Exploration for Developmental and Active Sensorimotor Learning," in Sigaud, O. and Peters, J. eds., *From Motor Learning to Interaction Learning in Robots,* pp. 107-146, Studies in Computational Intelligence, vol. 264/2010, Springer Berlin/Heidelberg, 2010.

[13] S. Roa, G. J. M. Kruiff, and H. Jacobson, "Curiosity-driven acquisition of sensorimotor concepts using memory-based active learning," *Proceedings of the IEEE Intl. Conf. on Robotics and Biometrics*, pp. 665-670, 2008.

[14] A. G. Barto and S. Mahadevan, "Recent advances in hierarchical reinforcement learning," Discrete Event Systems, Special issue on reinforcement learning, 13, pp. 41–77, 2003.

[15] P. Dayan and G. E. Hinton, "Feudal reinforcement learning," In *Advances in Neural Information Processing Systems*, 5, 1993.

[16] R. Parr and S. Russell, "Reinforcement learning with hierarchies of machines," In *Advances in Neural Information Processing Systems: Proceedings of the 1997 Conference*, Cambridge, MA, 1998.

[17] T. G. Dietterich, "Hierarchical reinforcement learning with the maxq value function decomposition," *Journal of Artificial Intelligence Research*, 13, pp. 227–303, 2000.

[18] B. Bakker and J. Schmidhuber, "Hierarchical Reinforcement Learning with Subpolicies Specializing for Learned Subgoals," in M. H. Hamza (Ed.), Proceedings of the 2nd IASTED International Conference on Neural Networks and Computational Intelligence, NCI 2004, Grindelwald, Switzerland, pp. 125-130, 2004.

[19] J. A. Starzyk, "Motivation in Embodied Intelligence" in Frontiers in Robotics, Automation and Control, I-Tech Education and Publishing, Oct. 2008, pp. 83-110, 2008.

[20] J. Bach, Principles of Synthetic Intelligence, Oxford University Press, 2009.

[21] J. A. Starzyk, J. T. Graham, P. Raif, and A-H.Tan "Motivated Learning for Autonomous Robots Development", Cognitive Science Research, January 2011. doi:10.1016/j.cogsys.2010.12.009

[22] Rummery, Niranjan, "On-Line Q-Learning Using Connectionist Systems," Tech. rep. CUED/F-INFENG/TR166, Cambridge University, 1994.