Challenging the Intuition about Memory and Computation in Theories of Cognition

Jochen Kerdels and Gabriele Peters
FernUniversität in Hagen, University of Hagen, Human-Computer Interaction,
Faculty of Mathematics and Computer Science, Universitätsstrasse 1, D-58097 Hagen, Germany

Keywords: Memory, Computation, Theories of Cognition, Cognitive Architectures, Neurobiology.

Abstract: In this position paper we argue that the concepts of memory and computation as they are commonly used in theories of cognition are strongly influenced by our intuitive understanding of the corresponding concepts in contemporary computer systems, leading to an implicit loss of biological plausibility. To support our argument we provide an alternative perspective on memory and computation that allows a closer comparison of the capabilities of computer programs running on computer systems and neurobiological systems showing that computer programs exhibit a computational flexibility that is difficult to reconcile with neurobiological constraints. We end this paper by offering a neurobiologically plausible perspective on memory that views memory as a dynamic, distributed process that is an intrinsic part of a neurobiological network that integrates information, e.g., sensory information.

1 INTRODUCTION

Theories of cognition seek to present models of the human mind that enlighten our understanding of a wide spectrum of cognitive processes including integration of sensory information, generation of behavior, and processes of high-level thought. Cognitive architectures in particular aim at providing computational instantiations of such models that facilitate the exploration of dynamic properties and behaviors. Prominent examples of such architectures are ACT-R (Anderson, 1996), SOAR (Laird et al., 1987; Laird, 2008), CLARION (Sun, 2007), LIDA (Franklin et al., 2013), or the highly influential Global Workspace Theory (GWT) by Bernard Baars (Baars, 2017).

In this paper we would like to discuss two basic assumptions that commonly underlie theories of cognition that seem to be true almost trivially but may impact the neurobiological plausibility of these theories severely. The first assumption concerns the concept of memory, the second assumption concerns the related concept of computation. We will argue that the concept of *memory* as it is commonly used in theories of cognition is strongly influenced by our understanding of memory as it is used in typical computer systems with von Neumann architecture. It is assumed that memory is a kind of container that allows to store pieces of information and to retrieve these pieces again later on. Similarly, the second assumption that we would like to challenge concerns the concept of *computation*. It is the idea that pieces of information present in such a memory can be used by different parts of a cognitive system implying an encoding of information whose meaning is shared by these different parts. We hope to show that both assumptions, when scrutinized, are difficult to reconcile with the properties and constraints inherent to a plausible neurobiological substrate. As an alternative, we present a different perspective that views memory as a process that is an intrinsic, distributed part of a neurobiological system.

The following section highlights typical ideas of memory and its usage in a number of cognitive architectures, and relates these ideas to a common and widespread intuition about computation. Section 3 elaborates on this intuition and proposes an alternative perspective on computation that views computer programs as transformation networks, which enable a closer comparison of the computational capabilities of general purpose computer systems and neurobiological systems. Section 4 then proposes an alternative perspective on the structure and function of memory that emphasizes its integrative and distributed characteristics. Finally, section 5 concludes this position paper.

Kerdels, J. and Peters, G.
Challenging the Intuition about Memory and Computation in Theories of Cognition.
DOI: 10.5220/0008493605220527
In *Proceedings of the 11th International Joint Conference on Computational Intelligence (IJCCI 2019)*, pages 522-527
ISBN: 978-989-758-384-1

2 MEMORY IN THEORIES OF COGNITION

Memory is a central part of many theories of cognition where it is used for multiple purposes. Amongst other uses, it can serve as a communication hub that allows to dynamically route information between different parts of the cognitive system and establish a kind of global context (seen as analog to “short term memory”), it can serve as a storage of past experiences (“episodic memory”) and knowledge about the world (“declarative memory”), or it can serve as a mechanism that facilitates the acquisition and reproduction of behavioral patterns (“procedural memory”). In all these cases, either explicitly or implicitly, memory is typically seen as a container that can store and reproduce signal patterns much like the memory in a computer system that stores and reproduces bit patterns. In particular, *pieces of information* are seen as self-contained entities that can “move around” and exist at different locations within the computational system. This notion is reflected by, e.g., John Laird (SOAR) as he outlines what a cognitive architecture consists of in his view (Laird, 2008):

> *A cognitive architecture consists of:*
>
> - *memories for storing knowledge*
> - *processing units that extract, select, combine, and store knowledge*
> - *languages for representing the knowledge that is stored and processed*
>
> *[..] Cognitive architectures must embody strong hypotheses about the building blocks of cognition that are shared by all tasks, and how different types of knowledge are learned, encoded, and used, making a cognitive architecture a software implementation of a general theory of intelligence.*

Another example that highlights this perspective is given by John Anderson (ACT-R) in the context of describing how knowledge about the environment is processed in their system (Anderson, 1996):

> *We have only developed our ideas about environmental encodings of knowledge with respect to the visual modality. In this area, it is assumed that the perceptual system has parsed the visual array into objects and has associated a set of features with each object. ACT-R can move its attention over the visual array and recognize objects. [..] Features within the spotlight can be synthesized into recognized objects. Once synthesized, the objects are then available as chunks in ACT’s working memory for further processing.*

Finally, a good example of how memory is utilized as a central mechanism to coordinate and control the flow of information in a cognitive system is given by Bernard Baars as he describes the core idea of his Global Workspace Theory (Baars, 2017):

> *Global Workspace Theory (GWT) suggests that the brain has a fleeting integrative capacity that enables access between functions that are otherwise separate. This makes sense in a brain that is viewed as a massive parallel set of highly specialized neuronal processors. In such a system coordination and control may take place by way of such a central information exchange, allowing some specialized processors – such as sensory regions in cortex – to distribute information to the system as a whole. This solution also works in large-scale computer architectures, which show typical “limited capacity” behavior when information flows by way of a global workspace.*

These quotes illustrate how the concept of memory in cognitive architectures is closely aligned with the corresponding concept in contemporary computer systems. Linked to this understanding of memory is a matching intuition about computation. In its most basic form this intuition about computation consists in the idea of executing a list of basic operations that each take some data as input and produce some data as output that affects the subsequent steps of the computation in a deterministic way, e.g., by generating new input data, controlling the program flow, or causing desired side effects. For many, this intuition about computation matches their introspective understanding of the apparent sequential nature of high-level, linguistic thought or the planning and control of deliberate actions necessary to perform a complex task. One important aspect of this analogy between computation and high-level mental processes is the sense of agency that the latter imply. *We* think thoughts. *We* perform actions. *The computer*[1] processes data. This implicitly assumed existence of agency makes thoughts and data alike into objects that are acted upon. This object quality in turn implies the existence of such objects as coherent entities in some medium. In case of data this medium appears to be memory.

This basic intuition about computation is a powerful tool, an abstraction, to reason about and write complex computer programs, and it is tempting to use it to interpret and model the function of neurobiological information processing systems as well. However, we argue that the latter use of this abstraction is counterproductive and leads to misleading analogies

[1]or its central processing unit (CPU)

between computer and brain, especially regarding the purpose and function of memory in both systems. To elucidate this aspect the next section provides an alternative intuition about memory and computation that allows for a more direct comparison of computer and neurobiological systems. It will highlight fundamental differences of both systems regarding their respective capabilities to process information.

3 COMPUTER PROGRAMS AS TRANSFORMATION NETWORKS

The intuition about computation outlined above is supported and reinforced by modern programming languages as they facilitate the definition of complex data types and their instantiation as objects that seem to exist as such entities in memory. In addition, these objects are commonly described in a way that make it seem as if these objects exhibit forms of agency by possessing a local, encapsulated state, and being responsible for a specific set of subproblems. This useful abstraction allows a programmer to describe the solution to a computational problem in a way that is both meaningful to other programmers and herself as well as "understandable" by the computer system. This "understanding", however, is for the most part just an illusion. When a program is actually loaded into memory and executed, the resulting bit patterns that are stored in memory are by themselves meaningless. Their meaning is only encoded in the program that reads and interprets these patterns, and much of this encoded meaning is already stripped away when the high-level program is translated into machine code. More precisely, most of the meaning attributed to certain high-level program structures by a programmer are not part of the program that will actually run on the computer. In that sense, the "actual meaning" of a bit pattern in memory is as elusive to the computer system as an evolved, instinctive behavior might be to an insect. Yet, the implicit sense of agency mentioned above is suggestive of locating the meaning of a bit pattern intuitively somewhere within the machine rather than within the programmer who is describing and interpreting the machine's behavior.

To gain an alternative perspective that avoids this intuitive, but – in our view – misleading interpretation of a program we take a closer look at the idea of computation as executing a list of basic operations. In essence, such a basic operation can be understood as a prescription of how a particular input pattern is transformed into a new output pattern given a limited set of hardware resources. Using such basic operations over and over again to build a complex web of input output transformations is what a computer program does. The "glue" that enables the computer program to form this network of transformations is memory.

When a computer program executes it orchestrates a constant circulation of bit patterns between memory and CPU. By doing so the limited hardware resources of the CPU are reused over and over again to successively perform all the input output transformations in the network of transformations that is defined by the program. Hence, every computer program can be seen as the emulation of a vastly more complex, special purpose CPU that would implement every input output transformation defined in the program in hardware. In that sense, every memory retrieval within a computer program - even retrieval from distant memories like a hard disk or some network resource - corresponds to a dynamic reconfiguration and/or expansion of this virtual, special purpose CPU.

In this alternative view on computation memory serves as a pattern buffer between transformations that endows the resulting, virtual transformation network with a number of capabilities that would be challenging or impossible to implement in a real, physical network in which the hardware performing each transformation is connected directly. In particular, the use of memory enables the network to operate independent of timing constraints, i.e., the output of a transformation can be buffered for an arbitrary amount of time before it is processed by further transformations. Typical uses of this capability include waiting for the results of multiple transformations before proceeding, combining the output of a transformation with its past outputs, or even distributing a computation over multiple devices. Furthermore, memory enables the virtual transformation network to have arbitrary connectivity patterns that are not bound to physical constraints like limited signal speed, limited fan-out and fan-in of the transformation hardware, or limited physical space. Together, these capabilities of the virtual transformation network reflect the flexibility and power of computer programs running on contemporary computer systems. The key to this flexibility and power is the use of memory as the central interface mechanism for this transformation network. We argue that this view on computation provides a more accurate and simpler picture of the function and purpose of memory than the high-level perspective provided by modern programming languages where the apparent program structure diffuses into the structure of the virtual transformation network when the program is translated into machine code.

Using this alternative view on computation allows a more direct comparison to the processing of information in neurobiological systems. From an evolutionary perspective neurobiological systems developed to solve specific computational problems, e.g., to control muscles or to process the information coming from a sensory system. As such, the respective networks of neurons or groups of neurons could be seen as specialised transformation networks similar to the transformation networks described above. However, instead of being emulated and virtual, these natural transformation networks are actually implemented within the neurobiological substrate removing the necessity for memory as a "transformation glue" while at the same time adding a number of constraints regarding possible connection patterns and types of transformations. In particular, an on-the-fly reconfiguration and expansion of the neurobiological transformation network is not possible in general. Instead, the basic structure of the network emerges during the development of the organism and is controlled by, e.g., periods of cell proliferation, cell migration along cellular support structures, or guidance by short and long range chemical gradients (Squire et al., 2008). The variability inherent in this self-organized formation limits the extent with which the solution to a specific computational problem, e.g., a behavioral pattern, can be hardwired into the network structure and adapted over time. To mitigate this constraint neurobiological networks show forms of plasticity that facilitate fast changes to the network by adapting the response of individual neurons or local groups of neurons to a given input signal. As a consequence, the corresponding input output transformation implemented by those neurons changes accordingly. However, subsequent transformations are not informed about this change but have to adapt themselves as well if necessary and propagate the change further. This means that from a global perspective the way by which signals are encoded and processed within the transformation network is only known locally and remains in a constant flux.

This dynamic change of how information is encoded and processed by different parts of the transformation network conflicts with an idea of memory that sees memory as a container that stores patterns of information for later use since the encoding of that information might have changed while it was stored. Such an outdated encoding would then become unintelligible for the network. Similarly, the idea of memory as a means to coordinate and control the flow of information relies as well on a consistent encoding, which is not guaranteed when local neuroplasticity is taken into account. Despite these doubts regarding the biological plausibility of a container-like memory one might argue that neurobiological systems clearly do have the ability to remember, e.g., past experiences and therefore must have some form of memory. To address this point we will outline our view on a neurobiologically plausible memory in the next section.

4 MEMORY AS A PROCESS

In the previous section we described how local neuroplasticity continually changes the way how neurons or local groups of neurons respond to their inputs and thus encode these inputs differently for neurons downstream in the network. A memory system that is based on storing and recreating patterns of information is not well suited to cope with this drift of encoding. We therefore suggest that memory in a neurobiological network does not store and recreate the signals that pass through the network but is rather a mechanism that allows to store and reestablish the activation state of the network or parts thereof, i.e., to not recreate the result of some computation but to reestablish the conditions that led to the result.

We argue that such a type of memory has to be a *distributed, intrinsic part of the network* instead of being a dedicated, localized memory system. Moreover, we see local neuroplasticity, the characteristic responsible for the drift of encoding, as a key mechanism for this memory. It enables individual neurons to learn typical input patterns that capture some information about the statistical nature of their inputs (Kerdels and Peters, 2018). If such a neuron n_a receives its inputs from sensory cells, then the neuron learns something about the statistical nature of that part of the world that is transduced by these cells. If the neuron n_a receives its inputs from multiple other neurons n_i, it learns something about the statistical nature of the coactivation of these input neurons. It learns or forms an association between these cells, i.e., it becomes a small associative memory. However, at this stage it is more of an association detector or something akin to a hash function. The cell could answer the question "Have I seen this activation pattern before?", but it can not be "read out" to reestablish that activation pattern. A solution to this problem arises when one assumes that the neurons n_i are capable of forming associative memories themselves. In that case, the original associative memory neuron n_a would just have to form reciprocal feedback connections to all its inputs. Reading out this cell, i.e., activating it, would then result in reestablishing the activation state of the input neurons n_i. Although this reciprocal connection appears rather simple it can exhibit a range of interesting behaviors in relation to memory function depending on the balance of activation between the cell n_a and its inputs n_i. If the feedback connection between n_a and n_i is inactive then signals can flow in a bottom up direction allowing the neuron n_a to learn typical activation patterns of the neurons n_i. If the feedback connection becomes active the neuron n_a can keep the neurons n_i in a specific pattern of activation even if the original input signals to the neurons n_i are no longer present, i.e., the cells n_a and n_i behave now like a small short term memory of the original inputs to the cells n_i. Lastly, if the feedback of neuron n_a is dominant it can force the cells n_i to take on a pattern of activation that might disagree with the current bottom up input to the neurons n_i. In this case the behavior resembles the recall of a long term memory that overrides the current, bottom up driven state.

Viewed in isolation it might appear as a bold claim that the small neuronal circuit described above would constitute a proper memory mechanism. In particular, the capacity of the single cell n_a to learn and represent the activity patterns of its inputs n_i seems too limited. This issue can be addressed by expanding this basic mechanism to a local group of cells g_a that exhibit a form of competitive Hebbian learning via local inhibition. It can be shown that the resulting ensemble activity of such a group has a significantly higher capacity to learn and represent the activity patterns of their inputs (Kerdels and Peters, 2017). Expanding this model further one might imagine a network of such reciprocally connected groups of neurons that interact with each other according to the dynamics described above. Depending on the respective influences between the groups such a network would be continually driven towards states of temporarily stable, local equilibria that integrate current bottom up signals from, e.g., sensory cells, hold onto parts of this bottom up stream at appropriate stages of integration using local short term memory, and augment this integrated information by forcing activity patterns top down that correspond to a form of long term knowledge. Reaching a temporary equilibrium in this network could signal an episodic memory mechanism like the hippocampus in mammals to capture the activity of a small subset s of neuron groups at key locations within the network. When the activity pattern of this subset s is reestablished again, it would then force large parts of the network into a configuration that would approximate the state of the network when the episodic memory was formed. In principle, the episodic memory itself could rely on similar basic mechanisms of associative pattern recognition as described above.

In summary, we argue that memory in a neurobiological network is a distributed, dynamic process that is an intrinsic part of the network both structurally and functionally. This perspective of memory is in strong contrast to the concept of memory prevalent in well-known cognitive architectures that see memory as a distinct module that stores and retrieves data like the memory in a contemporary computer system. Integrating neurobiologically plausible forms of memory and computation in theories of cognition is a challenging task as it requires a significant shift in the perspective on typical computational paradigms. A recent example of a first step at such an integration is the work of Hamid and Braun who suggest to infuse the theory of reinforcement learning with the framework of attractor neural networks in order to benefit from the latter's capacity of pattern-completion and noise-insensitivity (Hamid and Braun, 2019).

5 CONCLUSION

In this position paper we argued that common intuitions about memory and computation adversely affect how we think about the processing of information in neurobiological systems, and how this perspective is reflected by prominent cognitive architectures. We provided an alternative view on memory and computation to show more clearly how these intuitions imply computational capabilities that are difficult to reconcile with the constraints inherent in neurobiological systems. In order to present a perspective on memory that is more neurobiologically plausible we outlined the idea of memory as an intrinsic, distributed part of a neurobiological network whose dynamic processes combine aspects of memory and computation in a way that questions the otherwise sharp delineations of these two concepts present in contemporary theories of cognition.

REFERENCES

Anderson, J. R. (1996). Act: A simple theory of complex cognition. *American Psychologist*, (51):355–365.

Baars, B. (2017). *The Global Workspace Theory of Consciousness: Predictions and Results*, pages 227–242.

Franklin, S., Madl, T., D'Mello, S., and Snaider, J. (2013). Lida: A systems-level architecture for cognition, emotion, and learning. *IEEE Transactions on Autonomous Mental Development*, 6.

Hamid, O. H. and Braun, J. (2019). *Reinforcement Learning and Attractor Neural Network Models of Associative Learning*, pages 327–349. Springer International Publishing, Cham.

Kerdels, J. and Peters, G. (2017). Entorhinal grid cells may facilitate pattern separation in the hippocampus. In *Proceedings of the 9th International Joint Conference on Computational Intelligence, IJCCI 2017, Funchal, Madeira, Portugal, November 1-3, 2017.*, pages 141–148.

Kerdels, J. and Peters, G. (2018). A grid cell inspired model of cortical column function. In *Proceedings of the 10th International Joint Conference on Computational Intelligence (IJCCI 2018), Seville, Spain, September 18-20*, pages 204–210.

Laird, J. E. (2008). Extending the soar cognitive architecture. In *Proceedings of the 2008 Conference on Artificial General Intelligence 2008: Proceedings of the First AGI Conference*, pages 224–235, Amsterdam, The Netherlands, The Netherlands. IOS Press.

Laird, J. E., Newell, A., and Rosenbloom, P. S. (1987). Soar: An architecture for general intelligence. *Artificial Intelligence*, 33(1):1 – 64.

Squire, L., Bloom, F., Spitzer, N., Squire, L., Berg, D., du Lac, S., and Ghosh, A. (2008). *Fundamental Neuroscience*. Fundamental Neuroscience Series. Elsevier Science.

Sun, R. (2007). The importance of cognitive architectures: an analysis based on clarion. *Journal of Experimental & Theoretical Artificial Intelligence*, 19(2):159–193.