9-2010

A biologically-inspired cognitive agent model integrating declarative knowledge and reinforcement learning

Ah-hwee TAN
Singapore Management University, ahtan@smu.edu.sg

Gee-Wah NG

A Biologically-Inspired Cognitive Agent Model Integrating Declarative Knowledge and Reinforcement Learning

Ah-Hwee Tan
Nanyang Technological University
Nanyang Avenue, Singapore 639798
Email: asahtan@ntu.edu.sg

Gee-Wah Ng
DSO National Laboratories
20 Science Park Drive, Singapore 118230
Email: ngeewah@dso.org.sg

***Abstract*—The paper proposes a biologically-inspired cognitive agent model, known as FALCON-X, based on an integration of the Adaptive Control of Thought (ACT-R) architecture and a class of self-organizing neural networks called fusion Adaptive Resonance Theory (fusion ART). By replacing the production system of ACT-R by a fusion ART model, FALCON-X integrates high-level deliberative cognitive behaviors and real-time learning abilities, based on biologically plausible neural pathways. We illustrate how FALCON-X, consisting of a core inference area interacting with the associated intentional, declarative, perceptual, motor and critic memory modules, can be used to build virtual robots for battles in a simulated RoboCode domain. The performance of FALCON-X demonstrates the efficacy of the hybrid approach.**

***Keywords*-Cognitive Agents; Knowledge Representation; Reinforcement Learning**

I. Introduction

In the fields of artificial intelligence and cognitive science, there has been a debate over symbolic and sub-symbolic (connectionist) representation of human cognition [1], motivating two parallel streams of research directions. The symbolic field holds the view that, the human cognitive system uses symbols as a representation of knowledge and intelligence is through the processing of symbols and their respective constituents. Soar [2] and ACT-R [3], for example, are representatives of symbolic systems. On the other hand, the sub-symbolic camp argues that the human cognitive system uses a distributed representation of knowledge and is capable of processing this distributed representation of knowledge in a complex and meaningful way [4]. Sub-symbolic or connectionist systems are most generally associated with the metaphor of neural models, composing of neural circuits that operate in parallel. The key strengths of sub-symbolic systems lie in their learning abilities and allowance for massively parallel processing.

In this paper, a cognitive agent model, known as Fusion Architecture for Learning and Cognition - eXtension (FALCON-X), is proposed, based on an integration of the Adaptive Control of Thought (ACT-R) architecture [3] and the fusion Adaptive Resonance Theory (fusion ART) neural model [5]. Fusion ART is a generalization of self-organizing neural models known as Adaptive Resonance Theory [6]. By expanding the original ART model consisting of a single pattern field into a multi-channel architecture, fusion ART unifies a number of network designs supporting a myriad of learning paradigms, including unsupervised learning, supervised learning and reinforcement learning. While retaining the structure of the visual, manual, intentional and declarative modules of ACT-R, the proposed architecture replaces the symbolic production system with a fusion ART neural network serving as the core inference area for fusing and updating the pattern activities in the four memory buffers. In addition, a critic channel is incorporated to regulate the attentional and learning processes of the core inference area.

FALCON-X may potentially be used to model a wide range of cognitive processes. In this paper, we describe how procedural knowledge can be learned as sensory-motor mappings through reinforcement learning. We also illustrate how declarative knowledge can be encoded using a class of composite neural circuits as long-term memories. The proposed model has been used to build virtual robots for battles in a simulated RoboCode domain. Based on this domain, we show how the learned procedural and declarative knowledge can be integrated for decision making and problem solving.

The rest of this paper is organized as follows: After a brief review of ACT-R, we present the FALCON-X architecture. We then describe the generic FALCON-X dynamics, followed by details on how it may learn procedural knowledge and represent declarative knowledge. The experimental study on Robocode AI game is reported subsequently, followed by the conclusion.

II. The ACT-R Architecture

Adaptive Control of Thought (ACT-R) is developed as a model of human cognition using empirical data derived from experiments in cognitive psychology and brain imaging [3]. Among the various cognitive architectures proposed, ACT-R notably has packed a wide range of cognitive functions in a generic and scalable architecture supported by extensive references to human brain anatomy.

ACT-R consists of four basic modules, namely the visual, manual, intentional and declarative modules, interconnected via a central production system. The visual module receives sensor input from the external world, the manual module executes the actions produced by the production system, the intentional module maintains the task-related objectives

978-0-7695-4191-4/10 $26.00 © 2010 IEEE
DOI 10.1109/WI-IAT.2010.210

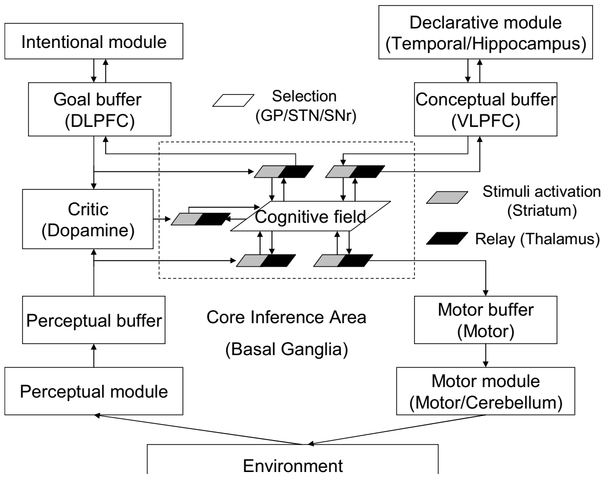

Figure 1. The FALCON-X architecture.

during the cognition process, and the declarative module contains the long-term memory. Each module has a buffer to keep track of one's internal state during problem solving. The central production system activates the relevant production rules based on the information available in the buffers of the four modules.

The high level knowledge representation of ACT-R (i.e. chunks and production rules) has made it easy to represent complex relationships. However, much of the knowledge that is acquired within ACT-R is typically written by experienced programmers and not learned through interaction with outside stimuli [1]. In addition, although ACT-R has made extensive references of its modules to specific regions in the human brains, the central production system of ACT-R remains symbolic and not neurally plausible.

III. The FALCON-X Architecture

The FALCON-X architecture is presented herein, based on an integration of the ACT-R cognitive architecture and the fusion ART neural model (Figure 1). While retaining the structure of the ACT-R's peripheral memory modules, the proposed architecture replaces the symbolic production system with a fusion ART neural network serving as the core inference area for fusing and regulating the pattern activities in the four memory buffers. As a key departure from ACT-R, an explicit critic module is incorporated, which provides reward signals to the core inference area. The roles and functions of the various modules are described as follows.

- The **Perceptual Module** receives input signals from the external environment. In actual applications, some preprocessing of the input signals may be necessary. The input signals are typically represented as a set of vectors of values in the perceptual buffer, taken from the sensors.
- The **Motor Module** receives and executes the actions, produced by a readout action from the core inference area. The actions are typically represented as a set of discrete values in the motor buffer, each of which denotes one of the possible actions.
- The **Intentional Module** consists of the task-relevant goals serving as the context. Each goal is represented as a target state vector in the goal buffer, representing the active goals of the agent.
- The **Declarative Module** consists of middle-term and long-term memories, relevant to the tasks. The memory can be represented in many ways, such as rules or neural networks.
- The **Critic Module** computes reward signals that indicate the goodness of the actions taken. Generally, there can be two type of critics, namely, reward signals received from the external environment; and estimated payoff computed based on the current states and the target states.
- The **Core Inference Area** receives activations from the five memory modules and acts as a key driver of the inference process. In FALCON-X, the inference mechanism is realized via a five-step bottom-up and top-down neural processes, namely code activation, code competition, activity readout, template matching and template learning, described in the next section.

IV. The FALCON-X Dynamics

In each inference cycle, the core inference area of FALCON-X receives input signals from the perceptual, intentional and declarative modules, and selects a cognitive node based on a bottom-up code activation and competition process. Whereas the intentional buffer maintains the active goals, the declarative module provides the relevant conceptual memory for code selection. The inference engine may also receive reward signals from the critic module. It is important to note that at any point in time, FALCON-X does not require input to be present in all the pattern channels.

Upon activity readout, a template matching process takes place to ensure that the matched patterns in the four memory modules satisfy their respective criterion. If so, a state of resonance is obtained and the template learning process encodes the matched patterns using the selected cognitive node. Otherwise, a memory reset occurs, following which a search for another cognitive node begins. During prediction or action selection, the readout patterns typically include the actions to be executed in the motor module. In other cases, the conceptual memory buffer is updated and the goals may change as a result of inference.

V. Learning Procedural Knowledge

In this section, we illustrate how FALCON-X, specifically the core inference area together with the perceptual, motor and critic modules, can acquire procedural knowledge through reinforcement learning in a dynamic and real-time environment.

A. Reactive Learning

A reactive learning strategy, as used in the R-FALCON (Reactive FALCON) model [7], performs fast association between states and actions, based on reward signals. Given a reward signal (positive feedback) in the critic buffer, FALCON associates the current state in the perceptual buffer with the selected action represented in the motor buffer. If a penalty is received, it learns the mapping among current state, the complement pattern of the action taken and the complement value of the given reward.

B. Temporal Difference Learning

A key limitation of reactive learning is the reliance on the availability of immediate reward signals. TD-FALCON [8], [9] is a variant of FALCON that incorporates Temporal Difference (TD) methods to estimate and learn value functions of action-state pairs $Q(s, a)$ that indicates the goodness for an agent to take a certain action a in a given state s.

Given the current state s, TD-FALCON first decides between exploration and exploitation by following an action selection policy. For exploration, a random action is picked. For exploitation, TD-FALCON performs instantaneous searches for cognitive nodes that match with the current states and at the same time provide the highest reward values using a direct access procedure. Upon receiving a feedback from the environment after performing the action, a TD formula is used to compute a new estimate of the Q value of performing the chosen action in the current state. The new Q value is then used as the teaching signal for TD-FALCON to learn the association of the current state and the chosen action to the estimated Q value.

VI. Incorporating Declarative Knowledge

Declarative knowledge refers to long-term memories that are consciously available. Symbolic representations, such as rules and concept hierarchy, are typically used to model declarative memory. In this work, we illustrate that declarative knowledge can also be represented using connection weights of neural networks. In view that most neural networks only deal with proposition logic, a type of composite neurons is used here for representing complex rules involving variables.

A composite neuron consists of two interconnected subneurons: gating node CNg and activation node CNa. The CNg node is responsible for condition gating, while the CNa node is used to infer output concepts and update the conceptual buffer. As shown in Figure 2, the declarative module can be implemented as a field of composite neurons that receive input from the conceptual buffer and in turn update the conceptual buffer through the activation of a selected composite neuron.

The conceptual buffer consists of a set of conceptual nodes, each of which serves as the input and may be activated by the core inference area and declarative module.

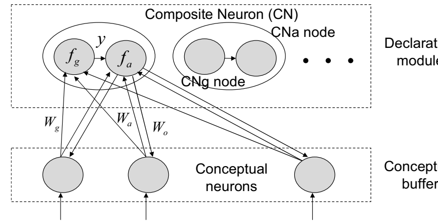

Figure 2. The declarative module implemented by composite neurons with direct connection to conceptual buffer.

Table I
TWO SAMPLE RULES FOR ROBOCODE DOMAIN.

Rule 1:	IF Relative_Target_Bearing = x THEN Turn_Gun_Angle = x
Rule 2:	IF Turn_Gun_Angle = x AND Adjust_Angle = y THEN Final_Gun_Bearing = x + y

The *conceptual nodes* thus function as a working memory for accumulating short-term memory activities.

VII. Experimental Study

Robocode (http://robocode.sourceforge.net) is a virtual battle simulator, in which one can create robots to fight against each other (Figure 3). A large number of robots of various design (http://robowiki.net) have been developed over the years. However, most of them employ some predefined strategies without any learning or adaption ability.

A. Incorporating Domain Knowledge

A simple and intuitive firing strategy is to point the gun towards the enemy's direction and fire. However, as bullets take time to travel, the enemy may have moved to another location during the time interval. To overcome this problem, FALCON-X adopts an "aim-off" strategy. Specifically, instead of firing directly towards the enemy's current position, it turns the gun an angle away from the enemy's bearing anticipating the enemy's future location. This firing strategy is captured into two rules listed in Table I and illustrated in Figure 3.

B. Learning Firing Strategy

In this section, we illustrate how FALCON-X can learn the "aim-off" angle through reinforcement signals. Here we name our robots as R-FALCONBot and TD-FALCONBot, which use the reactive and temporal difference (TD) learning rules respectively.

- **Sensory Input:** Two state variables are used for learning, namely the distance from FALCONBot to the enemy's robot and the enemy's heading relative to its bearing.
- **Motor Output:** The motor output will be the Adjust_Angle from the enemy bearing. As it is a continuous variable, the angle is discretized into several blocks of equal interval.

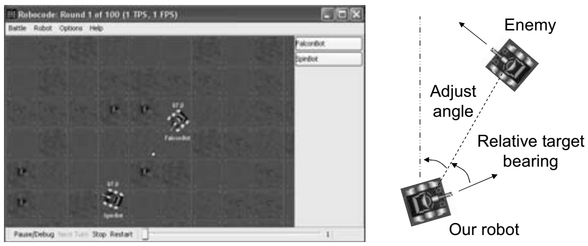

Figure 3. The Robocode battle field and firing strategy.

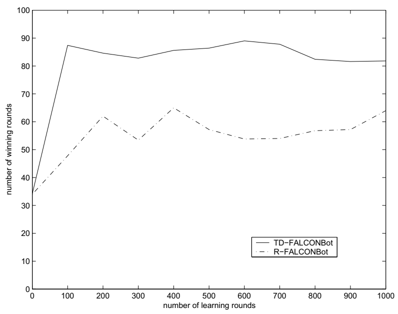

Figure 4. The performance of FALCON-X against *SpinBot*.

- **Critic Feedback:** When a bullet hits the enemy, a reward of 1 is credited to the action. Conversely, when the bullet misses the target, a zero reward is given.

C. Results and Analysis

Our evaluation consists of five set of experiments, each of which lasts for 1000 learning rounds of battles. We measure the system performance in terms of the number of winning rounds within each learning period of 100 rounds.

With the initial set of domain knowledge, R-FALCONBot and TD-FALCONBot can already defeat almost all the sample robots provided by the Robocode platform, with the exceptions of *SpinBot* and *Walls*. As shown in Figure 4, we see that the number of winning rounds of TD-FALCONBot against *SpinBot* increases to more than 80 right after 100 rounds. On the other hand, R-FALCONBot achieves a success rate of over 60% after 200 rounds of learning. These results show that the performance of both TD-FALCONBot and R-FALCONBot consistently improves upon learning. In addition, the learning efficacy of TD-FALCONBot is significantly stronger than that of R-FALCONBot. Referring to Figure 5, we can see that, against *Walls*, the performance of TD-FALCONBot improves steadily and achieves a winning rate of around 50%. This implies that its performance is roughly comparable to *Walls* after learning. In comparison, R-FALCONBot achieves a much lower performance.

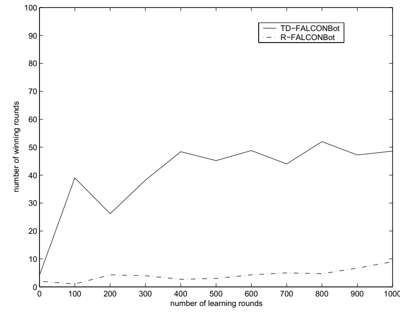

Figure 5. The performance of FALCON-X against *Walls*.

VIII. Conclusion

We have presented a cognitive model by merging the ART-R cognitive architecture with fusion ART neural network. We have illustrated how FALCON-X can be used to learn procedural knowledge and how the procedural knowledge can be integrated with declarative knowledge encoded as pre-existing long-term memories using composite neurons for problem solving. The proposed integration of ACT-R and ART is interesting as the parallel memory processes and production rule firing of ACT-R can now be regulated by the neurally plausible mechanism of ART. In addition, the competitive coding processes of ART enable a natural and powerful means for learning of associations across the five memory modules.

References

[1] T. D. Kelley, "Symbolic and sub-symbolic representations in computational models of human cognition: What can be learned from biology?" *Theory and Psychology*, vol. 13(6), pp. 847–860, 2003.

[2] J. E. Laird, A. Newell, and P. S. Rosenbloom, "Soar: An architecture for general intelligence," *Artificial Intelligence*, vol. 33, pp. 1–64, 1987.

[3] J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, and Y. Qin, "An intergrated theory of the mind," *Psychological Review*, vol. 111, pp. 1036–1060, 2004.

[4] S. Haykin, *Neural Network: A Comprehensive Foundation*. Prentice Hall, 1999.

[5] A.-H. Tan, G. A. Carpenter, and S. Grossberg, "Intelligence through interaction: Towards a unified theory for learning," in *Proceedings, ISNN, LNCS4491*, 2007, pp. 1098–1107.

[6] G. A. Carpenter and S. Grossberg, "Adaptive Resonance Theory," in *The Handbook of Brain Theory and neural Networks*. MIT Press, 2003, pp. 87–90.

[7] A.-H. Tan, "FALCON: A fusion architecture for learning, cognition, and navigation," in *Proceedings, IJCNN*, 2004, pp. 3297–3302.

[8] A.-H. Tan, N. Lu, and D. Xiao, "Integrating temporal difference methods and self-organizing neural networks for reinforcement learning with delayed evaluative feedback," *IEEE Transactions on Neural Networks*, vol. 9, no. 2, pp. 230–244, 2008.

[9] A.-H. Tan, "Direct code access in self-organizing neural networks for reinforcement learning," in *Proceedings, IJCAI*, 2007, pp. 1071–1076.