Learning sequential control in a Neural Blackboard Architecture for in situ concept reasoning

Frank van der Velde
University of Twente, CPE-CTIT; IOP, Leiden University, The Netherlands
f.vandervelde@utwente.nl

Abstract. Simulations are presented and discussed of learning sequential control in a Neural Blackboard Architecture (NBA) for in situ concept-based reasoning. Sequential control is learned in a reservoir network, consisting of columns with neural circuits. This allows the reservoir to control the dynamics of processing by responding to information given by questions and the activations in the NBA. The in situ nature of concept representation directly influences the reasoning process and learning in the architecture.

Keywords. Learning • Neural blackboard architecture • In situ concepts • Reasoning • Reservoir • Wilson-Cowan dynamics

1 Introduction

Neural representation and processing of symbol-like structures as presented here takes its inspiration from the observation that concept representations in the brain are 'in situ', in line with the neural assemblies as proposed by Hebb ([1]). Neural assemblies, as Hebb argued, will develop over time when neurons that process information about, for example, a concept become interconnected. Such concept representations could be distributed, but parts of the assembly could also consist of more local representations. They will generally consist of neurons involved in processing information but also of neurons involved in actions. In this way, in situ concepts representations are always grounded in perception or action, so that the neural connection structure underlying a concept is determined by both its 'incoming' (perception-based) connections and its 'outgoing' (action-generating) connections [2].

The in situ nature of neuronal concept representations imposes constraints on the way they can be combined to represent and process more complex forms of information. However, complex conceptual structures (e.g., sentences with hierarchical structures) can be represented and processed when the neural assemblies underlying in situ concept representation are embedded in a 'Neural Blackboard Architecture' or NBA [3].

In general, "in-situ concept-based computing" would be achieved by embedding in situ concepts in several NBAs, each needed for a specific form of processing. Blackboards are also used in computer domains, e.g., to store arbitrary forms of (symbolic) information. NBAs as intended here, however, are fundamentally different. They possess structural information, (e.g., related to sentence structures as in [3]), and are implemented with dedicated structures (e.g., neural circuits, as in the brain). In this way they cannot store arbitrary information, but they can process specific forms of (high-level cognitive) information, e.g., by the interactions between the structured representations in the blackboards. Fig 1. illustrates that there will be NBAs for sentence

structures, phonological structures (e.g., forming new words), sequential structures based on in situ concept representations, relation structures as used in reasoning, and potentially other NBAs (blackboards) as well. The interaction between these NBAs derives from the in situ concept representations they share. For example, a word (concept) would be shared by the sentence, phonology, sequential and relation blackboards (and potentially more). In turn, concepts are also related to each other in several "feature spaces", which can influence processing in the NBAs as well.

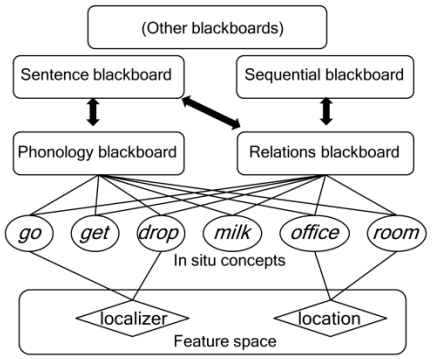

Figure 1. Overview of in situ concept-based computing with Neural Blackboard Architectures (NBAs).

The NBA in [3] can account for sentence structure and processing, including sentence learning [4] and examples of ambiguity resolution and garden path modelling [5]. A recent extension of the NBA approach to symbol-like processing is the NBA for (basic forms of) reasoning presented in [6], which can be used in basic reasoning processes, such as BABI reasoning tasks as presented in [7].

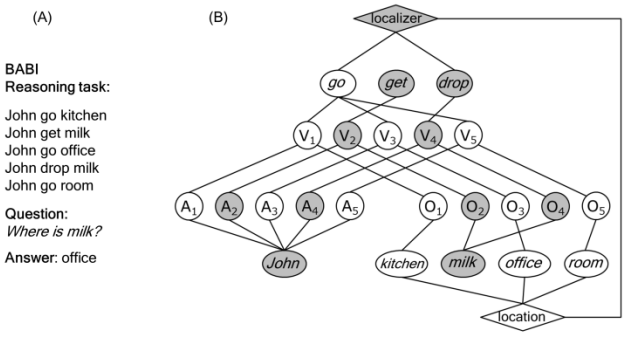

Figure 2. (A) BABI task for reasoning (after [7]). (B) Propositions in a relation blackboard (A = agent, O = object). Grey nodes activated by the question Where is milk?. <localizer> and <location> are concept features, belonging to feature space.

Fig. 2A presents an example of a BABI reasoning task. A set of relations is given and a question has to be answered on the basis of these relations (propositions). So, the question *Where is milk?* can be answered by first retrieving *John drop milk* (providing a

location for *milk*) and then retrieving *John go office* as the last location of *John* before *John drop milk*. This would provide *office* as the location of *milk*. Fig. 2B presents the representations of these relations in the relation NBA, and the representations activated by the question *Where is milk?* are illustrated in grey.

Here, a first set of simulations will be presented and discussed that address the way NBAs can learn to perform reasoning processes of this kind. This paper focusses on the fundamental issue of whether questions can retrieve information needed for reasoning in NBAs. Simulations of other aspects of the NBAs and reasoning process are outside the scope of this paper, or under development (e.g., the selection of the more recent activated relation in the sequence blackboard, which is assumed here). A further description of how BABI tasks can be solved in NBAs is presented in [6].

2 Learning control of reasoning in an NBA

A key element of in situ concept processing in NBAs is that the information provided by a question is directly used in the activation of the concepts involved. In Fig. 2B, for example, the question *Where is milk*? directly activates the in situ concepts *is* and *milk*. In turn, they will activate their features (e.g. <localizer> for *is*) in feature space. Due to the activation of the in situ concept *milk*, all relations in Fig. 2A in which *milk* occurs can be activated as well, because they share the same in situ concept (*milk*). So, *John drop milk* and *John get milk* can be directly activated in this way.

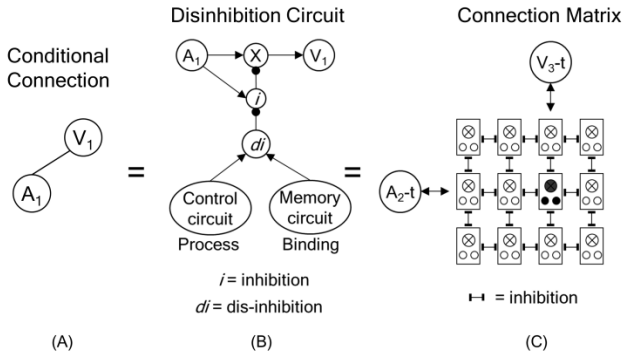

Figure 3. (A). Conditional connection in Fig. 2B. (B) Conditional connections with disinhibition circuits . (C) A connection matrix of conditional connections for binding.

Activation of concept structures (here relations) in an NBA depends on the nature of the structure representations, and the control and binding circuits in the NBA. In [3] an extensive discussion of these is given for the sentence NBA, but they are the same in the relation NBA of Fig. 2. Fig. 3 illustrates a basic overview. Each connection in Fig. 2B represents a conditional connection. These connections can operate only when they are activated. In this way, relations can be represented and processed in NBAs (instead of just associations as with unconditional connections). A conditional connection can be implemented with a disinhibition circuit, as illustrated in Fig. 3B. The circuit can be activated by a control circuit or by a memory circuit. The latter produces (temporal)

bindings in the NBA. The process of binding and (re)activation is determined by the control circuits.

For example, the binding and (re)activation of *John drop milk* in Fig. 2B proceeds as follows. First, *John* is activated. To achieve the representation of *John* as agent, *John* and an (arbitrary) Agent node in the NBA (e.g., A4) are bound by activating the memory circuit between them ([3]). This results in the binding of *John* and A4. In the same way, *drop* binds to V4 and *milk* as object to O4. Then, a control circuit activates the conditional connections between A4 and V4 to represent *John drop*. To achieve a binding between arbitrary A and O nodes, all A and O nodes are connected to each other in a connection matrix, as illustrated in Fig. 3C. A connection matrix is a matrix of circuits ('columns') that regulate the binding process. To bind A4 with V4, the control circuit activates the conditional connections between A4 and V4 and their corresponding column in the connection matrix. This, in turn results in the activation of the memory circuit in that column. As long as this circuit remains active (with sustained or 'delay' activation), A4 and V4 are bound, even when they themselves are deactivated again. This binding process is used for all bindings in the NBA.

The relation *John drop milk* can be reactivated by activating one of the in situ concepts involved and the required conditional connections. For example, the question *Where is milk?* activates *milk*. Because of their binding with *milk*, O4 (and O2) are activated as well. By activating the conditional connections for Object between Object and Verb nodes, O4 activates V4, which activates *drop*. A4 can be activated by activating (enabling) the Agent conditional connections between Verb nodes and Agent nodes. This results in the reactivation of *John drop milk*. In Fig. 2B, this process also results in the reactivation of *John get milk*, because these binding are active as well (i.e., this relation is also stored in the NBA). A distinction between these two relations can be made by a sequential ordering in a sequence blackboard (assumed here, e.g., see [6]).

The reactivation of stored relations is crucial for a reasoning process as illustrated in Fig. 2A. For example, *Where is milk?* can be answered by first activating *John drop milk* and *John get milk*, using the activation of *milk* by the question. Then by selecting *John drop milk* as the more recent relation. In this case, *drop* indicates a location for *milk*, given by the <localizer> feature of *drop* in Fig. 2. This would initiate a second question *Where is agent-drop?*, i.e., *Where is John?* here. This question would activate *John go office*, selected as the most recent location of *John*. This produces *office* as the location of *milk*. In other words, new questions in the reasoning process derive from activations in the NBA initiated by previous questions.

Hence, the activations initiated by the (first) question and the resulting interactions with the blackboard determine the process of answering questions like *Where is milk*? Here, this interaction is simulated by using a control network to recognize the (first) question and initiate the required interaction (e.g., further questions) with the blackboard. The control network consists of a form of reservoir computing (e.g., [8]).

2.1 Reservoir for control

A reservoir is a set of neurons or 'nodes' that are sparsely interconnected in a random (fixed) fashion. Also, the nodes are connected (randomly) to input neurons providing

external information. A reservoir can learn to activate specific output neurons in response to a sequence presented to it. In this way, they can learn to process and recognize sequential information [8].

Hinaut and Dominey [9] used a reservoir to recognize sets of sentences. However, in a reservoir the nodes activate each other based on their (node) activation dynamics. When a sequence with a specific sequential dynamics is presented to the reservoir, it can learn to 'resonate' to the external dynamics because that is predictable [8]. This is typically more difficult for language, because timing differences between words can vary. In [9] this was solved by adjusting the dynamics of the reservoir nodes to regular word presentation timing. Here, however, the sequence to be learned is not only determined by the presented question but also by the interactions with and within the neural blackboard, which could vary given the amount of information stored and processed in it. So, a direct adjustment of timing of node activation is not possible. Therefore, the reservoir presented and simulated here is more complex.

Fig. 4 illustrates that the reservoir consists of columns, which in turn consist of neural circuits. The sequential activation as produced by the reservoir is given by the 'sequence' (S) nodes. They are randomly and sparsely connected to each other, in a fixed manner. However, S nodes do not directly activate each other. Instead, an active node Si will activate a 'delay' population in the column of a node Sj to which it is connected. The delay population remains active (unless inhibited). It activates Sj but also an inhibitory node i, which inhibits Sj. In this way, the timing of the sequence in the reservoir is under control. Sj can be activated only when node i is inhibited. As indicated, this will happen when an 'Item' node activates another inhibitory node that inhibits i. When this happens, Sj will be activated and it will in turn activate other S nodes in the sequence in the same manner.

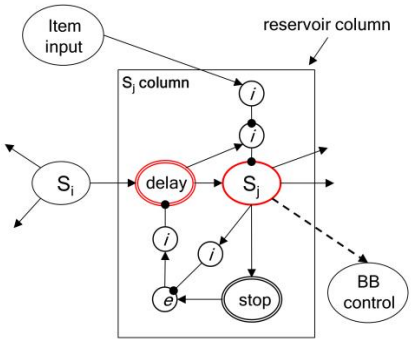

Figure 4. Reservoir of columns. Circles and ovals represent neural populations. Double lined ovals remain active (sustained or delay activity). I = inhibition, e = excitation. S = sequence. Dashed connections are modifiable by learning (e.g. LTP).

Item nodes represent the external inputs to the reservoir. Here, they consist of sentence information and/or information derived from the blackboard. Hence, the sequential dynamics produced by the reservoir is under control by the information given by the question and the interactions produced in the blackboard.

The aim of the reasoning NBA is to simulate and learn reasoning in this way. That is, the reservoir will learn to recognize sequential information given by the question and by activations in the blackboard to initiate new activations in the blackboard, until the question can be answered. Learning can be achieved by the adaptive connections between S nodes and nodes that control the binding and (re)activation process in the blackboard. So, Sj could learn to activate a specific control in the blackboard, such as the control to activate the Agent or Object conditional connections.

Here, basic aspects of this process are simulated for answering the questions *Where is John* and *Where is milk?* in the task illustrated in Fig. 2A.

3 Simulation of reservoir activity

All the populations in the NBA are modelled with Wilson Cowan population dynamics [10]. Each population consist of groups of interacting excitatory (E) in inhibitory (I) neurons. The behavior of the E and I groups are each modeled with an ODE at population level. Both ODEs interact and they receive input from outside. A working memory (or delay) population consists of two interacting populations, say A and B. The output results from A. The role of B is to sustain the activity by its interaction with A. It is assumed that B has a lower activation maximum than other populations. This results in a reduced activity of a working memory population when it relies on delay activity only (i.e., does not receive input). The E neurons are used for output to other populations. Populations are excitatory when their output connection has a positive weight. They are inhibitory when their output connection has a negative weight. All populations operate with the same parameters and all weights are the same (as in [5]). The behavior of the populations is simulated with a fourth order Runge Kutta numerical integration (with h = 0.1).

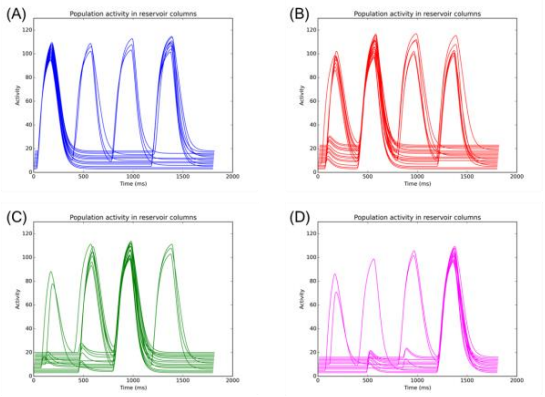

Figure 5. Activations of reservoir S nodes. (A) Where. (B) <localizer>. (C) noun. (D) Agent.

The question *Where is John?* is presented to the reservoir word by word. However, for the reservoir word type and feature information is used. Words like *is* and *go* are represented as <localizer>, words like *John* and *milk* are presented as nouns. The specific words in the questions are used to activate their (in situ) representations in the

backboard. So, the active concept *John* activates the nodes A4 and A2 in the blackboard. This provides information that *John* is the agent in the relations stored in the blackboard. In turn, this information can be used to learn that the blackboard should provide the object information related to (bound to) *John is*.

The reservoir can learn to do this by recognizing the item sequence *Where* - <*localizer*> - *noun* – *Agent* and producing the activation of the Object conditional connections in the blackboard. This will produce the activations of *John go kitchen*, *John go office*, and *John go room*, from which *John go room* can be selected as the most recent, using the sequential blackboard in Fig. 1 (see [6]).

Fig. 5 presents the activations of sets of S nodes in a reservoir of 750 columns with sparse connectivity in response to the item sequence *Where* - <*localizer*> - *noun* – *Agent*. The first three items (*Where* - <*localizer*> - *noun*) are based on the question, the fourth item (*Agent*) is derived from the blackboard. Each color represents a different set of S nodes in the reservoir. The blue set are S nodes that are initially activated by start nodes (not shown), that respond to the start of a question. They also respond to the item *Where* (specifically). However, as the figure shows, some of these nodes also respond to the other items in the item sequence presented to the reservoir.

The red S nodes are activated by the blue S nodes (columns) and also specifically respond to the second item <*localizer*>. But some of them also respond to the other items at other steps in the sequence. Likewise, the green S nodes specifically respond to the third item because they are activated by the combination of the active red S nodes and the item *noun*. Again, however, some of them respond to other items at other sequence steps as well. The magenta S nodes specifically respond to the fourth item, because they are activated by the green S nodes and the item *Agent*.

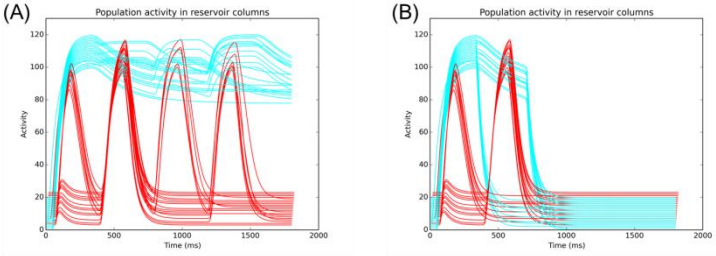

Figure 6. (A). Activation of the red nodes in Fig 5 and the delay populations in their colums (cyan). (B). The same activations after activation stop.

The active magenta S nodes could be used to learn that the blackboard should activate the Object conditional connection, because *John* (*noun*) is an Agent and the question asks for an object bound to that agent (and a localizer word). Here, that would be possible when the magenta S nodes dominate the activation in the reservoir at the fourth step. However, a substantial amount of other nodes are active as well. But reservoir nodes can learn specific responses based on their distributed activation ([9].

But when the activation of S nodes is more specific, such learning could be achieved by direct adjustments of neural weights (the dashed connections in Fig. 4), which would allow rapid forms of learning. To achieve more specific activation of S nodes it is

important to look more closely at the activations produced in the columns.

Fig. 6A shows the activations of the red S nodes and the delay populations in their columns. The delay populations remain active. So, when a new item is presented, some of the red S nodes respond to that item because it is connected to their column. This accounts for the repeated activation of some of the S nodes of all color in Fig. 5.

Delay activity can be stopped, however, by a neural circuit illustrated in Fig. 4. To this end, the S node is connected to a 'stop' population (consisting of sustained activation). When Sj is active it will activate this population. But it will also activate an inhibitory neuron that prevents the effect of the stop population. The stop population can inhibit the delay population only when Sj is deactivated, and it continues to do so as long as it is active. The Sj node is deactivated when the Item node is deactivated, which will occur with the presentation of a new item in the sequence. It that case, the delay population ensures the deactivation of the Sj node, which in turn ensures the deactivation of the delay population.

Fig. 6B shows the effect of stopping the delay activation. After the red S nodes are deactivated, the delay activation is deactivated as well. This prevents further activation of the red S nodes. Yet, some of the red S nodes are active at the first step, even though they are not activated by the start nodes. This activation results from the rapid activation of their delay nodes by the blue S nodes (Figure 5) and the fact that these red S nodes also respond to the first item (*Where*). However, due to their activation at the first step, these red S nodes are no longer activated at the second step (unlike in Fig. 5) because their delay populations are deactivated. So, the S nodes in the second step are now specifically active for the second step in the sequence. Similarly, the first step in the sequence is now given by the blue S nodes and the red S nodes active at that step. In this way, specific sets of S nodes are activated at specific steps in the sequence. This allows for rapid learning by direct synaptic modification.

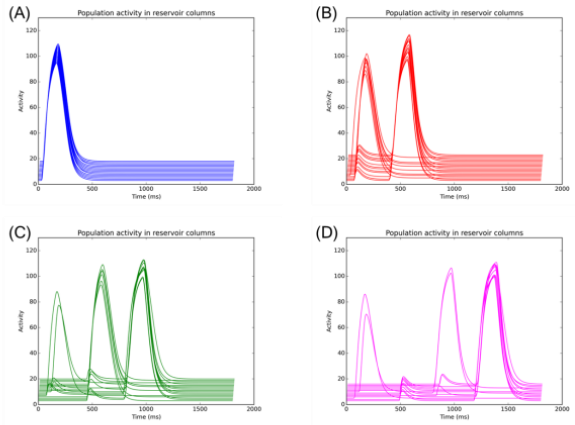

Figure 7. Activations of reservoir S nodes, with Stop of activation. (A) Where. (B) <localizer>. (C) noun. (D) Agent.

Fig. 7 shows the results for all S nodes presented in Fig. 5. The colored nodes are specifically active at the step in the sequence that is related to the item they represent.

Activation after that step is prevented. In some cases, some of the S nodes are active before that step. In that case, however, they will not be activated again. So, at each step a specific set of S nodes will be active that uniquely represents the item of that step. Specifically, the magenta S nodes can learn to produce the activation of Object conditional connections by direct synaptic modification.

3.1 Learning more complex control

The question *Where is John?* generates a direct answer by the reactivation of *John go room*. The question *Where is milk?*, however, does not directly produce an answer in this way. To answer this question, a second representation needs to be activated after *John drop milk*. In turn, this requires a longer and more complex sequential sequence to be learned by the reservoir and a longer interaction process with the blackboard. A reservoir can indeed learn such a process and interaction with the blackboard to produce the answer. Here, however, only a few aspects of that can be illustrated.

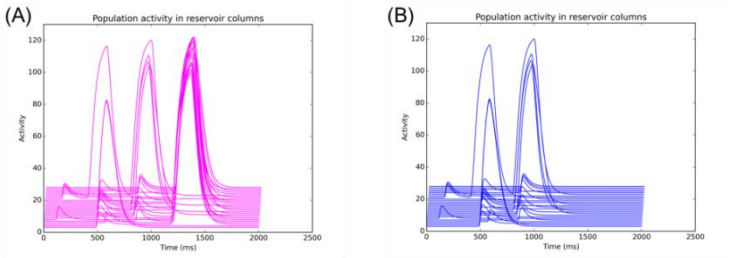

Figure 8. (A). Activation of S nodes in the fourth step in *Where is milk?* (B) Activation of the same S nodes with the question *Where is John*?

First, the question *Where is milk?* generates Object instead of Agent as response from the blackboard in the fourth step. Yet, the first three steps are the same as with *Where is John?*. Fig. 8A illustrates the activation of S nodes in the fourth step of *Where is milk?* These nodes are thus activated by the S nodes active at the third step and by the item *Object*, derived from the activation of O4 and O2 in the blackboard (Fig. 3). Fig. 8B illustrates the activation of the same S nodes when, at the fourth step, the item *Agent* is presented. It is clear that the S nodes selectively respond to the item *Object*, instead of *Agent*. This allows them to learn to activate the Agent conditional connections in the blackboard, to reactivate the relation *John drop milk*.

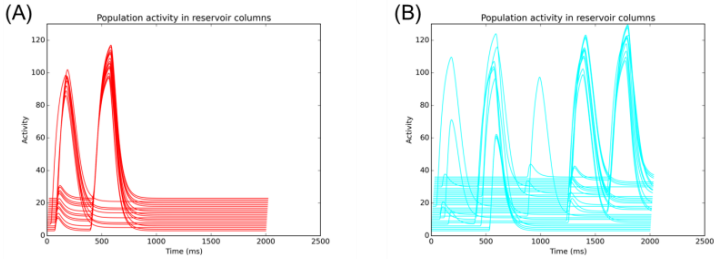

Figure 9. (A). Activation of S nodes in the second step in *Where is milk?* (B) Activation of the S nodes in the fifth step of that question.

Second, the question *Where is milk*? gives location information (*is*) at step two, but it also requires location information at step five in the process, to retrieve the location of *John* after *John drop milk* has been reactivated. Fig. 9 illustrates the activation in the reservoir related to the same information at different steps.

The red S nodes in Fig. 9A respond to the first activation of <*localizer*> in the process. The cyan S nodes in Fig. 9B respond to the second activation of <*localizer*> in the process. That is, these S nodes would all be activated by the active S nodes in the fourth step and by the item <*localizer*>. Some of them are already activated in some of the previous steps, however, which prevents their activation in the fifth step. Hence, the control (stop) of activation in the reservoir results in a set of S nodes that selectively respond to the item <*localizer*> in the fifth step, irrespective of the presence of that item in a previous step. Such a selective response to repeated activation of item information will be crucial for the success of learning reasoning in a neural reasoning architecture as presented here.

5 Conclusions

Simulations of the learning of sequential control in a neural blackboard architecture (NBA) for reasoning were presented. The NBA is based on in situ concept representation. This entails that concepts are always represented by the same underlying neural assemblies (although different parts of them might be activated at different occasions). The in situ nature of concepts imposes constraints on the ways they can be used to represent and process complex forms of conceptual information, as found in language or reasoning.

But it also provides distinctive benefits. First, in situ concepts are content addressable. Thus, as illustrated here, the concept and other information given by a question will directly select the related information stored in the neural blackboard by reactivating the in situ concept representations. This, in turn, can guide the reasoning process by interactions between the neural blackboard and a reservoir network that selectively responds to sequential information.

The interaction between neural blackboards and control networks (e.g., reservoirs) also offers new forms of learning, in which the distinction between structured neural blackboards, control circuits and content addressable activation by in situ concepts strongly reduces the number of contingencies that have to be learned.

Furthermore, in situ representations are not moved or copied. And, as noted, they are content addressable. Therefore, neural blackboard architectures of reasoning and other forms of (high-level) cognitive processing with in situ representations would be very suitable for implementation in (e.g., new) forms of parallel and power reduced hardware.

Acknowledgements

The work of the author was funded by the project ConCreTe. The project ConCreTe acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET grant number 611733.

References

1. Hebb, D. O. (1949). *The organisation of behaviour*. New York: Wiley.
2 van der Velde, F (2015). Communication, concepts and grounding. *Neural networks, 62* , 112 - 117.
3 van der Velde, F. and de Kamps, M. (2006). Neural blackboard architectures of combinatorial structures in cognition. *Behavioral and Brain Sciences, 29*, 37-70.
4. van der Velde, F & de Kamps, M. (2010). Learning of control in a neural architecture of grounded language processing. *Cognitive Systems Research, 11*, 93–107.
5 van der Velde, F., and de Kamps, M. (2015). Combinatorial structures and processing in neural blackboard architectures. In *Proceedings of the Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches (CoCo@NIPS 2015)*, eds T. R. Besold, A. d'Avila Garcez, G. F. Marcus, and R. Miikkulainen (Montreal). CEUR Workshop Proceedings (pp. 1-9).
6. van der Velde, F. (2016). Concepts and relations in neurally inspired in situ concept-based computing. *Frontiers in Neurorobotics. 10:4*. doi: 10.3389/fnbot.2016.00004
7 Bordes, A., Weston, J., Chopra, S., Mikolov, T., Joulin, A., Rush, S., & Bottou, L. (2015). *Artificial Tasks for Artificial Intelligence.* Facebook AI Research. ICLR – San Diego – May 7, 2015. http://www.iclr.cc/lib/exe/fetch.php?media=iclr2015:abordes-iclr2015.pdf
8. Jaeger, H and Haas, H. (2004). Harnessing nonlinearity: Predicting chaotic systems and saving energy in wireless communication. *Science*, *304*, 78-80.
9. Hinaut X, Dominey PF (2013) Real-Time Parallel Processing of Grammatical Structure in the Fronto-Striatal System: A Recurrent Network Simulation Study Using Reservoir Computing. *PLoS ONE 8(2):* e52946. doi:10.1371/journal.pone.0052946
10 Wilson HR, Cowan JD (1972) Excitatory and inhibitory interactions in localized populations of model neurons. *Biophysical Journal, 12*, 1–24