Integrating computation into the mechanistic hierarchy in the cognitive and neural sciences

Lotem Elber-Dorozko and Oron Shagrir

Abstract: It is generally accepted that, in the cognitive sciences, there are both computational and mechanistic explanations. We ask how computational explanations can integrate into the mechanistic hierarchy. The problem stems from the fact that implementation and mechanistic relations have different forms. The implementation relation, from the states of an abstract computational system (e.g., an automaton) to the physical, implementing states is a homomorphism mapping relation. The mechanistic relation, however, is that of part/whole; the explanans in a mechanistic explanation are components of the explanandum phenomenon. Moreover, each component in one level of mechanism is constituted and explained by components of an underlying level of mechanism. Hence, it seems, computational variables and functions cannot be mechanistically explained by the medium-dependent properties that implement them. How then, do the computational and implementational properties integrate to create the mechanistic hierarchy? After explicating the general problem (section 2), we further demonstrate it through a concrete example, of reinforcement learning, in cognitive neuroscience (sections 3 and 4). We then examine two possible solutions (section 5). On one solution, the mechanistic hierarchy embeds at the same levels computational and implementational properties. This picture fits with the view that computational explanations are mechanism sketches. On the other solution, there are two separate hierarchies, one computational and another implementational, which are related by the implementation relation. This picture fits with the view that computational explanations are functional and autonomous explanations. It is less clear how these solutions fit with the view that computational explanations are full-fledged mechanistic explanations. Finally, we argue that both pictures are consistent with the reinforcement learning example, but that scientific practice does not align with the view that computational models are merely mechanistic sketches (section 6).

1. Introduction

The question of how different explanations in the cognitive sciences relate to each other is widely debated (Kaplan and Craver, 2011; Piccinini and Craver, 2011; Piccinini, 2015; Shapiro, 2017). We focus here on the relations between mechanistic explanations and computational explanations in the neuro-cognitive sciences. Mechanistic models describe the phenomenon's underlying mechanism. Often, they are considered explanatory because they describe a relevant causal structure, namely, the causal structure that underlies the explanandum. Moreover, there is a hierarchy of mechanistic explanations - each component in a mechanistic explanation is itself explained mechanistically. Computational explanations are similar to mathematical explanations in that they describe phenomena in abstract – mathematical or formal – terms. Computational explanations, however, are abstract in a further sense. They arguably describe abstract, "medium-independent", features. Thus, in computational explanations both the describing terms and the described objects/properties are abstract.

Several authors have recently suggested that computational explanations are a species of mechanistic explanation (Kaplan, 2011; Kaplan and Craver, 2011; Piccinini and Craver, 2011; Milkowski, 2013; Piccinini, 2015; Boone and Piccinini, 2016; Coelho Mollo, 2018; Dewhurst, 2018). The focus of most of these accounts is the neuro-cognitive sciences, in which computational models and explanations are central to the scientific investigation. Though the accounts are different in detail, they all share the starting point that computational explanations are in some sense abstract, whereas mechanistic explanations describe causal relations between physical entities. Each account offers a unique way to bridge the apparent disparity between computational and mechanistic explanations.

Whether computational models are indeed mechanistic is still under controversy (Huneman, 2010; Piccinini and Craver, 2011; Weiskopf, 2011; Kaplan, 2011; Kaplan and Craver, 2011; Lange, 2013; Chirimuuta, 2014, 2018; Bechtel and Shagrir, 2015; Rathkopf, 2015; Craver, 2016; Shagrir and Bechtel, 2017; Shapiro, 2017; Craver and Povich, 2017; Egan, 2017). Here we do not focus on this controversy (though our analysis might have some implications regarding the nature of computation). Our concern is with the integration of computation – mechanistic or not – within the hierarchy of mechanistic explanations. The concern arises from the disparity between the implementation (or realization) relation and the explanans-explanandum relation in mechanistic explanations. The implementation relation from the states of an abstract computational system (e.g., an automaton) to the states of its implementing physical system is a homomorphism mapping relation, so that each distinct computational state is mapped onto a distinct physical state, which realizes it. The mechanistic relation, however, is that of part/whole. The explanans in a mechanistic explanation are components of the explanandum phenomenon. Moreover, each component in one level of mechanism is constituted and explained by components of another, underlying, level of mechanism. Hence, it seems, computational states are realized in some physical structures, but they do not stand in part/whole relations to them and therefore they cannot be mechanistically explained by the same structures. So, the question is: how do computational states integrate with implementational states to form the mechanistic hierarchy?

Before turning to address this question, we want to describe the main features of mechanistic and computational explanations. Mechanistic explanations have three main features: they are causal, decompositional and hierarchical. They are causal in that they explain phenomena by describing their underlying mechanism. Consider the reflex that is responsible for keeping the direction of gaze constant when the head is rotated horizontally. It is called the horizontal vestibulo-ocular reflex. Its function is explained by reference to an underlying mechanism whose inputs are the effects of head movements on the vestibular organ and whose outputs are given to the ocular muscles. Within the mechanism there are feedforward inhibitory and excitatory synaptic connections, so that each pre-synaptic neuron causally affects the post-synaptic neurons through the synaptic connections (Kandel *et al.*, 2013, chap. 40). Mechanistic explanations are decompositional because the explanandum phenomenon is explained in terms of its components, their organization and their activities (functions). In our example the constant gaze when the head is rotated is explained by appeal to the specific synaptic connections between neurons, as well as the neurons' change in firing rate in response to their synaptic inputs. Finally, mechanistic explanations are hierarchical: each explaining component in one level is itself the explanandum for another level of mechanism. Accordingly, the release of neurotransmitter to the synapse by the pre-synaptic neuron, is also explained mechanistically (see (Piccinini and Craver, 2011)). Our focus here is the third feature of mechanistic explanations, namely, the mechanistic hierarchy. An important point about the hierarchy is that each level in the hierarchy is a mechanistic explanation.

Computational explanations are taken to be abstract in that they refer to abstract, "medium-independent", properties. This claim is fairly uncontroversial.[1] What perhaps is more controversial is the claim that computational explanations refer *only* to abstract, formal properties. Some authors argue that computational explanations also refer to semantic properties, namely to the specific content of the states (Shagrir, 2006; Sprevak, 2010); others might insist that computational explanations also refer to some implementational, medium-dependent, properties (Some of the writings of (Kaplan, 2011, 2017; Dewhurst, 2018) may be interpreted this way). We will not get into the debate about the nature of physical computation. Our concern is with the integration of abstract states and properties of computation in the mechanistic hierarchy[2]. We take abstract here to mean 'medium-independent' in the sense that they can be implemented in very different physical media (e.g., both in brains and in computers). We will refer to these states and properties as computational. But by this we assume in no way that computational states and processes are only abstract.

[1] There are, however, different ways to account for the nature of these "medium-independent" properties. Fodor (1975) and Stich (1983) describe them as "syntactic" properties, and Fodor (1994) accounts for the latter in terms of high-level physical properties. Haugeland (1981) describes them as "formal" (see also (Fodor, 1980)). Piccinini (2015) describes computational properties as "mathematical" or "formal", and others have suggested that, regarding computations, the relevant physical properties of the implementing physical systems are only their degrees of freedom (Piccinini and Bahar, 2013; Coelho Mollo, 2018).

[2] While it seems straightforward to associate the computational explanations discussed here with Marr's computational level (1982), algorithmic descriptions of a system can also be abstract and computational in the meaning we discuss here, as long as they are 'medium-independent'. These algorithmic descriptions are more similar to mechanistic explanations in that they usually decompose the explanandum into its parts, while computational level explanations describe 'what' function the system performs and 'why' (Shagrir and Bechtel, 2017).

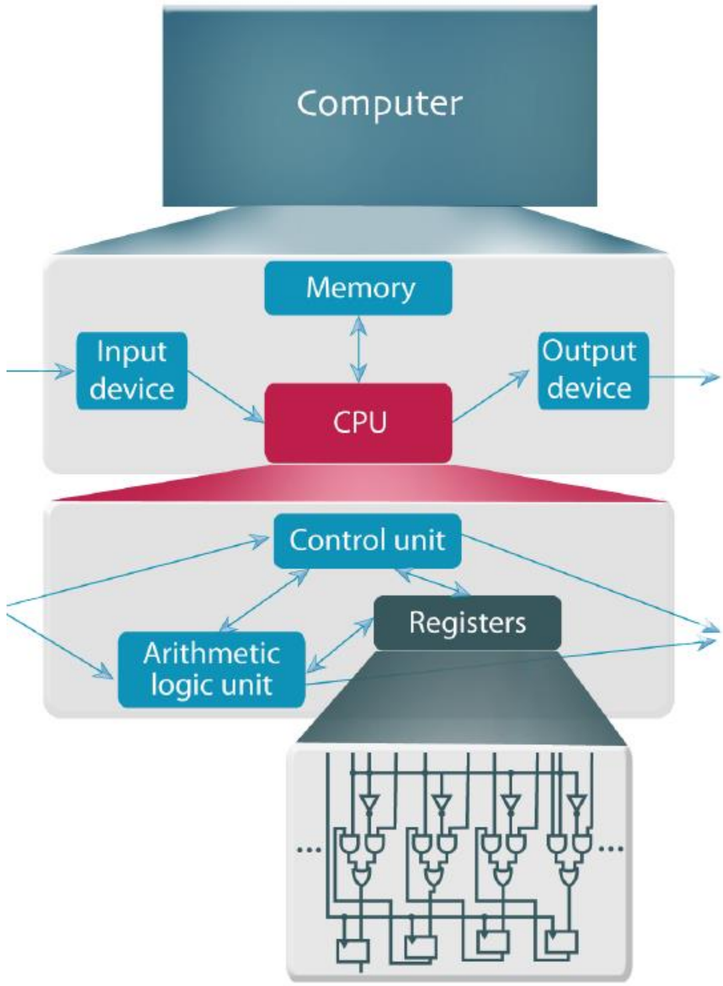
2. The computational and implementational hierarchies

Let us turn to the problem of integrating computational states and properties within the mechanistic hierarchy. As a warm-up, let us look at the way Piccinini describes this integration. Piccinini (2015), who defends the view that computational explanations are mechanistic, takes those computational levels to be levels of mechanism. In a crucial paragraph in his book he says the following:

> The mechanistic account flows naturally from these theses. Computing systems, such as calculators and computers, consist of component parts (processors, memory units, input devices, and output devices), their function and organization. Those components also consist of component parts (e.g., registers and circuits), their function, and their organization. Those, in turn, consist of primitive computing components (paradigmatically, logic gates), their functions, and their organization. Primitive computing components can be further analyzed mechanistically but not computationally (2015, pp. 118–119).

Now, we think that it is uncontroversial that Piccinini describes here levels of computation that relate to each other in a part/whole relation. As Piccinini depicts it, computers consist of processors, memory etc., which in turn consist of registers and circuits, which in turn consist of logic gates (figure 1).

Figure 1 – The computational hierarchy



However, Piccinini does make a controversial claim, namely that computational explanations are mechanistic. This claim has been criticized on three main grounds. Some critics argue that, even if some computational explanations are decompositional as in the described case, there are other cases in which computational explanations do not decompose the explananda into components, but instead refer to general structural or topological properties of the system, and so are not mechanistic (Huneman, 2010; Rathkopf, 2015; but see Craver, 2016). A second criticism is that computational explanations do not always aim to reveal causal structures. Egan (2017) suggests that computational models are explanatory because they are abstract and normative. Chirimuuta (2014) suggests that some computational models explain why a computation takes place by appeal to efficient coding principles, and Shagrir and Bechtel suggest that some computational models also explain the existence of a computation by appeal to environmental constraints

(Bechtel and Shagrir, 2015; Shagrir and Bechtel, 2017). According to these two criticisms, computational explanations are not wholly mechanistic, but it still may be that some computational explanations, which refer to medium-independent properties, are decompositional, and therefore may be mechanistic.

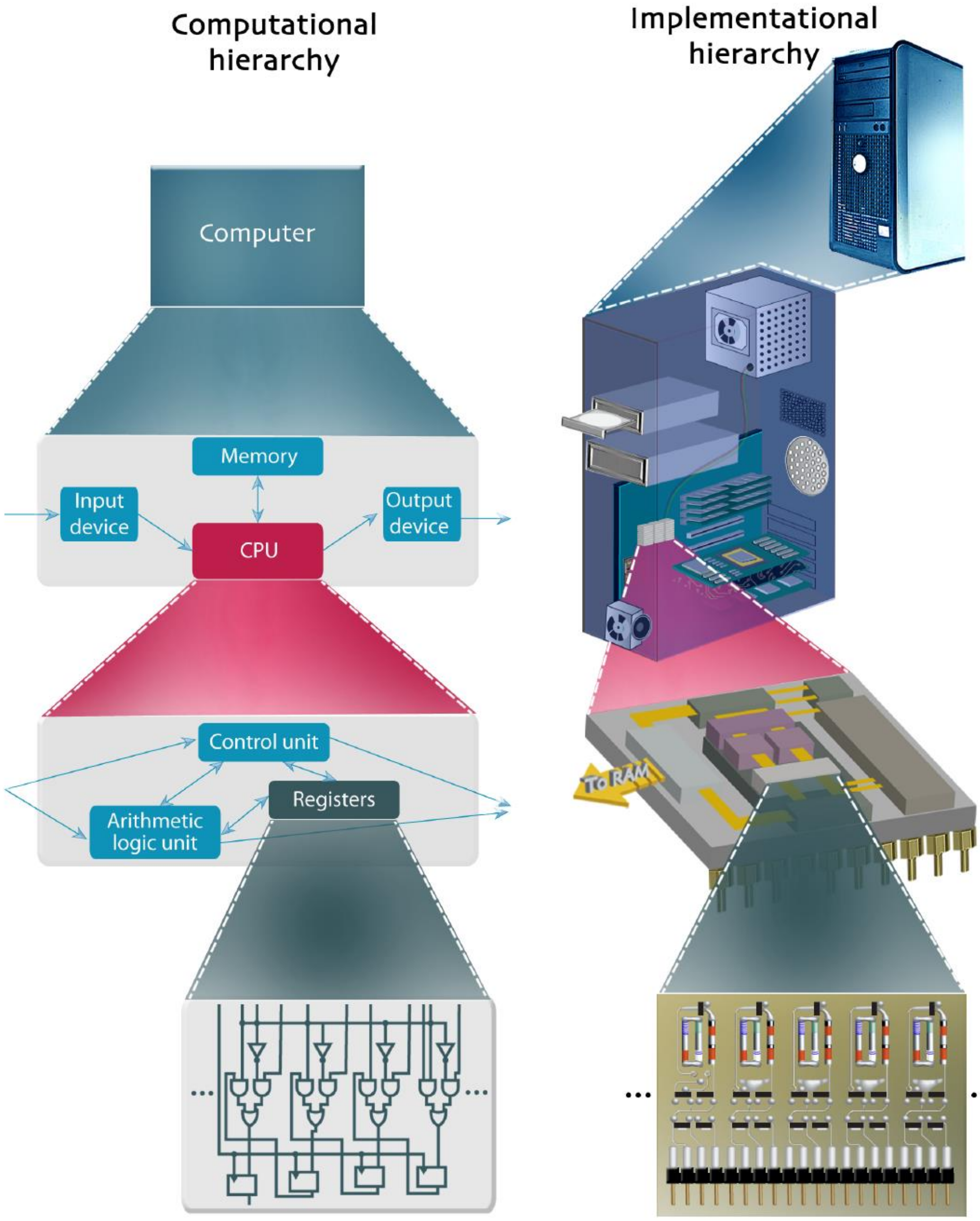
Other critics argue that, even when computational explanations involve decomposition, the resulting levels of computation are not levels of mechanisms. Instead, they argue that these levels are functional; they are part of a functional analysis which explains the capacity (Fodor, 1968; Cummins, 1983, 2000). These critics would agree that the levels are decompositional, relating to each other in a part/whole fashion, which is perfectly consistent with the functional account of computational explanations. They would also agree that the pertinent computational properties are "medium-independent", at least in the sense that they refer to abstract and not to medium-dependent, implementational, properties. The critics would argue, however, that the divide between the abstract/medium-independent properties and implementational properties is indicative of the divide between functional and mechanistic explanations (Weiskopf, 2011; Shapiro, 2017). Because functional and implementational entities are inherently different, computational and mechanistic explanations take place in different levels of explanation. Piccinini (2015) in turn rejects the functional/mechanistic distinction, arguing that functional explanations are sketches of mechanism (Piccinini and Craver, 2011). Moreover, he argues that computational explanations are (ideally) both abstract and full-fledged mechanistic. They are abstract in the sense that they refer to medium-independent properties. They are mechanistic in the sense that the medium-independent properties constrain the implementation ((Piccinini, 2015) But see Shapiro (2017) for criticism).

We put aside the question of whether the computational level – as a level of abstract, medium-independent, properties – sufficiently constrains implementation to be considered mechanistic. We want to highlight a different issue that Piccinini and others do not discuss, namely, the way that computational (medium-independent) and implementational (medium-dependent) properties relate to each other in the mechanistic hierarchy.

The picture depicted by Piccinini raises two (related) issues. The first pertains to the primitive computing components. Piccinini says that “primitive computing components can be further analyzed mechanistically but not computationally”. He means that we can further analyze the logic gates in terms of non-computational, medium-dependent properties. The difficulty is that the logic gates are also *implemented* in some medium-dependent properties. The inputs and outputs of logic gates – typically characterized as 1s and 0s – are often implemented in systems with specific voltages. The implementing physical objects with specific voltages, however, are not *parts* of the digits. More generally, implementation is often characterized as a mapping homomorphism relation from the states of an abstract computing system (e.g., an automaton) to groups of states of a physical system. For example, there is a mapping from the digits 0 and 1 to the sets of voltages, 0-5 volts and 5-10 volts. The sets of voltages, however, are not themselves the mechanism that constitute the digits. The question raised, then, is about the relations between the medium-independent properties that analyze computation in the mechanistic explanation and the medium-dependent properties that implement computation. The first ones, the analyzing properties, seem to be parts of the digits, whereas the second ones, the implementing properties, are not. Are these the same properties and how do they relate to each other? We expect a part-whole mechanistic analysis, but we can only find in this stage an implementation-relation and not a part-whole relation, so how can logic gates be explained mechanistically?

A second issue concerns the non-primitive computing components. The components of a higher-level computation are analyzed by an underlying computational level. But they are also implemented in some medium-dependent properties. How are these underlying properties – the computational and implementational – related? Take the computational level that consists of “component parts (e.g., registers and circuits), their function, and their organization”. Let us call it C_n. The components of C_n can be analyzed, computationally, by the computational components of an underlying computational level C_{n-1} (e.g., logic gates). However, the computational components of C_n are also implemented in some medium-dependent properties that belong to some mechanistic level, P_k. But how are P_k and C_{n-1} related in the mechanistic hierarchy? Moreover, P_k itself is part of a hierarchy, P_0, P_1, P_2,… So, there are two hierarchies, one computational, C_1, C_2,… and one implementational, P_0, P_1, P_2,… (figure 2).

Figure 2 The computational and implementational hierarchies



Several issues are worthwhile addressing regarding this picture. First, in some cases computational explanations are not decompositional (Huneman, 2010; Chirimuuta, 2014; Bechtel and Shagrir, 2015; Rathkopf, 2015; Egan, 2017; Shagrir and Bechtel, 2017), and therefore are not hierarchical. Although in such cases we will not find two or more hierarchies, the question of how the single-level computational explanation is integrated into the implementational hierarchy persists.

We would also like to note that much of the structure of these two hierarchies and their relations depends on how one defines 'a level of explanation'. There is practically unanimous agreement that in the scientific investigation of cognitive capacities both the underlying computation and the underlying implementation should be addressed eventually. The question that is under debate addresses the relevant details for a complete explanation of a capacity at a specific level. According to the mechanistic framework, a complete explanation at each level will include all the causally relevant relations and activities that constitute the explanandum capacity.

Our question then is how the computational, medium-independent properties and their implementational, medium-dependent, properties relate to each other in the scientific explanation.[3] Do we really find two hierarchies, one computational and one implementational, in which each level in each hierarchy is a complete explanation? And if this is indeed the case, then how do the two hierarchies relate to each other?

3. A hierarchical computational model for reinforcement learning

It could be argued that the two hierarchies we describe in the decomposition of the computer are the result of a specific man-made design, and that the observations from a computer cannot be generalized to the cognitive sciences. For this reason, it

[3] One can also ask how the implementational hierarchy is decomposed. Depending on one's view of a level of explanation, the implementational hierarchy will include different details. It can include merely a reference to the physical structures that underlie the computational function. Alternatively, this hierarchy can also describe functions executed by these structures, albeit, medium-dependent functions. To illustrate, diodes, which are used on occasion to build logic gates in computers, have the function of passing electric current in exactly one direction. Description of such functions can be a part of the implementational hierarchy, because such functions are not abstract, but instead describe medium-dependent processes. In both cases the decomposition of the implementational hierarchy will depend on some function, in the first case it is the computational function, and in the second it is the medium-dependent function (which may or may not coincide with the computational function).

is useful to examine the relation between computation and implementation in the mechanistic hierarchy with the help of an example from neuro-cognitive science.

Reinforcement learning is a behavior in which the subject learns to choose specific actions according to their consequences, with the goal of maximizing rewards. It is widely investigated; it has received attention both from computer scientists who have suggested algorithms for action selection that maximize specific outcomes (Sutton and Barto, 1998), and from neural and cognitive scientists who have compared various reinforcement learning models with subjects' behaviors (Mongillo, Shteingart and Loewenstein, 2014; Shteingart and Loewenstein, 2014) and searched for neural correlates of variables from reinforcement learning algorithms (Samejima *et al.*, 2005; Li and Daw, 2011; Wang, Miura and Uchida, 2013).

Reinforcement learning is a process that requires multiple different computations, and as such it can be viewed hierarchically. At the highest level, reinforcement learning is divided into four main processes, each involving its own computations: recognizing the subject's state, evaluating potential actions, selecting an action, and reevaluating the action based on the outcome (Doya, 2008).

Each one of these processes has been discussed in large bodies of literature and can be further decomposed in various ways. To provide more concrete examples we will discuss reinforcement learning in the context of a multi-armed bandit task, where there is only one state in which the subject repeatedly chooses between multiple actions, each associated with a certain magnitude and probability of reward. We describe here a simple and widely used algorithm for reinforcement learning, which is called Q-learning (because the values associated with the actions are called Q-values) (Sutton and Barto, 1998). In a multi-armed bandit task, reinforcement learning has two main modules (instead of the four we originally mentioned), action reevaluation and action selection.

Consider the module which is responsible for reevaluating an action after an outcome. In Q-learning, each Q-value is meant to reflect the expected reward associated with each action, also called the action-value. In order to learn this action-value, after each trial a variable called the reward prediction error (RPE) is computed. The RPE is the difference between the reward that was just received and the current value of the chosen action:

$$for\ the\ chosen\ action\ a_i \rightarrow RPE(t) = R(t) - V_i\ (t) \quad (1)$$

Where $R(t)$ is the reward given at time t, a_i is action i and $V_i\ (t)$ is the action-value of action i at time t. Then, the value of the chosen action is updated by summing the previous value with a magnitude that is proportional to the RPE. Written formally:

$$\begin{matrix} if\ a_i\ was\ chosen \rightarrow V_i\ (t+1) = V_i\ (t) + \alpha \cdot RPE(t) \\ if\ a_i\ was\ not\ chosen \rightarrow \quad V_i\ (t+1) = V_i\ (t) \end{matrix} \qquad (2)$$

Where α is a parameter that indicates the learning rate. The larger α is, the more weight recent trials are given at the expense of previous trials.

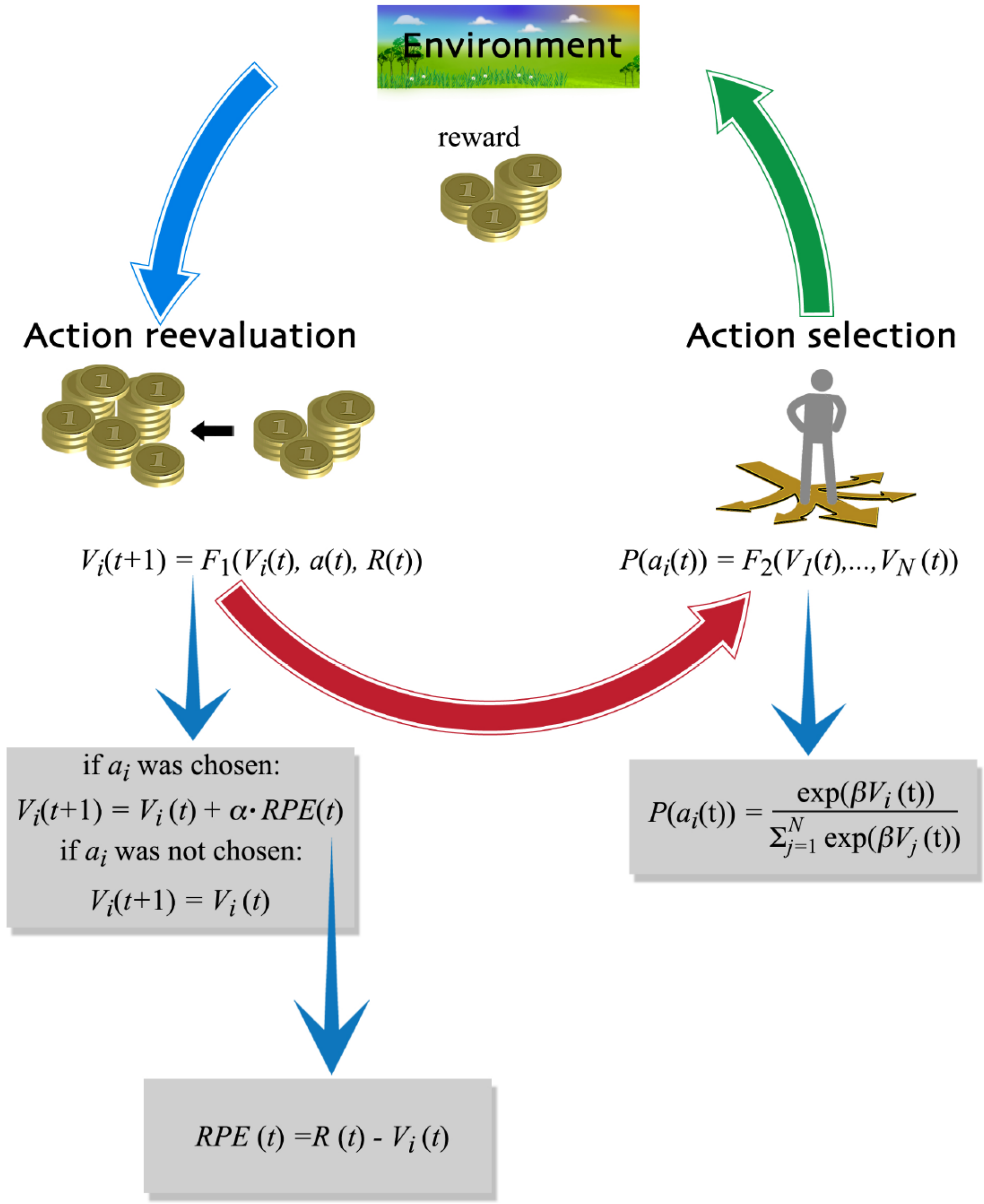
If we wish, we can continue this hierarchical computational explanation, by explaining how the components in eq. (1)-(2) are computed. For example, we can explain how the learning rate 'α' is computed. We can also explain how the reward is evaluated, or what the initial conditions set for $V_i(t=0)$ are.

Consider now the second module, the module that is responsible for selecting between different actions. The simplest kind of module would just select the action that has the highest value, according to the computation in eq. (2). However, this method may never sample actions that initially received lower values, even in cases where these lower values were underestimates of the true values. Therefore, it is generally agreed that some form of exploration is required, i.e., actions with lower values should be chosen with a non-zero probability. A common model that incorporates exploration into the choice is a 'softmax' function where actions with higher values have a higher probability to be chosen. The 'softmax' function is:

$$P(a_i(t)) = \frac{e^{\beta V_i\ (t)}}{\sum_{j=1}^{n} e^{\beta V_j\ (t)}} \quad (3)$$

Where a_i is action i, $P(a_i(t))$ is the probability of choosing action i at time t, $V_i\ (t)$ is the action-value of action i at time t, n is the number of possible actions, and β is a parameter that determines the bias of the choice towards the higher valued actions. The components of this action selection function can also be further explained. For example, in this equation, the choice is stochastic. We can also provide a model for this stochasticity. Or we can explain the choice of β, which may be a constant, or change throughout learning. Fig. 3 presents a summary of the hierarchical model we described so far.

Figure 3 The computational hierarchy of the Q-learning model



Using the two modules described above, in a multi-armed bandit task, in which subjects choose between several actions repeatedly, it is possible to learn to choose the action that is associated with the largest expected reward most frequently. Hence, a popular theory in the cognitive sciences is that people employ a model similar to Q-learning in various instances of reinforcement learning tasks.

Q-learning is not the only model that has been suggested for reinforcement learning, it has a few competitors at several different levels. First, some reinforcement learning algorithms do not compute the values of actions at all. Instead, learning is done directly on the 'policy': the probability of choosing each action. These are called direct-policy learning algorithms (Mongillo, Shteingart and Loewenstein, 2014; Shteingart and Loewenstein, 2014). Second, in the Q-learning model the action selection function (eq. 3) utilizes the same action-values as the action reevaluation function (eq. 2). However, in some reinforcement learning algorithms, the action selection function does not employ the action-value estimates of the action reevaluation function. Instead, the only signal the action-selection function receives from the action-reevaluation function is the RPE. In these algorithms, these two modules are also called the 'actor' and the 'critic', respectively (Sutton and Barto, 1998). A third issue concerns the complexity of Q-learning. It is argued that it is too simple to explain a wide variety of behaviors and therefore this original model has been developed into alternative, more complicated models (Botvinick, Niv and Barto, 2009; Botvinick, 2012). Each of these three groups of competing models challenges a different part of the computational hierarchy of Q-learning. The first group of models challenges whether there is an action reevaluation function at all, the second group of models questions the relation between the action selection and the action reevaluation functions and the third presents alternatives to the structure within each function.

We believe that the point is clear, the Q-learning model is hierarchical in nature. Furthermore, all properties discussed in the Q-learning model are medium-independent: they do not necessitate a specific physical structure. In fact, they are abstract enough that they can be both implemented in computers and, as many scientists hypothesize, in brains (Schultz, Dayan and Montague, 1997; Doya, 2000, 2008; O'Doherty *et al.*, 2004; Samejima *et al.*, 2005).

4. The computational and implementational hierarchies of reinforcement learning

A great deal of scientific research has been dedicated to the characterization of the neural correlates of the Q-learning model (Hollerman and Schultz, 1998; Doya, 2000, 2008; Samejima *et al.*, 2005; Ito and Doya, 2009; Kable and Glimcher, 2009; Tai *et al.*, 2012; Wang, Miura and Uchida, 2013). Experimental evidence has implicated the basal ganglia, a group of several subcortical nuclei, including the striatum, pallidum and substantia nigra, in decision making, and specifically in the context of reinforcement learning (Doya, 2000). With regard to the different modules of reinforcement learning, the coding of state and possible actions in each state has been attributed to the cortex, the calculation of the expected reward associated with each action (action reevaluation) has been attributed to the striatum, action selection has been attributed to the pallidum, etc. In Fig. 4 you can see a scientific hypothetical model which describes the implementation of the computational modules in reinforcement learning.

Figure 4. The implementational model for reinforcement learning. Adopted from (Doya, 2008). Legend is taken from the original paper.

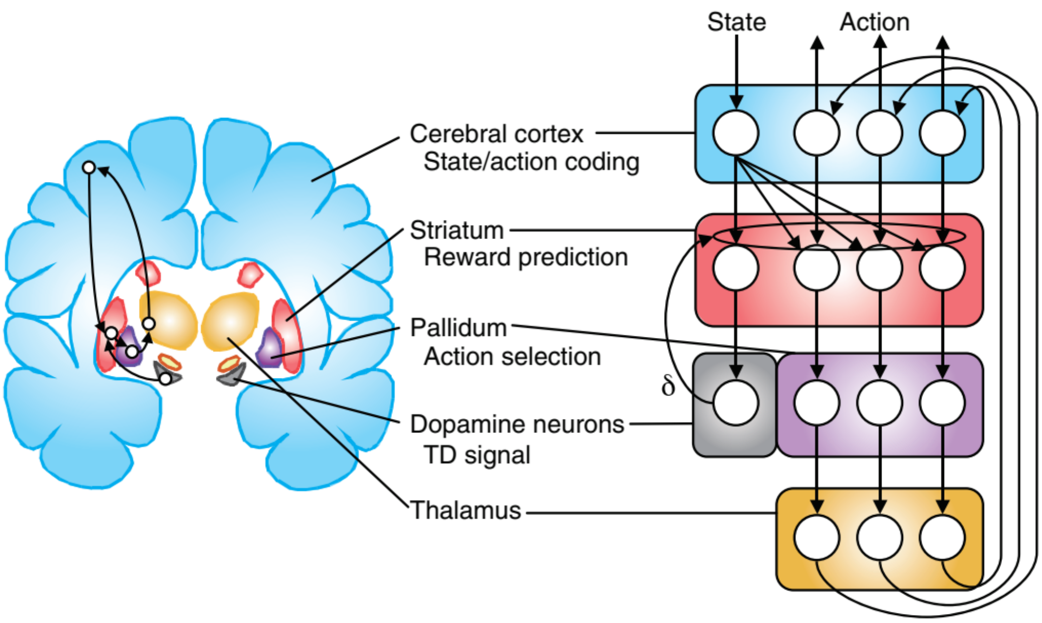


Figure 2 A hypothetical model of realization of reinforcement learning in the cortex–basal ganglia network[2]. Left, coronal section of the brain. Right, functional model, where δ denotes the reward prediction error carried by the midbrain dopamine neurons.

The attribution of specific computational properties to brain areas corresponds to their connectivity patterns. On the Q-learning model we expect action-values to play a part in the action selection function (eq. 3). On our implementational model striatal neurons represent action-values and pallidal neurons are responsible for action selection. Indeed, in line with the computational model, we see that striatal neurons target and causally affect pallidal neurons. Hence, on this description, abstract computational relations are translated into causal relations between physical brain areas.[4]

One can wonder about the model on the right-hand side of Fig. 4. While the model on the left-hand side clearly describes causal relations between brain areas, the model on the right-hand side is abstract and is termed functional by (Doya, 2008). Although its drawing is abstract, this model is committed to specific brain areas, sometimes describing brain areas without an apparent function (such as the Thalamus). For this reason, it would be difficult to consider this model a functional analysis, as described by (Fodor, 1968; Cummins, 1983, 2000). Furthermore, this

[4] Some may argue that relations between computational components can already be considered causal relations. We discuss the possible outcomes of this position in section 5.

model is committed to specific media, namely, brain areas, and therefore it does not describe medium-independent properties. For this reason, we consider it an implementational model. However, for those who believe that computational models are both complete mechanistic explanations and medium-independent (Piccinini, 2015), this model, which focuses on the abstract functions of specific brain areas, may be similar to what they have in mind[5].

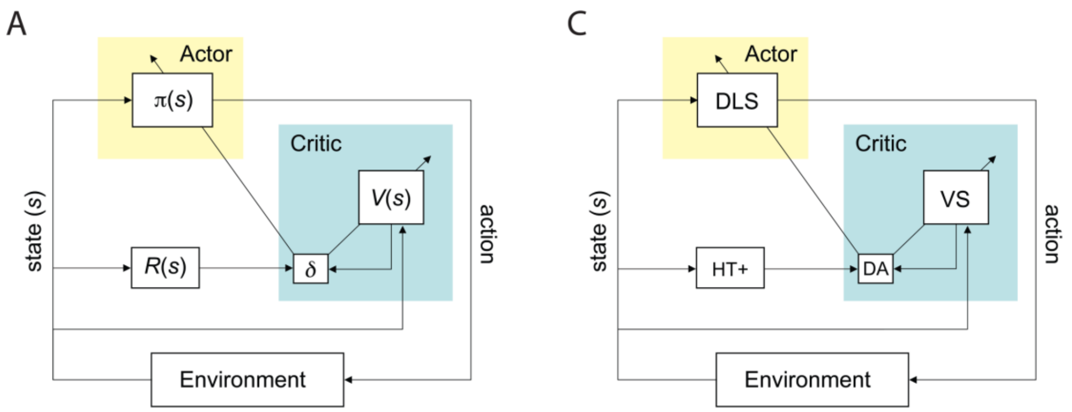
The components in the implementation described in Fig. 4 can be decomposed themselves into subparts, which correspond to parts of the computations. For example, there is experimental evidence that midbrain dopaminergic neurons that provide input to striatal neurons, encode the reward prediction error (RPE) (eq. 1), which is a component in the calculation of action-values (eq. 2) (Schultz, Dayan and Montague, 1997; Hollerman and Schultz, 1998). To provide another example, neurons in both the ventral and dorsal striatum receive inputs from midbrain dopamine neurons, which are taken to encode the RPE (note the arrow from the gray box to the red box in Fig. 4). Therefore, both are taken to play a role in reward prediction. Experimental findings have suggested that neuronal activity in the striatum can be divided into two anatomically and functionally separate parts of reward prediction: the dorsal striatum plays a role in associating stimuli with responses, corresponding primarily to an 'actor' (action selection) module, while the ventral striatum plays a role in updating the predictions of future rewards expected in each state, corresponding to a 'critic' (action reevaluation) module (O'Doherty *et al.*, 2004).

We see in this example two distinct hierarchies, one computational and one implementational. Parts of the computational hierarchy can be seen in Fig. 3. This hierarchy is abstract, medium-independent and can be discussed without mention of any brain structures. We can also see an implementational hierarchy, part of it is depicted in fig. 4, where brain structures are decomposed into functionally and anatomically individuated components. In some scientific publications we even see

[5] If this is the case, some issues regarding this view should be resolved. Most importantly, how function can remain medium-independent when it is necessary to state the brain structure in which they occur (Haimovici, 2013).

computational and implementational models for decision making (albeit slightly different models from the Q-learning model) depicted side by side, as in Fig. 5.

Figure 5 Computational and implementational models, side by side. Adopted from (Botvinick, Niv and Barto, 2009). R(s): reward function; V(s): value function; δ: reward prediction error; π(s): policy (action-selection function). DA: dopamine; DLS, dorsolateral striatum; HT+: hypothalamus and other structures; VS, ventral striatum.



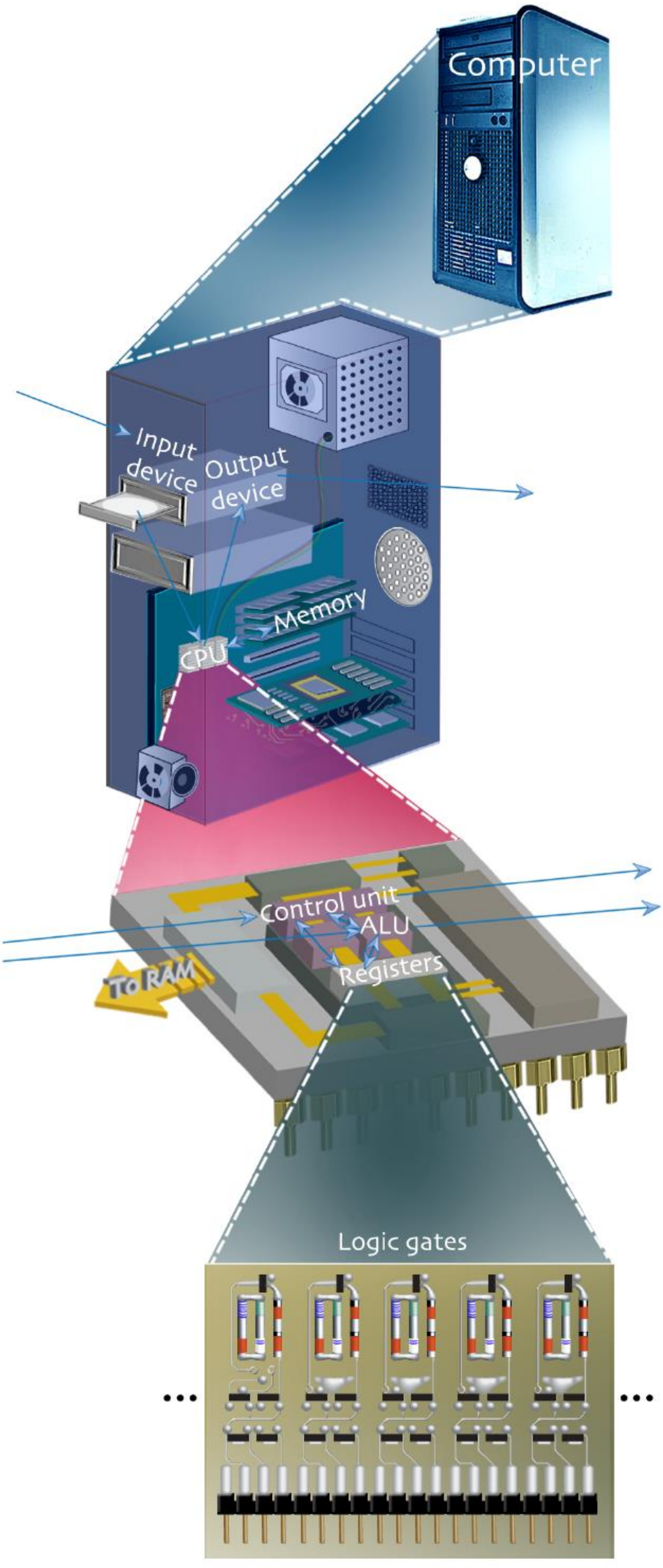
The relation between these two hierarchies is that of implementation, throughout the scientific literature brain structures are described as ‘implementing’ (Ito and Doya, 2011), ‘realizing’ (Doya, 2008), ‘representing’ (Samejima *et al.*, 2005) and ‘encoding’ (Schultz, Dayan and Montague, 1997) computational properties.

5. The relation between the computational and implementational hierarchies

We found in our scientific example two hierarchies, like the ones described in Fig. 2. However, there are still many open questions about these hierarchies, both in general and in our example. How do these hierarchies relate to each other within the scientific explanation? How does this relation reflect the explanatory role of the computational and implementational models? Finally, what role do implementation relations and part/whole relations play in the explanation of cognitive phenomena? In this section, we suggest possible answers to these questions and investigate their merit. We relate these possible answers to the different views about abstractness and completeness of computational models. We do not aim to support one stance on this question, but instead wish to examine the consequence of the different positions about computational models as explanations and start a debate about these possible solutions.

We can think of two ways to relate computation and implementation to each other within the mechanistic hierarchy. One is lumping together the implementational and the abstract properties in each level, namely C1 and P1, C2 and P2 and so on. Figure 6 shows an example of this picture on the decomposition of a computer.

Figure 6 A single combined mechanistic hierarchy. Each level includes both abstract and implementational properties that are related through implementation. The implementational properties are denoted by the drawings in the figure, while the computational properties are denoted by the words and arrows appearing on top of the implementational properties.

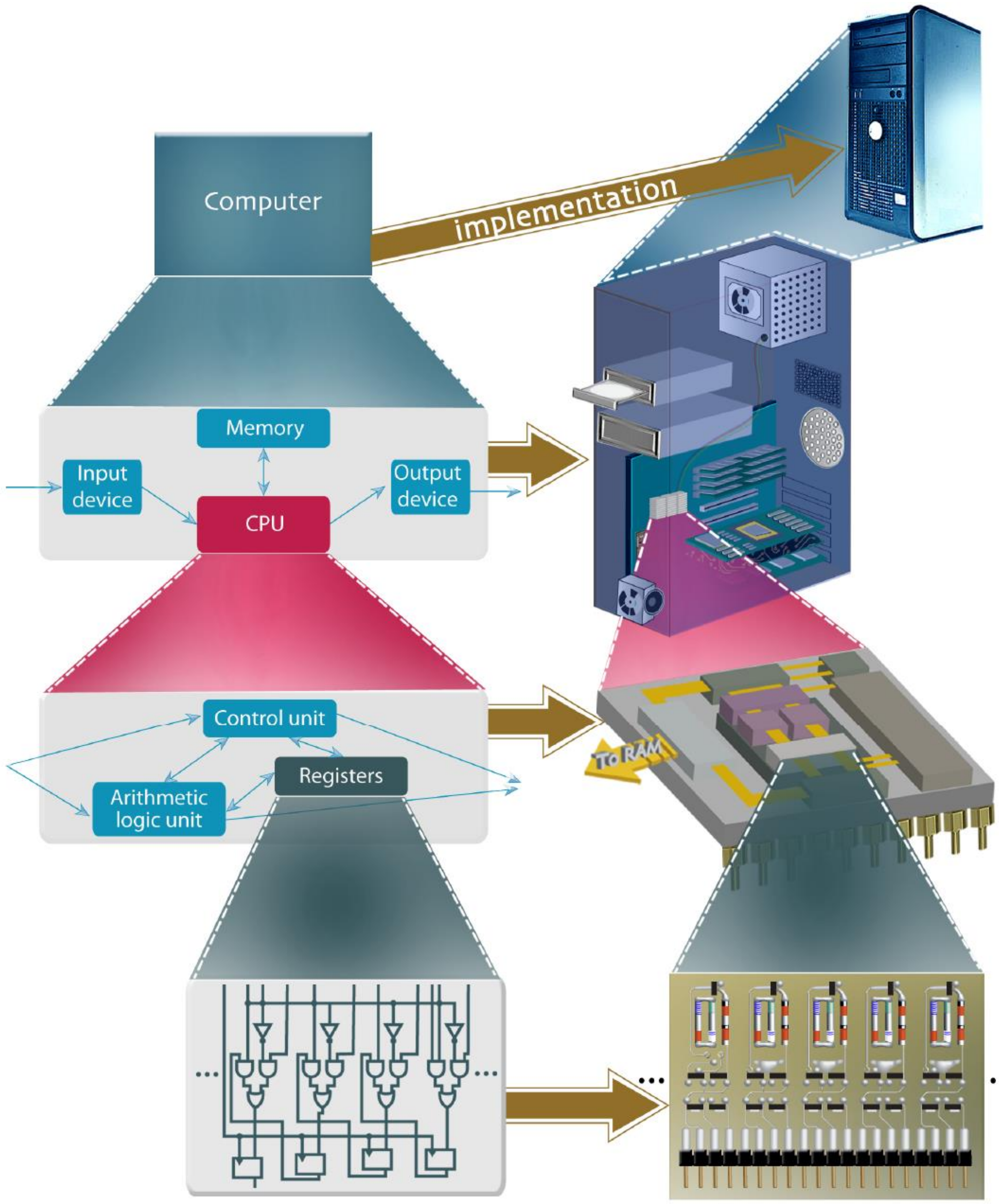

Computer
Input device
Output device
Memory
CPU
Control unit
ALU
Registers
To RAM
Logic gates

On this picture we do not really have two separate hierarchies, but only one: The pertinent computational properties are lumped together with their implementational properties in the same level(s) of explanation (a similar structure of explanation is presented in (Harbecke, under review)). This simple solution implies that computational and implementational properties figure together in the same explanation and in the same levels of the mechanistic hierarchy. This solution is in tension with the view that computational explanations are autonomous from implementation and therefore do not require implementation details to be complete, but fits quite nicely with the picture on which computational explanations are sketches of mechanisms (some people, e.g., (Rusanen and Lappi, 2016; Shagrir, 2016) interpret (Kaplan and Craver, 2011; Piccinini and Craver, 2011) as advocates of this position). On this picture, the computational sketches turn into a full-fledged mechanistic explanation only when we complement the sketches with the same-level implementational properties. When both kinds of properties are mentioned then we have a full-fledged mechanistic explanation, hence a level of mechanism. The mechanistic hierarchy simply embeds within it, a sub-hierarchy of computational sketches.

We can see two possible upshots of this construal, depending on one's view of computational models as sketches. One may consider computational sketches to simply be partial descriptions of the implementational model and computational properties to simply be abstract facets of the implementing properties, stripped away from their medium-dependent aspects. On this formulation, when the implementing properties are described in an explanation, the computational properties, which are merely a part of the implementational properties, become redundant. We are left with an implementational hierarchy, partial descriptions of which are computational models. On such a view it is clear how there is only one mechanistic hierarchy – an implementational hierarchy. However, this view completely dismisses any explanatory value of computational descriptions that goes above implementational descriptions and some may argue that this is inconsistent with scientific practice, which often appeals to computational explanations as more than partial implementational descriptions (Haimovici, 2013). Alternatively, one may believe that computational sketches can include details and aspects which are not part of the implementational model. For example, that they address environmental constraints or efficient coding principles (Chirimuuta, 2014; Bechtel and Shagrir, 2015; Shagrir and Bechtel, 2017). Therefore, in the complete model both computational and implementational properties figure together. This view takes computational descriptions to be more than partial implementational descriptions, but it brings up the original problem discussed in this paper - how the unique computational properties relate to the implementational properties in each level of the hierarchy.

A second option is to keep the two hierarchies apart (figure 7). The two hierarchies are related through the implementation relation. The computational properties of C1 are mapped (implemented by) to the implementational properties of P1, the computational properties of C2 are mapped to the implementational properties of P2, and so on. While objects by the same name may appear in both hierarchies, such as CPUs and registers in Fig. 7, the computational hierarchy includes only abstract, medium-independent properties (e.g., digits in logic gates) and the implementational hierarchy includes physical, medium-dependent properties (e.g., voltages). Fig. 7 presents a simple case where each computational level is mapped to each implementational level. In reality there might not be a perfect match between the hierarchies and computational properties at the same level may be implemented in implementational properties in different levels. However the structure of the implementation relation, in all cases in this picture there are two hierarchies and the computational properties in the computational hierarch are implemented by implementational properties in the implementational hierarchy. This solution is more hospitable to the notion that there is multiple realization of cognitive functions, since the same computational hierarchy can be related to (i.e., implemented in) different implementational hierarchies.

Figure 7 Two separate hierarchies, one computational and one implementational, that are related through implementation. Each level in each hierarchy is a complete explanation of the phenomenon at the higher level.



This picture fits quite nicely with the functional view of explanation, namely, the idea that computational explanations are full-fledged functional (yet non-mechanistic) explanations. According to this functional picture, computational explanations are distinct and autonomous from mechanistic explanations (Fodor, 1968; Cummins, 1983), which fits with the solution in which the two hierarchies are distinct. Computational and implementational properties do not figure together in the decompositional explanation of the same capacities. Instead, only computational properties are part of the decomposition of computations. Implementational properties can still figure in explanations of computations, but these explanations will not be mechanistic because there is no part/whole relation between explanans and explanandum. While on this picture the two hierarchies are separate, they still constrain each other: the relevant implementational properties are determined according to the computational function, and the computational hierarchy must be one which can be implemented in the physical system. Despite these mutual constraints, those supporting this picture will argue that the computation performed as part of some cognitive capacity can be given a complete explanation at one level without any reference to implementation and that the implementation details explain a different aspect of this capacity, namely, how the capacity is implemented. That is, computational and implementational explanations answer different questions.

On both pictures, primitive computing processes are analyzed mechanistically, if at all, only indirectly. The primitive computational components, e.g., logic gates, are *implemented* in some implementational properties, e.g., voltages, whereas only the latter can be further analyzed mechanistically. On the combined-hierarchy picture (Fig. 6), the computational properties will figure together with implementational properties in each level, until at some point the primitive computing processes can no longer be decomposed, and only implementational properties will continue to be decomposed in the hierarchy. On the separate-hierarchies picture (Fig. 7), the computational hierarchy will terminate at the primitive computing components.

On both pictures, the implementation is not a part/whole relation and therefore the description of implementation cannot be taken as a mechanistic explanation. Nonetheless, these two pictures do differ in how they view the role of implementation in explanation in general. On the combined picture, both computational and implementational details figure together in one mechanistic hierarchy. Therefore, it is natural to take relations of implementation to not have an explanatory role. Instead, medium-dependent details are taken to explain by decomposition of the phenomena. On the separate-hierarchies picture implementation can be considered to have a non-mechanistic explanatory role: it explains how the explanandum, as well as the computational hierarchy are implemented (see (Coelho Mollo, 2018)).

What about the view that computational explanations are both abstract and full-fledged mechanistic explanations? It would be difficult to see how the first solution in Fig. 6 can be consistent with it; if computational explanations are complete mechanistic explanations why do they require additional implementation details in the same mechanistic level of explanation? The second solution in Fig. 7 is not necessarily inconsistent with this view. For example, if one takes computational states and properties to have causal powers, then one can view the computational hierarchy as a hierarchy of complete mechanistic explanations. However, on this view the role of the implementational hierarchy still needs to be explicated. A possible implication is that the overall mechanistic picture is more complex: We have different mechanistic hierarchies that apply to different properties of the same objects/components. But under this picture any computational capacity has at least two hierarchical explanations, and it is not obvious which one of them should be considered *the* mechanistic explanation. A possible way to elucidate this complex picture is to maintain that the implementational hierarchy explains how the computational hierarchy is implemented, rather than how the cognitive capacity is performed (Coelho Mollo, 2018). On this view, the computational hierarchy is the mechanistic hierarchy which decomposes the cognitive capacity and the implementational hierarchy is an appendix which explain the implementation of the computation.

6. Some insights from reinforcement learning

It can be useful to examine the relation between the hierarchies in reinforcement learning. When considering the computational and implementational hierarchical models for reinforcement learning, which solution best describes the relation between these hierarchies? We believe that evidence in this case is mixed and can support both suggested solutions for the relation between the hierarchies. On the picture seen on Fig. 6, each level combines computation and implementation into one mechanistic explanation. Therefore, we would expect the scientific investigation of lower levels to include a physical decomposition of the higher level, as occurs in mechanistic explanations. However, in our example the scientific investigation of the implementation of the computational hierarchy searches for the implementation of variables at various levels of this hierarchy, such as the representations of action-value (Samejima *et al.*, 2005), RPE (Schultz, Dayan and Montague, 1997) and learning rate (α in eq. 1) (Behrens *et al.*, 2007). Often, the search for a lower-level variable such as the learning rate takes place in the absence of a scientifically supported neural correlate for the higher level computational variable of which it consists (In this case the calculation of action-value). Hence, the search for neural correlates here is more akin to searching for relations between two separate computational and implementational hierarchies than to physically decomposing mechanisms.

Moreover, scientific investigation of both hierarchies can and has been conducted separately. The Q-learning algorithm for reinforcement learning has been investigated both analytically (Watkins and Dayan, 1992) and behaviorally (Shteingart, Neiman and Loewenstein, 2013). These methods ignore the neural correlates of this model. Similarly, the basal ganglia have been investigated anatomically and functionally without addressing computational models for reinforcement learning (Hoshi *et al.*, 2005). This suggests that a framework of two hierarchies, as presented in Fig. 7, is the appropriate one in this case.

On the other hand, it can be argued that current scientific research is still preliminary and not indicative of the final form of a fully-fledged scientific explanation. Hints that such a form will include one combined mechanistic hierarchy can be found in the fact that scientific debates today about the plausibility of specific computational models of reinforcement learning often also appeal to the plausibility of the implementation of these models (Botvinick, Niv and Barto, 2009).

Moreover, findings of implementation of specific computational variables can be used to support or refute abstract computational models. Recall the three challenges to the computational model we presented in the section 3. The first one suggested that instead of learning the values of the actions, there is 'direct-policy' learning where the probability of choosing each action (i.e., the policy) is reevaluated at each step. However, the finding that striatal neurons represent the expected reward associated with each action (Samejima *et al.*, 2005) can be taken as support for the hypothesis that a Q-learning model is implemented in the brain, rather than a 'direct-policy' model[6].

The finding in (O'Doherty *et al.*, 2004) that striatal neurons can be divided into 'actor' and 'critic' modules can be used as evidence in the second challenge: whether the action selection and action reevaluation modules can be separated into 'actor' and 'critic'. It is also increasingly popular to suggest computational models that are informed by the structure of neural networks, with the purpose of suggesting models that are more biologically plausible (Mnih *et al.*, 2016). Note that, even though physical structures are used as evidence in this debate, the questions pertain to the architecture of the abstract computational model, which can be implemented both in computers and in brains.

Given these examples it can be argued that the practice of developing a complete explanation at each level of the explanatory hierarchy involves a close and reciprocal relation between the computational models and their possible implementation, and that computational models are not considered explanations until they have been shown to be implemented in the brain. This suggests that computation and implementation belong together in one level of the explanation. Therefore, the pictures presented in Figs. 6-7 are both still possible regarding this example.

However, when considering whether computational descriptions are merely sketches of mechanisms, on the interpretation of sketches as partial descriptions of implementation, the evidence is more conclusive. We see that, in our example of reinforcement learning, evidence from scientific practice is strongly against the view of computational models as sketches. Moreover, scientific practice tends to take implementational details to explain the implementation of the computational model rather than the cognitive capacity directly. Often, when findings of neural correlates of reinforcement learning models are reported, they are reported as discoveries about the implementation of these models. Hence, such findings are taken to answer questions about how, and whether a specific computational model is implemented in the brain and they do not attempt to explain reinforcement learning

[6] But see (Elber-Dorozko and Loewenstein, 2018)

(or decision making in general) without appeal to some computational model. Perhaps the strongest indication for this is in experiments where there is some causal intervention on brain areas and behavioral changes are measured. If computational models are merely partial descriptions of implementation, they will be unnecessary in the interpretation of causal experiments, where the causal structure is already described in the results of the experiment. However, often, results in such experiments are interpreted in the framework of a computational model of reinforcement learning (Tai *et al.*, 2012; Wang, Miura and Uchida, 2013; Lee *et al.*, 2015). For example, (Tai *et al.*, 2012) find that stimulation of striatal neurons causes a bias in choices, and they interpret these results by saying that stimulation of striatal neurons mimics changes in action-value. Hence, instead of utilizing the causal finding to explain the behavior of the subjects, (Tai *et al.*, 2012) use their finding as an indication of implementation of action-value – a computational variable. Such a computational interpretation to causal results is difficult to explain if computational models are taken to be merely partial descriptions of causal mechanisms and is much more in line with the view that computational models have a unique explanatory value. Moreover, this scientific practice can be taken to support the claim that implementational details are taken to explain the computational model rather than the cognitive capacity itself.

For this reason, we believe that our example does not support the view that computational models are partial descriptions or that computational models are explanatory only because they describe causal relations. Instead, this reinforcement learning example is more consistent with the view that computational properties play an invaluable role in the explanation of cognitive phenomena.

Nonetheless, reinforcement learning is just one example of computational models of cognitive capacities. Future investigation of other computational models will be telling regarding the relation between computation and implementation.

7 Conclusions

After raising the problem of how computational explanations integrate in the mechanistic hierarchy, we analyzed reinforcement learning as an example of a computational model in neuroscience and reviewed two possible pictures of the relations between computation and implementation in the mechanistic hierarchy. On the one-hierarchy picture computational and their implementational properties reside in the same level(s) of explanation. On the two-hierarchy picture computational and implementational properties reside in different computational and implementational hierarchies. We concluded that both pictures are possible regarding the reinforcement learning example, but that scientific practice does not align with the view that computational models are merely mechanistic sketches.

Bibliography

Bechtel, W. and Shagrir, O. (2015) 'The Non-Redundant Contributions of Marr's Three Levels of Analysis for Explaining Information-Processing Mechanisms', *Topics in Cognitive Science*, 7, pp. 312–322.

Behrens, T. E. J. *et al.* (2007) 'Learning the value of information in an uncertain world', *Nature Neuroscience*, 10, pp. 1214–1221. doi: 10.1038/nn1954.

Boone, W. and Piccinini, G. (2016) 'The cognitive neuroscience revolution', *Synthese*, 193, pp. 1509–1534.

Botvinick, M. M. (2012) 'Hierarchical reinforcement learning and decision making', *Current Opinion in Neurobiology*, 22, pp. 956–962. doi: 10.1016/j.conb.2012.05.008.

Botvinick, M., Niv, Y. and Barto, A. (2009) 'Hierarchically organized behavior and its neural foundations: A reinforcement learning perspective', *Cognition*, 113, pp. 262–280. doi: 10.1016/j.cognition.2008.08.011.Hierarchically.

Chirimuuta, M. (2014) 'Minimal models and canonical neural computations: the distinctness of computational explanation in neuroscience', *Synthese*, 191, pp. 127–153.

Chirimuuta, M. (2018) 'Explanation in Computational Neuroscience: Causal and Non-causal', *The British Journal for the Philosophy of Science*, 69, pp. 849–880. doi: 10.1093/bjps/axw034.

Coelho Mollo, D. (2018) 'Functional individuation, mechanistic implementation: the proper way of seeing the mechanistic view of concrete computation', *Synthese*, 195, pp. 3477–3497. doi: 10.1007/s11229-017-1380-5.

Craver, C. F. (2016) 'The Explanatory Power of Network Models', *Philosophy of Science*, 83, pp. 698–709.

Craver, C. F. and Povich, M. (2017) 'The directionality of distinctively mathematical explanations', *Studies in History and Philosophy of Science*, 63, pp. 31–38. doi: 10.1016/j.shpsa.2017.04.005.

Cummins, R. (1983) *The Nature of Psychological Explanation*. MIT Press.

Cummins, R. (2000) '"How does it work?" vs. "What are the laws?" Two conceptions of psychological explanation.', in Keil, F. and Wilson, R. A. (eds) *Explanation and Cognition*. MIT Press, pp. 117–145.

Dewhurst, J. (2018) 'Individuation without Representation', *The British Journal for the Philosophy of Science*, 69, pp. 103–116. doi: 10.1093/bjps/axw018.

Doya, K. (2000) 'Complementary roles of basal ganglia and cerebellum in learning and motor control', *Current Opinion in Neurobiology*, 10, pp. 732–739. doi: 10.1016/S0959-4388(00)00153-7.

Doya, K. (2008) 'Modulators of decision making', *Nature Neuroscience*, 11, pp. 410–416. doi: 10.1038/nn2077.

Egan, F. (2017) 'Function-Theoretic Explanation and Neural Mechanisms', in Kaplan, D. M. (ed.) *Explanation and Integration in Mind and Brain Science*. Oxford University Press, pp. 145–163.

Elber-Dorozko, L. and Loewenstein, Y. (2018) 'Striatal action-value neurons reconsidered', *eLife*, 7, p. e34248. doi: 10.7554/eLife.34248.

Fodor, J. (1968) *Psychological Explanation: An Introduction To The Philosophy Of Psychology*. Random House.

Fodor, J. (1980) 'Methodological solipsism considered as a research strategy in cognitive psychology', *Behavioral and Brain Sciences*, 3, pp. 63–73.

Fodor, J. (1994) *The elm and the expert*. MIT Press.

Fodor, J. A. (1975) *The Language of Thought*. Harvard University Press.

Haimovici, S. (2013) 'A Problem for the Mechanistic Account of Computation', *Journal of Cognitive Science*, 14, pp. 151–181.

Harbecke, J. (under review) 'Multiple Level Hierarchies in Cognitive Neuroscience and the Mechanistic-Computational Model of Explanation'.

Haugeland, J. (1981) 'Semantic Engines: an Introduction to Mind Design', in Haugeland, J. (ed.) *Mind Design, philosophy, Psychology, Artificial Intelligence*. MIT Press.

Hollerman, J. R. and Schultz, W. (1998) 'Dopamine neurons report an error in the temporal prediction of reward during learning', *Nature neuroscience*, 1, pp. 304–9. doi: 10.1038/1124.

Hoshi, E. *et al.* (2005) 'The cerebellum communicates with the basal ganglia', *Nature Neuroscience*, 8, pp. 1491–1493. doi: 10.1038/nn1544.

Huneman, P. (2010) 'Topological explanations and robustness in biological sciences', *Synthese*, 177, pp. 213–245.

Ito, M. and Doya, K. (2009) ‘Validation of Decision-Making Models and Analysis of Decision Variables in the rat basal ganglia’, *The Journal of Neuroscience*, 29(31), pp. 9861–9874. doi: 10.1523/JNEUROSCI.6157-08.2009.

Ito, M. and Doya, K. (2011) ‘Multiple representations and algorithms for reinforcement learning in the cortico-basal ganglia circuit’, *Current Opinion in Neurobiology*, 21, pp. 368–373. doi: 10.1016/j.conb.2011.04.001.

Kable, J. W. and Glimcher, P. W. (2009) ‘The Neurobiology of Decision: Consensus and Controversy’, *Neuron*. Elsevier Inc., 63(6), pp. 733–745. doi: 10.1016/j.neuron.2009.09.003.

Kandel, E. R. *et al.* (2013) *Principles of Neural Science*. Fifth. New York: McGraw-Hill.

Kaplan, D. M. (2011) ‘Explanation and description in computational neuroscience’, *Synthese*, 183, pp. 339–373.

Kaplan, D. M. (2017) ‘Neural computation, multiple realizability, and the prospects for mechanistic explanation’, in Kaplan, D. M. (ed.) *Explanation and Integration in Mind and Brain Science*. Oxford University Press, pp. 164–189.

Kaplan, D. M. and Craver, C. F. (2011) ‘The Explanatory Force of Dynamical and Mathematical Models in Neuroscience : A Mechanistic Perspective’, *Philosophy of Science,* 78, pp. 601–627.

Lange, M. (2013) ‘What Makes a Scientific Explanation Distinctively Mathematical?’, *The British Journal for the Philosophy of Science*, 64, pp. 485–511. doi: 10.1093/bjps/axs012.

Lee, E. *et al.* (2015) ‘Injection of a Dopamine Type 2 Receptor Antagonist into the Dorsal Striatum Disrupts Choices Driven by Previous Outcomes , But Not Perceptual Inference’, *The Journal of Neuroscience*, 35, pp. 6298–6306. doi: 10.1523/JNEUROSCI.4561-14.2015.

Li, J. and Daw, N. D. (2011) ‘Signals in Human Striatum Are Appropriate for Policy Update Rather than Value Prediction’, *Journal of Neuroscience*, 31, pp. 5504–5511. doi: 10.1523/JNEUROSCI.6316-10.2011.

Marr, D. (1982) *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. MIT Press.

Milkowski, M. (2013) *Explaining the Computational Mind*. MIT Press.

Mnih, V. *et al.* (2016) ‘Human-level control through deep reinforcement learning’, *Nature*, 518, pp. 529–533. doi: 10.1038/nature14236.

Mongillo, G., Shteingart, H. and Loewenstein, Y. (2014) ‘The misbehavior of reinforcement learning’, *Proceedings of the IEEE*, 102, pp. 528–541. doi: 10.1109/JPROC.2014.2307022.

O’Doherty, J. P. *et al.* (2004) ‘Dissociable Role of Ventral and Dorsal Striatum in Instrumental Conditioning’, *Science*, 304, pp. 452–454. doi:

10.1126/science.1094285.

Piccinini, G. (2015) *Physical Computation: A Mechanistic Account*. Oxford University Press.

Piccinini, G. and Bahar, S. (2013) ‘Neural Computation and the Computational Theory of Cognition’, *Cognitive Science*, 34, pp. 453–488.

Piccinini, G. and Craver, C. F. (2011) ‘Integrating psychology and neuroscience: functional analyses as mechanism sketches’, *Synthese*, 183, pp. 283–311.

Rathkopf, C. (2015) ‘Network representation and complex systems’, *Synthese*, 195, pp. 55–78.

Rusanen, A. and Lappi, O. (2016) ‘On computational explanations’, *Synthese*, 193, pp. 3931–3949.

Samejima, K. *et al.* (2005) ‘Representation of Action-Specific Reward Values in the Striatum’, *Science*, 310, pp. 1337–1340. doi: 10.1126/science.1115270.

Schultz, W., Dayan, P. and Montague, P. R. (1997) ‘A Neural Substrate of Prediction and Reward’, *Science*, 275, pp. 1593–1599. doi: 10.1126/science.275.5306.1593.

Shagrir, O. (2006) ‘Why we view the brain as a computer’, *Synthese*, 153, pp. 393–416.

Shagrir, O. (2016) ‘Advertisement for the Philosophy of the Computational Sciences’, in Paul Humphreys (ed.) *The Oxford Handbook of Philosophy of Science*. Oxford University Press, pp. 15–42.

Shagrir, O. and Bechtel, W. (2017) ‘Marr’s Computational Level and Delineating Phenomena’, in Kaplan, D. M. (ed.) *Explanation and Integration in Mind and Brain Science*. Oxford University Press, pp. 190–214.

Shapiro, L. A. (2017) ‘Mechanism or Bust? Explanation in Psychology’, *The British Journal for the Philosophy of Science*, 68, pp. 1037–1059.

Shteingart, H. and Loewenstein, Y. (2014) ‘Reinforcement learning and human behavior’, *Current Opinion in Neurobiology*, 25, pp. 93–98. doi: 10.1016/j.conb.2013.12.004.

Shteingart, H., Neiman, T. and Loewenstein, Y. (2013) ‘The role of first impression in operant learning’, *Journal of Experimental Psychology: General*, 142, pp. 476–488. doi: 10.1037/a0029550.

Sprevak, M. (2010) ‘Computation, individuation, and the received view on representation’, *Studies in History and Philosophy of Science Part A*, 41, pp. 260–270.

Stich, S. (1983) *From Folk Psychology to Cognitive Science: The Case Against Belief*. MIT Press.

Sutton, R. S. and Barto, A. G. (1998) *Reinforcement Learning: An Introduction*. MIT

Press.

Tai, L.-H. *et al.* (2012) ‘Transient stimulation of distinct subpopulations of striatal neurons mimics changes in action value’, *Nature neuroscience*, 15, pp. 1281–9. doi: 10.1038/nn.3188.

Wang, A. Y., Miura, K. and Uchida, N. (2013) ‘The dorsomedial striatum encodes net expected return, critical for energizing performance vigor.’, *Nature neuroscience*, 16, pp. 639–47. doi: 10.1038/nn.3377.

Watkins, C. J. C. H. and Dayan, P. (1992) ‘Q-Learning’, *Machine Learning*, 8, pp. 279–292.

Weiskopf, D. A. (2011) ‘Models and mechanisms in psychological explanation’, *Synthese*, 183, pp. 313–338.