Opportunistic Motivated Learning Agents

James Graham[1], Janusz Starzyk[1,2], Daniel Jachyra[2]

[1]School of Electrical Engineering and Computer Science, Ohio University, Athens, OH, USA
{jg193404, starzykj}@ohio.edu
[2]University of Information Technology and Management, Rzeszow, Poland.
djachyra@wsiz.rzeszow.pl

Abstract. This paper presents an extension of the Motivated Learning model that includes environment masking, and opportunistic behavior of the motivated learning agent. Environment masking improves an agent's ability to learn by helping to filter out distractions, and the addition of a more complex environment increases the simulation's realism. If conditions call for it opportunistic behavior allows an agent to deviate from the dominant task to perform a less important but rewarding action. Numerical simulations were performed using Matlab and the implementation of a graphical simulation based on the OGRE engine is in progress. Simulation results show good performance and numerical stability of the attained solution.

Keywords: Motivated learning, cognitive agents, reinforcement learning, goal creation.

1 Introduction

In this paper we expand on our previous Motivated Learning (ML) work and show how it can yield an opportunistic learning system. The goal of this paper is to discuss the changes made to the algorithm presented in [2, 9] and to indicate how they yield a more complex and efficient system.

Motivated Learning is defined as an extension of reinforcement learning that uses intrinsic motivations as a way to build a motivated agent. These motivations are internal to the agent and are derived from various "pain" signals. "Pain" signals represent an underlying need, drive, or irritation. These signals can originate from internal states of the agent (memories and/or other internal drives) or external environmental states such as basic needs for sustenance and shelter.

This previously proposed system [2] is self-organizing and controls an agent's behavior via competition between the dynamically changing needs of the system. In some respects, ML can be seen as an extension of reinforcement learning, in that it receives its reinforcement from the environment for its primitive objectives (i.e. its most basic needs). Upon this initial structure a more complex systems of goals and values can be built to establish complex internal motivations for advanced stages of development. This includes the creation of abstract concepts and needs and the creation of internal rewards for satisfying these needs.

However, unlike reinforcement learning, motivated learning, does not suffer from the "credit assignment" problem [6, 7]. Reinforcement learning typically spreads the value of a reward to earlier actions via a temporal difference mechanism; however, this often leads to credit assignment to actions that had no relation to the reward. In contrast, the ML approach, while not necessarily immune is much more resistant to this problem due to the focus on motivations and the creation of abstract needs. The abstract pains/needs serve to "break-up" the reward chain to the "primitive" needs (or basic reward in RL) and improve an agent's overall learning ability.

Motivated learning also has some similarities and uses some elements of BDI (belief-desire-intention) agents. The work by Rao and Georgeff [3] is one of the first papers to consider how an actual BDI agent might be implemented and serves as a bridge between BDI theory and actual practice. Others such as Dastani et al. [4] and Wooldridge [5] deal more with individual aspects of BDI such as the deliberative processes and the open-mindedness of an agent.

The presented ML model can be related to a BDI model in several respects. ML has belief in the sense that it observes the environment and extracts rules and state information to create its own internal "representation" of the state of the world around it. The ML agent's beliefs link its perceptions to semantic knowledge about the environment as coded through the system of perceptions, pain centers, goals, and planned actions. Desires in BDI agents correspond to motivations as expressed by the pain centers in ML. The pains (or needs) compete for attention and are not handled unless one of them dominates or passes some threshold. And lastly, intentions are represented in the ML agent as the selected (or preferred) method for implementing a goal chosen by the agent. The most notable difference between BDI and ML agents is that BDI agents have their actions predetermined by their designers, while ML agents create their own as they learn to exist within their environment. BDI agents lack the ability to define more complex abstract motivations as is typically performed by ML agents.

In our earlier work [2], we presented a basic implementation of a goal creation system and the motivated learning idea. In this paper, several enhancements of ML are implemented, ranging from enhancements to the environment and the ability to attempt and track multiple tasks. Our current work is modified to accommodate the computational model of a cognitive machine [1]. We discuss the effects these enhancements have had on the ML algorithm. Finally, we discuss ongoing work and present our future plans.

2 Design of the Motivated Learning System

Motivated Learning uses a neural network to weigh its options and create goals. Goals are created not only in response to externally set motivations (primitive pains) but also in response to needs determined at the various levels of abstract goals created by the machine during the learning process. As the machine learns, it develops associations between its percepts and builds representations of discovered abstract concepts. Initially these representations relate directly to its perception of the

environment, however, over time, and with experience, the machine will begin to perceive increasingly complex relationships and behaviors.

In order to satisfy an agent's motivation, a mechanism is needed to select which goals, or actions, to pursue. This mechanism needs to be able to process existing motivations and build new ones. Signals representing various abstract pains will compete against each other with input from the environment and other parts of the agent's architecture. Additionally, as the machine effects the environment, the changes in the environment affect the machine. These changes will be perceived by the agent and influence its cognitive process. It is possible, however, to partially block outside influence in instances where it is "desirable" for the machine to focus on some internal mental task. For instance an agent may need to spend time performing mental analysis of various possible scenarios, steps, and combinations of actions needed to perform the task.

Once the agent determines the dominant need/pain it will attempt to choose a goal or action to remedy the pain. To do so, it uses a winner-take-all (WTA) neural-network (NN) based approach, whereby bias weights and goal weights are decreased or increased based on the success or failure of a particular action. In the case of bias weights, they act on the previously mentioned pain biases, and increase when the resource associated with a specific pain is shown to be of relevance. However, they decrease gradually when a resource (or pain) is unused. Goal weights increase when a pain is decreased by the completions of the action associated with the goal. They decrease when the pain is unaffected, or worse, increased by the goal implementation. For a more detailed overview of the basic internal structures behind a ML agent refer to [2].

2.1 Expanded Environment and Masking

Presented in [2] environment for testing the ML agent consisted of only 6 sensors and 6 motor commands allowing for a total of 36 possible goals/actions.

While this environmental set-up is an appropriate first step, it is too simple to properly evaluate the efficacy of the model. Therefore, a significantly more complex environment has been implemented (see Figure 1) that utilizes 17 sensors and 26 motor commands for a total of 442 different possible goals/actions. This leads to a much longer search time for the correct actions.

In the basic simulation, there is only a single primitive pain, while in the more complex simulation, depicted in Figure 1, there are three primitive pains with six resources and 8 motor commands directly associated with them. Thus, at the very beginning of the expanded environment, the agent has a greater number of choices to process than in the basic simulation. To improve its learning in such environment we developed goal "masking".

The idea behind the use of masking is to block certain sensors and motor commands so that the agent does not perceive them until a certain time or environmental state is attained. The "masked" environment, combined with the masking of motor commands, emulates guided learning. The agent is not directly told what to do, it is guided and accelerates its learning by limiting available options.

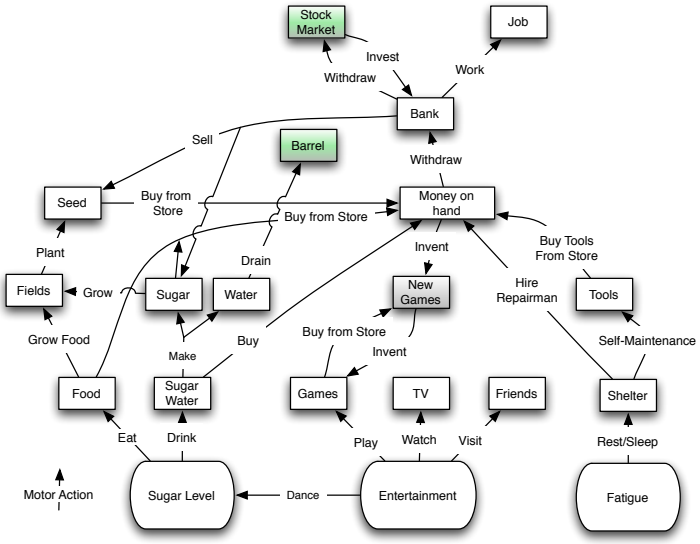

Figure 1. Expanded Environment.

Currently environments are focused on resource consumption. However, work is currently being undertaken to expand into areas such as location, motion, on/off options, and so on.

2.2 Resource Utilization and Multi-cycle Task/Goals

At the beginning of a simulation, the agent does not know which resources relieve what pains, nor does it know how the use of one resource impacts another one. Exploring a common situation such as dealing with "hunger" presents a good example of resource utilization. In this example, hunger is the pain, and the associated resource can be represented as blood sugar. When hungry the machine eats, but it does not know how much food it needs to consume to alleviate its hunger pain. As the simulation progresses, the machine will gradually be able to estimate the relationship between food and blood sugar level by evaluating the hunger pain. It does this by calculating the ratio of expected/desired pain reduction vs. actual pain reduction and determines the amount of resource needed by comparing it to what was used previously and its effects on the pain. Refining the ratio estimation will continue every time the hunger pain is above threshold. Additionally, because pain is a logarithmic function of resource utilization and because w_{BP} changes with time affecting the pain value, the ratio will continue to change over time.

Goals are defined in our system as a need, or in BDI terms, a desire to reduce a specific pain. A task is an action selected by the machine in the hope that it will reduce the dominant pain. In a real world, it will not always be possible to complete a task or action in one cognitive cycle. Therefore, the system has been modified to allow for the effort required to perform the actions selected. In the current

implementation, effort requirements are based solely on the amount of time needed to perform the selected motor action. The implementation takes into account the quantity of resource consumed to determine how long it takes to complete an action (to reduce pain below threshold) and travel time in cases where movement is needed. However, the rate at which pains change is not necessarily constant and depends on the task.

2.3 Opportunistic Behavior

The following describes an opportunistic task selection system, whereby the agent can decide to pause in execution of its current task to perform another task. In order to select another task, there needs to be sufficient difference in the required effort or change in the pain levels for the deviation from the original task to be worthwhile. In order to decide the value of a task (action value), the level of the pain that initiated the task, time needed to complete the task (shorter is better), and how useful is the task, are taken into account:

$$Action\ Value = \frac{1}{1+\Delta t}(P + \sum \Delta P_e)\ . \quad (1)$$

The amount of time needed to perform the action is represented by Δt, P is the pain considered for reduction, and ΔP_e represents the predicted changes in the pain levels. The agent computes "Action Values" in conjunction with w_{PG} weights generating a "Task Value" for all possible actions. It then operates on the winning action. If an ongoing action is interrupted, the agent can resume it where it was left off. The biological and algorithmic reasoning for "opportunistic behavior" has the potential to be fairly complex. However, we take a relatively simple approach by attempting to evaluate the "worth" or a task in progress against that of other potential tasks.

3 Simulation of Motivated Learning in Virtual Environments

Virtual environments are excellent developmental platforms for embodied intelligence concepts. Many robotics projects such as the iCub [8] make use of simulated environments. The iCub project is an open hardware and software platform for developmental robotics. Of course, it is not practical for most people to purchase the iCub hardware (over $200,000 for a complete robot), meaning many have to rely on a virtual environment, the iCub Simulator, which is included in the free software package.

The initial versions of the Motivated Learning software [9] used a very simple simulated environment to demonstrate its advantages over RL; however, this is inadequate for testing of more complex behaviors and systems.

This is why we are working on integrating our agent with NeoAxis [10]. NeoAxis provides a graphical game engine with many existing assets and the ability to add more as needed. It is designed to be easily modifiable by users, and is provided as a free SDK for non-commercial use. The NeoAxis engine itself is based on OGRE (Object-oriented Graphics Rendering Engine) [11]. Figure 2 shows an image from one of the demo maps included in the NeoAxis SDK.

Figure 2. NeoAxis graphics example.

To embed the ML agent into NeoAxis we decided to modify the game's AI and integrate the agent into the decision making part of the code. Additionally, a new class of environmental objects referred to as "Resources" was created to simplify the transition. Integrating the ML agent required providing information from the environment to the agent and receiving and interpreting the agent's responses.

3.1 Simulation and discussion of results

While at the time of writing integration with NeoAxis is under development, the ML agent described in Section 2 is fully operational.

Figures 3 and 4 depict the results from the simulation in a more complex environment shown on Figure 1. Figure 3 shows resource utilization for simulated environment over a period of 30,000 iterations. The figure displays the results in 2000 iteration increments, or to clarify from zero to 2000, from 6000 to 8000, from 12,000 to 14,000, and so on. Figure 4 displays iterations in the same manner as Figure 3. Notice in Figure 4 how the pains stabilize at relatively low levels. And while there is at least one instance where the pains spike to greater levels, once the machine learns how to handle them they stabilize once more.

3.2 Impact of opportunistic behavior on results

In section 2.3 several changes to the original motivated learning algorithm were discussed including, an expanded environment model, masking of sensors and motors, quantitative rather than probability based resource availability, actions requiring multiple cycles, and opportunistic behavior.

One can observe how increasing the complexity of the environment changes the simulation dynamics. The most obvious effect is that the simulation requires longer time to run in a more complex environment. From Figure 4, it is apparent that only a fraction of the available resources have been utilized in 30,000 iterations. This is partially an effect of masking, since many resources are not available until the system reaches a higher level of development.

The effect of masking sensory inputs and motor commands should be clear. With higher "level" sensors and motor commands masked until the agent needs them, the agent will have fewer options to select from, and thus will be able to learn how to

navigate the environment more quickly. Masking can either be done manually (via setting the masks to be enabled or disabled at specific times) or "automatically" by probing the agent's internal weights to see if it has learned a desired concept and introduce the sensors/motors it would need for the next learning level. Automatic masking has the most favorable impact on improving the learning speed.

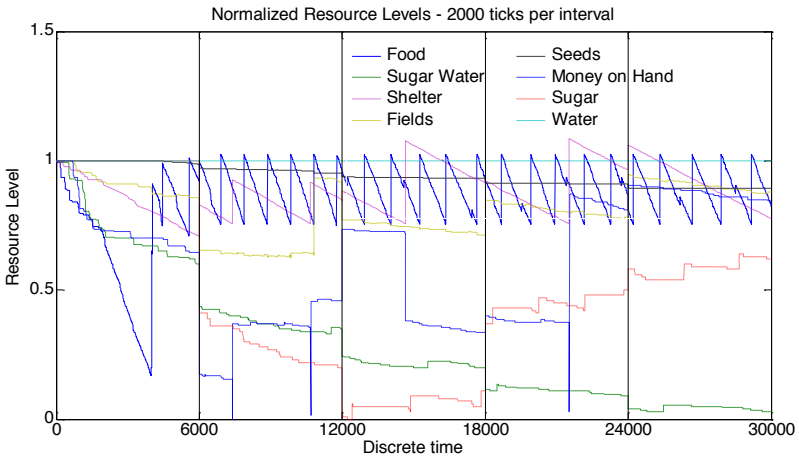

Figure 3. Resource Usage.

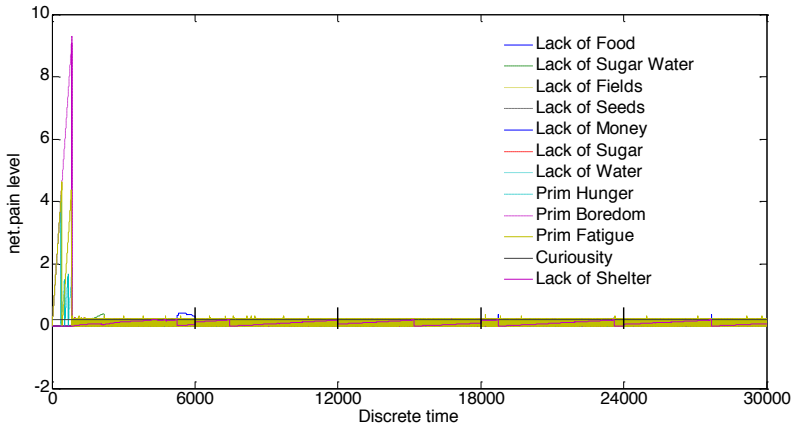

Figure 4. Pain vs. Iterations

Resource consumption is tied in with the environment model. Early versions of the environment model were probability based, however, in the recent version the environment contains measurable quantities of each resource. As such, it became important to the agent to be able to determine how much of each resource to use (and how it affects another resource). This change did not affect the performance of the agent in a significant way, however it helped to integrate the agent with a more realistic environment such as the NeoAxis.

The use of multi-cycle tasks was another significant change for the agent. It has the effect of significantly increasing the amount of time required for simulation. It also paves the way for a more realistic implementation of the agent. As has been mentioned, tasks can easily take variable amounts of time. The use of variable task times has also allowed for the implementation of more complex opportunistic behavior.

4. Conclusions

In this paper we presented Opportunistic Motivated Learning model. Also discussed was our effort to integrate the ML agent with simulated environments using NeoAxis graphics engine. As confirmed by the simulation results our model can surpass other reinforcement learning based models [2]. With the planned additions to the motivated learning model, combined with an effective cognitive model discussed in [1, 12] we hope to design a machine capable of cognition. Simulation results showed good performance and numerical stability of opportunistic ML. In the future we would like to implement our model not only in various simulated environments but also in real robots.

References

1. J. A. Starzyk and D. K. Prasad, "A Computational Model of Machine Consciousness" to appear in International Journal of Machine Consciousness, 2011.
2. J. A. Starzyk, J. T. Graham, P. Raif, and A-H. Tan, "Motivated Learning for Autonomous Robots Development", to appear in Cognitive Science Research, 2011.
3. Rao, A. S., Georgeff, M. P. (1995): BDI Agents: From Theory to Practice. In Lesser, V. (ed.): Proceedings of the 1st International Conference on Multi-Agent Systems (ICMAS):312–319. MIT Press.
4. Dastani, M., Dignum, F., Meyer, J.-J. (2003): Autonomy, and Agent Deliberation. In Proc. of the 1st International Workshop on Computational Autonomy (Autonomy 2003).
5. Wooldridge, M. (2000): Reasoning about Rational Agents. Intelligent Robots and Autonomous Agents. The MIT Press, Cambridge, Massachusetts.
6. R. S. Sutton, Temporal Credit Assignment in Reinforcement Learning. PhD thesis, University of Massachusetts, Amherst, MA, 1984.
7. W.-T. Fu and J. R. Anderson, "Solving the Credit Assignment Problem: Explicit and Implicit Learning with Internal and External State Information," Proceedings of the 28th Annual Conference of the Cognitive Science Society, Hillsdale, NJ: LEA, 2006.
8. iCub, "RobotCub – An Open Framework for Research in Embodied Cognition," http://www.robotcub.org/, 2004.
9. J. A. Starzyk, "Motivation in Embodied Intelligence," in *Frontiers in Robotics, Automation and Control*, I-Tech Education and Publishing, Oct. 2008, pp. 83-110.
10. NeoAxis -- all-purpose, modern 3D graphics engine for 3D simulations, visualizations and games. http://www.neoaxis.com/
11. OGRE -- Open Source 3D Graphics Engine. http://www.ogre3d.org
12. J.A. Starzyk, "Mental Saccades in Control of Cognitive Process", Int. Joint Conf. on Neural Networks, San Jose, CA, July 31 - August 5, 2011.