# Abstraction and Representation in Living Organisms: When Does a Biological System Compute?

**Dominic Horsman, Viv Kendon, Susan Stepney and J. P. W. Young**

**Abstract** Even the simplest known living organisms are complex chemical processing systems. But how sophisticated is the behaviour that arises from this? We present a framework in which even bacteria can be identified as capable of representing information in arbitrary signal molecules, to facilitate altering their behaviour to optimise their food supplies, for example. Known as Abstraction/Representation theory (AR theory), this framework makes precise the relationship between physical systems and abstract concepts. Originally developed to answer the question of when a physical system is computing, AR theory naturally extends to the realm of biological systems to bring clarity to questions of computation at the cellular level.

## 1 Introduction

The language of information processing is widespread in biology. From DNA replication to nerve impulses to brain activity, systems are frequently talked of as storing and processing data, and even as performing intrinsic computation. It has previously been difficult, however, to argue that this is more than an analogy: is there, in fact, computation happening in biological systems? Is it possible to model such systems

DH published previously as Clare Horsman.

D. Horsman
Department of Computer Science, University of Oxford, Oxford, UK

D. Horsman · V. Kendon
Department of Physics, Durham University, Durham, UK

S. Stepney (✉)
Department of Computer Science and York Centre for Complex Systems Analysis, University of York, York, UK
e-mail: susan.stepney@york.ac.uk

J.P.W. Young
Department of Biology, University of York, York, UK



G. Dodig-Crnkovic and R. Giovagnoli (eds.), *Representation and Reality in Humans, Other Living Organisms and Intelligent Machines*, Studies in Applied Philosophy, Epistemology and Rational Ethics 28, DOI 10.1007/978-3-319-43784-2_6

as computations? Is it even the case that the ability to compute is so basic to living organisms that we can use it as a definition of life?

Biological and computational processes share many similarities, such as: the encoding of process data in proteins; signal transduction from input to processed output [3]; cells mimicking computers [22]; similarities between computer networks and biological distributed systems and viruses [18].

Computational biology aims to use these similarities to model biological systems computationally, with the dual aims of better understanding their basic processes, and of producing biological systems artificially. Computer simulations of biological systems have inspired key areas in machine intelligence [23, Chap. 8]. Organisms such as bacteria [26], slime moulds [1], and hybrid biological/silicon devices [8] have been closely studied as potential non-standard computational devices.

Here we make precise this previously informal relationship between biological and computational processes. We locate computing within the broader category of *representational activity*, and give conditions for when a biological system is making fundamental use of representation to perform a range of tasks including engineering, communication and signalling and computing. We use the formal framework of the recently developed *Abstraction/Representation Theory* (AR theory), introduced in [13] and extended in [11]. AR theory was introduced to give a rigorous characterisation of the relationship between abstract representation and physical system, primarily in the context of determining when a physical system is being used as a computer. It was developed with non-standard human-designed computational devices in mind, and has already been put to good use determining whether representational activity, including computation, is occurring in unconventional computing substrates [12, 14].

AR theory's ability to deal with representation as a whole, and computing outside standard silicon-based digital models, gives it the capacity to extend further to considerations of the computing/representational activities of organic systems. There are, however, challenges to using AR theory with respect to biological systems that do not occur when considering devices that have been deliberately engineered.

At the centre of AR theory is the *representation relation*, mediating between physical and abstract objects. This permits the encoding and decoding of abstract information in physical systems, as is necessary for communication and computing. Physical states are represented abstractly, and in certain tightly-defined situations this representation relation from physical to abstract can effectively be 'reversed' (by engineering the system) to an instantiation relation from abstract to physical. By this means abstract information can be instantiated in a physical system.

In the human-user context, the representation relation is both determined by and located in easily-identifiable intelligent and conscious *representational entities*, namely the human users taking part in scientific, technological or computational activities. A representational entity is required for any representational activity to take place, and is required to be a physical entity that is part of the representational system. This distinguishes, in AR theory, a system being used as part of representational activity, and one that is 'going about its own business'. This also allows for the distinction between a system being used as a computer, and one that is *post hoc*

represented as computing: in the first case the representational entity is part of the system under consideration, and in the second it is not [13].

In natural biological systems, the challenge in identifying representational activity is both to identify the representational entity present, and to determine that representation is occurring within the system in the absence of a conscious or intelligent user who can inform us of this fact. The type of representational activity (engineering, communication, computing) is then a further property of the system to be determined.

Here we investigate representational activity (including signalling and computing) in low-level biological systems. We give methods for determining the presence of representational activity in the absence of high-level representational entities, and pose the question: how simple can a representational entity be? We find evidence for representation in systems far removed from conscious entities, and show that representation does not require structures as complex as a brain or collections of neurons. We see that key to determining representational activity (as opposed to 'manipulation of stuff') is the identification of *arbitrariness* within representation. That is, that the instantiation of information occurs in a one-to-many mapping between abstract and physical, so that the *same* outcome could have occurred using a *different* physical material or process. From the point of view of the biological process, it is the abstract process that is key in determining the correct physical outcome, not its particular instantiation.

We analyse three specific biological examples using AR theory to illustrate the presence and type of representational activity. By a close consideration of three biological processes—bacterial chemotaxis, the genetic code and photosynthesis—we show how the arbitrariness of information representation allows us to determine whether these systems are engaged in representational activity or not—yes for the first and second, no for the third—and we identify specific examples of computing happening in biological systems. We find that low-level biological systems use representation as an integral part of their behaviour and their interaction with their environment, and use this ability in certain situations to store and manipulate information in an equivalent process to that used by human-designed computers. Abstraction and representation can now be seen to be fundamental processes engaged in by most, maybe all, living organisms.

## 2 The Framework

AR theory was introduced in [13] and extended and formalised in [11]. These works should be consulted for the full physical, philosophical and formal background to the framework. It is a framework in which science, engineering/technology, computing and communication/signalling are all defined as *representational activity* requiring the fundamental use of the representation relation in order to define their operation.

AR theory was developed to answer the specific questions of when a physical system is computing [13]. This turns out to be a question about the relationship between

an abstract object (a computation) and a physical object (a computer). What is needed is a formal language of relations, not from mathematical objects to mathematical objects (as is usual in mathematics and theoretical computer science), but between physical objects and those in the abstract domain. The core of AR theory is the representation relation, mapping from physical objects to abstract objects. Experimental science, engineering and computing all require the interplay of abstract and physical objects via representation in such a way that formal descriptive diagrams commute: the same result can be gained through either physical or abstract evolutions. The key result of [13] defined computing as *the use of a physical system to predict the outcome of an abstract evolution*.

## 2.1 Formalising Representation

AR theory identifies objects in the domain of physical systems, abstract objects (including mathematical and logical entities) and the representation relation which mediates between the two. The distinction between the two spaces, abstract and physical, is fundamental in the theory, as is their connection *only* by the (directed) representation relation. An intuitive example is given in Fig. 1: a physical switch is represented by an abstract bit, where in this case it is zero for up, one for down.

An example of an object in the domain of physical entities is a *computer*. It has, usually, internal degrees of freedom, and a physical evolution that connects initial input and final output states. An example of an abstract object is a *computation*, which is a set of objects and relations as described in the logical formalisms of theoretical computer science. Likewise, an object such as a bacterium is a physical entity, and its theoretical representation within biology is an object in the domain of abstract entities. In what follows, we use bold font to indicate where an object $\mathbf{p}$ or evolution $\mathbf{H}$ is physical; and italic font for abstract objects represented within equations, for example, in giving the abstract object $m_{\mathbf{p}}$.

The elementary *representation relation* is the directed map from physical to abstract objects, $\mathcal{R} : \mathbf{P} \to M$, where $\mathbf{P}$ is the set of physical objects, and $M$ is the set of abstract objects. When two objects are connected by $\mathcal{R}$ we write them as $\mathcal{R} : \mathbf{p} \to m_{\mathbf{p}}$. The abstract object $m_{\mathbf{p}}$ is then said to be the *abstract representation* of

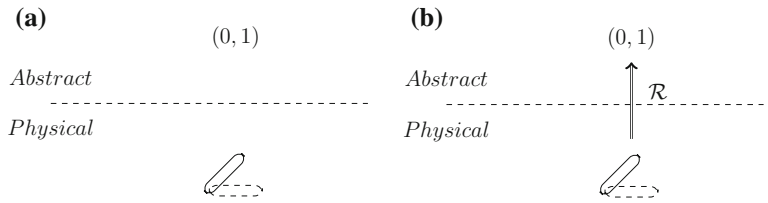


**Fig. 1** Basic representation. **a** Spaces of abstract and physical objects (here, a switch with two settings and a binary digit). **b** The directed representation relation $\mathcal{R}$ mediating between the spaces

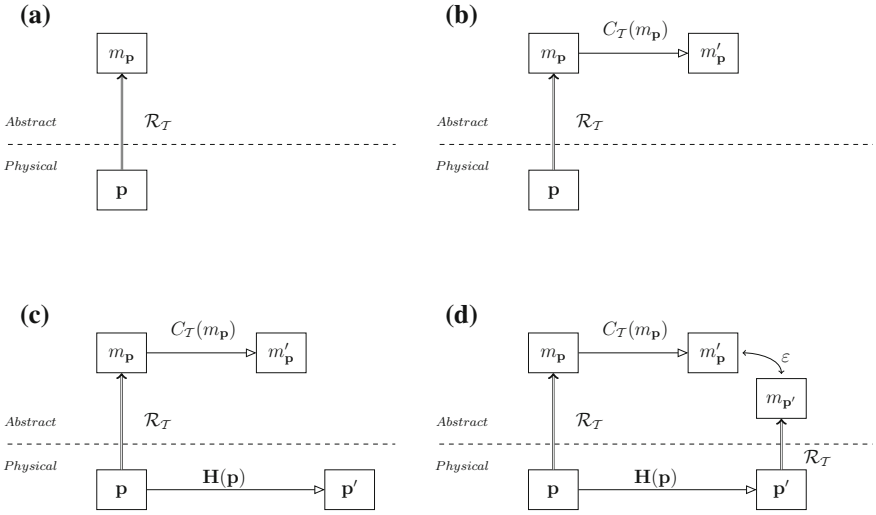


**Fig. 2** Parallel evolution of abstract evolution (e.g. an algorithm) and potential physical computing device. **a** The basic representational triple, $\langle \mathbf{p}, \mathcal{R}, m_{\mathbf{p}} \rangle$: physical system $\mathbf{p}$ is represented abstractly by $m_{\mathbf{p}}$ using the modelling representation relation $\mathcal{R}_{\mathcal{T}}$ of theory $\mathcal{T}$. **b** Abstract dynamics $C_{\mathcal{T}}(m_{\mathbf{p}})$ give the evolved abstract state $m'_{\mathbf{p}}$. **c** Physical dynamics $\mathbf{H}(\mathbf{p})$ give the final physical state $\mathbf{p}'$. **d** $\mathcal{R}_{\mathcal{T}}$ is used again to represent $\mathbf{p}'$ as the abstract output $m_{\mathbf{p}'}$, $|m_{\mathbf{p}} - m_{\mathbf{p}'}| = \varepsilon$. (Adapted from [13])

the physical object $\mathbf{p}$, and together they form one of the basic composites of AR theory, the *representational triple* $\langle \mathbf{p}, \mathcal{R}, m_{\mathbf{p}} \rangle$. The basic representational triple is shown in Fig. 2a.

Similarly, abstract evolution takes abstract objects to abstract objects, which we write as $C : M \to M$. An individual example is shown in Fig. 2b, for the mapping $C(m_{\mathbf{p}})$ taking $m_{\mathbf{p}} \to m'_{\mathbf{p}}$. The corresponding physical evolution map is given by $\mathbf{H} : \mathbf{P} \to \mathbf{P}$. For individual elements in Fig. 2c this is $\mathbf{H}(\mathbf{p})$, which takes $\mathbf{p} \to \mathbf{p}'$.

In order to reach the next key concept in AR theory, we now apply the representation relation to the outcome state of the physical evolution to give its abstract representation $m_{\mathbf{p}'}$, Fig. 2d. We now have two abstract objects, $m'_{\mathbf{p}}$ and $m_{\mathbf{p}'}$. For some (problem-dependent) error quantity $\varepsilon$ and norm $|.|$, if $|m_{\mathbf{p}'} - m'_{\mathbf{p}}| \leq \varepsilon$ then the diagram (Fig. 2d) *commutes*. Commuting diagrams are fundamental to the use of AR theory. If a set of abstract and physical objects form a commuting diagram under representation, then $m_{\mathbf{p}}$ is a *faithful abstract representation* of physical system $\mathbf{p}$ for the evolutions $C(m_{\mathbf{p}})$ and $\mathbf{H}(\mathbf{p})$.

The main reason why commuting diagrams are important, along with faithful abstract representations for physical systems, is that the final state of a physical object undergoing evolution can be known either by tracking the physical evolution and then representing the output abstractly, or by theoretically evolving the representation of the system. In the first case, the 'lower path' of a commuting diagram is followed; in the latter, the 'upper path'. Finding out which diagrams commute is the business

of basic experimental science; and once commuting diagrams have been established they can be exploited through engineering and technology.

## 2.2 Theory and Experiment

In experimental science, a test for commutation of a diagram involves producing a controlled physical setup (the experiment) that has both an abstract representation $\mathcal{R}$ and an abstract prediction of how it will behave, $C$. The physical system $\mathbf{p}$ is evolved under the physical experimental dynamics $\mathbf{H}$, and the outcome compared to the theoretical prediction. If they coincide within the error tolerance of the experiment and the desired outcome confidence, then the diagram commutes.

This is not, of course, the *purpose* of an experiment. Experiments are designed in order to test not a single scenario but a *theory* of a physical system. A physical **theory**, $\mathcal{T}$, is a set of representation relations $\mathcal{R}_\mathcal{T}$ for physical objects, a domain of such objects for which it is purported to be valid, and a set of abstract predictive dynamics for the output of the representations, $m_\mathbf{p}$, $C(m_\mathbf{p})$. If a theory supports commuting diagrams for all scenarios in which it has been both defined and tested, then it is a *valid* theory. A physical system or device that is both well tested and well understood will in general have a large number of commuting diagrams supporting it.

## 2.3 Engineering

The representation relation defined so far is directed, from physical to abstract objects. This is *modelling*: giving an abstract representation of a physical object. The question can now be posed: is it possible to give a reversed representation relation, an *instantiation* relation? This will not be a basic relation in the same way as the ordinary (modelling) representation relation is basic: abstract representation can be given for any physical object (this is language), but there are plenty of abstract objects that do not have a physical instantiation ('unicorn', 'free lunch', etc.). Only in very specific circumstances can an instantiation relation $\widetilde{\mathcal{R}}_\mathcal{T}$ be given for a theory $\mathcal{T}$.

To find these circumstances, consider again the 'upper' and 'lower' paths of a commuting diagram, $(\mathbf{p}_0 \to m_{\mathbf{p}_0} \to m'_{\mathbf{p}_0})$ and $(\mathbf{p}_0 \to \mathbf{p} \to [m_\mathbf{p} = m'_{\mathbf{p}_0}])$ respectively. Between them, these paths describe the process of finding some $\mathbf{p}_0$ such that when it is subjected to the physical process $\mathbf{H} : \mathbf{p}_0 \to \mathbf{p}$ it becomes the physical system $\mathbf{p}$ whose abstract representation is $m_\mathbf{p}$. In other words, if both paths are present and form a commuting diagram, the theory $\mathcal{T}$ can be used to *engineer* system $\mathbf{p}$ from system $\mathbf{p}_0$ given a desired abstract specification $m_\mathbf{p}$: this is the *instantiation relation* $\widetilde{\mathcal{R}}_\mathcal{T}$, Fig. 3.

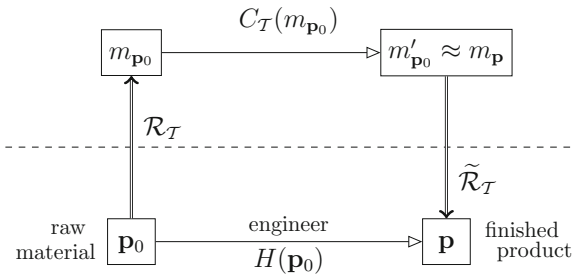


**Fig. 3** Engineering the system **p**, using the instantiation relation $\widetilde{\mathcal{R}}_{\mathcal{T}}$

A use of the instantiation relation can be seen as a counterfactual use of the representation relation: which physical system, when represented abstractly, would give the abstract representation that we are trying to instantiate? The method by which it is achieved will vary considerably given different scenarios: trial and error, abstract reasoning, numerical simulation, etc. What connects these methods is that they are not straightforward: it is generally a skilful and cumulative process to reverse a representation relation.

## *2.4 Computation*

A commuting diagram in the context of *computation* connects the physical computing device, **p**, and its abstract representation $m_{\mathbf{p}}$. It makes integral use of the instantiation relation: a computer is an engineered device. $m_{\mathbf{p}}$ can be a number of different abstract representations; a common one draws from the set of binary strings. The abstract evolution is then the (binary) program to be run on the computer, and the physical evolution is how the state of the computer changes during the program (change of voltages, etc.). The full commuting diagram describes the parallel evolution of physical computer and abstract algorithm, connected via the representation given by the theory of the computing device, $\mathcal{R}_{\mathcal{T}}$.

The AR description of physical computing is not simply the parallel evolution of physical and abstract. The *compute cycle* starts from a set of abstract objects: the program and initial state that are to be computed. The most important use of a computing system is when the abstract outcome $m'_{\mathbf{p}}$ is unknown: when computers are used to solve problems. Consider as an example the use of a computer to perform the binary arithmetical problem $01 + 10$. If the outcome were unknown, and the computing device being used to compute it, the final abstract state, $m'_{\mathbf{p}} = (11)$, would not be evolved abstractly. Instead, confidence in the technological capabilities of the computer would enable the user to reach the final, abstract, output state $m_{\mathbf{p}'} = m'_{\mathbf{p}}$ using the physical evolution of the computing device alone.

This use of a physical computer is the compute cycle, Fig. 4: the use of a physical system (the computer) to predict the outcome of a computation (an abstract evolution).

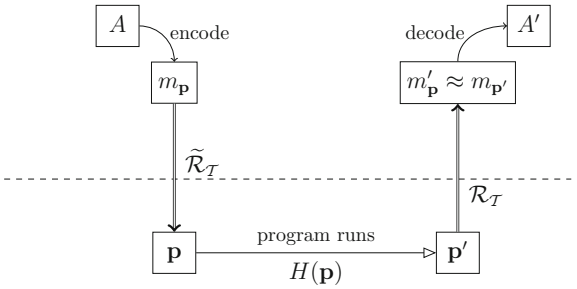


**Fig. 4** Physical computing in which an abstract problem $A$ is encoded into the model $m_{\mathbf{p}}$ of the computer $\mathbf{p}$, then instantiated into $\mathbf{p}$; the computer calculates via $\mathbf{H}(\mathbf{p})$, evolving into $\mathbf{p}'$, from which the representation relation is used to obtain the model $m_{\mathbf{p}'} \simeq m'_{\mathbf{p}}$, from which the output of the computation $A'$ can be decoded

## *2.5 Encoding and Representation*

An important element in the AR analysis of computing is the integral use of *encoding and decoding* in the compute cycle. These bear a close resemblance to the representation and instantiation relations, and we will see that in certain circumstances that they can be composed. However, unlike representation, encoding and decoding maps live entirely above the abstract/physical dividing line, and take abstract objects to abstract objects.

Encoding as the first step in the compute cycle embeds the computation to be performed into the abstract specification of the physical computer. This stage, frequently overlooked when analysing computation, is necessary in order to translate between the language of the problem specification (for human users, frequently linguistic) and that of the input interface of the device. Similarly, the decoding step is fundamentally necessary in order to translate the answer from the abstract representation of the end-state of the physical computer.

Encoding and decoding are, in general, fully composable: multiple encodings can be used within a system, and we can always consider a combination of encodings to be itself a single encoding (likewise for decodings). In other words, given a series of encodings $\gamma_i$, we can define the result of applying all of them in turn to be a single encoding, $\gamma = \gamma_1 \circ \gamma_2 \circ \ldots$. Encodings can also be composed with representation: encoding the abstract representation of a physical system is equivalent to representing it in a different manner, $\mathcal{R}_{\mathcal{T}} \circ \gamma = \mathcal{R}'_{\mathcal{T}}$. A single representation can also generally be decomposed in this way into another representation (often a simpler one) and an encoding. Encodings can in this way often be dispensed with *notationally* by rolling them in to the definition of a representation; care must be taken, however, to ensure that important elements of diagrams are not thereby obscured, because encoding and decoding can come with significant computational overheads. The converse is not possible: representation cannot fully be replaced with encoding or decoding. At some stage in between the physical system and the abstract problem, a representation relation *must* be used to cross the line between abstract and physical.

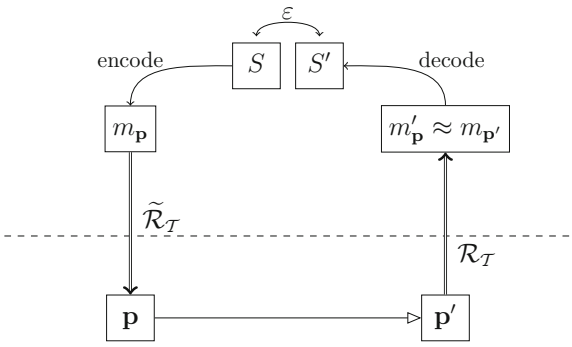


**Fig. 5** A signal $S$ instantiated into a physical carrier $\mathbf{p}$

## *2.6 Signalling*

Science, engineering and computing all build on each other within AR theory: faithful abstract scientific representations and good device theories are needed for engineering, and the instantiation relation and commuting engineering cycles are required in order to have a functioning computer. There is another important category of representational activity that sits alongside computing at the top of this stack: communication or, alternatively, signalling.

Signalling entails the encoding of a signal into a carrier, and then its decoding back to the original signal, within some $\varepsilon$ error tolerance, Fig. 5. This demonstrates how it can be considered as a specific type of computation, where the abstract evolution to be 'predicted' is the identity operation (that is, the abstract output is equal to the abstract input); hence the diagram ends at the same abstract output, the signal $S \approx S'$, as it begins. The physical carrier starts in state $\mathbf{p}$ and ends in state $\mathbf{p}'$; these states are typically separated either spatially or temporally. Furthermore, a given signal may be carried by several different physical substrates in the course of a single communication (for example, electrons in copper, and photons in fibre); what is important is that the correct *abstract* signal can be decoded at the end.

Signalling can therefore be viewed as a simpler form of representational activity than full computing, but requiring the same key elements of AR theory. Engineering is necessary to allow instantiation of an abstract signal in the carrier, and so good theories of the signalling device (and commuting 'science' diagrams) are required. These good theories may be discovered through the process of science, in the case of human-engineered signalling, or through evolution, in the case of *intrinsic* computation (see Sect. 3).

The blurred lines between communication and computation are familiar within Computer Science (where communication tasks between 'Alice' and 'Bob' can be used to perform some even quite complex computations). With AR theory, it becomes clearer that this boundary is similarly blurred when considering when biological system are performing signalling operations (a relatively common

consideration for biologists) and when they are computing (a rather less universally accepted situation within such systems).

## 2.7 Arbitrariness of Encoding

The consideration of signalling brings out a key aspect of computing and communication that is foregrounded by AR theory. An abstract object, such as a computation or a signal, is encoded into a physical system: a physical computer or signal carrier. An important aspect of this implementation is that there is not a necessary one:one mapping between abstract and physical objects. A given signal may be carried by many different physical systems, a situation familiar to human language users (a word can be written, transmitted through speech, or through physical sign language, to name just a few possibilities). This also occurs frequently in the biological domain. For example some plants coevolve a communication language with other organisms; a particularly sophisticated case of this is where plants use chemical carriers to implement a signal to their predators' predators [24]; different plants use different chemicals.

The converse can also be true: a single physical system can be represented by multiple different abstract representations. These abstractions may be related in a particular manner that occurs frequently in Computer Science, where an abstract computation or communication is encoded into a physical computing/signalling device. There can be a number of models at different levels of abstraction, each representing the same physical computer/signal carrier, connected through a *refinement relation* [5, 10]. That is, the same abstract computational object has several more concrete encodings, each progressively more 'refined' or 'reified'. These are all abstract as opposed to physical, but some of them, the more concrete ones, are somehow deemed 'closer' to the physical system. That is, the eventual instantiation relation, that maps down from these abstract states to a physical implementation, is somehow 'simpler' or 'more natural' for these concrete representations.

In refinement theory, an abstract object can be refined in many different ways. For example, an abstract set may be refined to a (still abstract, but more concrete) list, array, linked list, or other data structure. A data word may be encoded in a string of bytes in big-endian or little-endian order in memory (most significant byte first, or last). Another example is how an alphabetical character may be encoded as a string of eight bits in ASCII or EBCDIC formats. When larger character sets that require more than eight bits for encoding are considered, many more possibilities exist.

It is refinement theory that allows us to connect the two seemingly different situations within computing/signalling of a single abstract object encoded in multiple physical implementations, and a single physical implementation supporting multiple abstract representations. It is simplest to consider these within a signalling scenario, Fig. 6.

Figure 6a shows a refinement stack: a single physical carrier $\mathbf{p}$ has abstract representation $m_{\mathbf{p}}$, which is a refinement through the function $\gamma_2$ of the abstract object $n$,

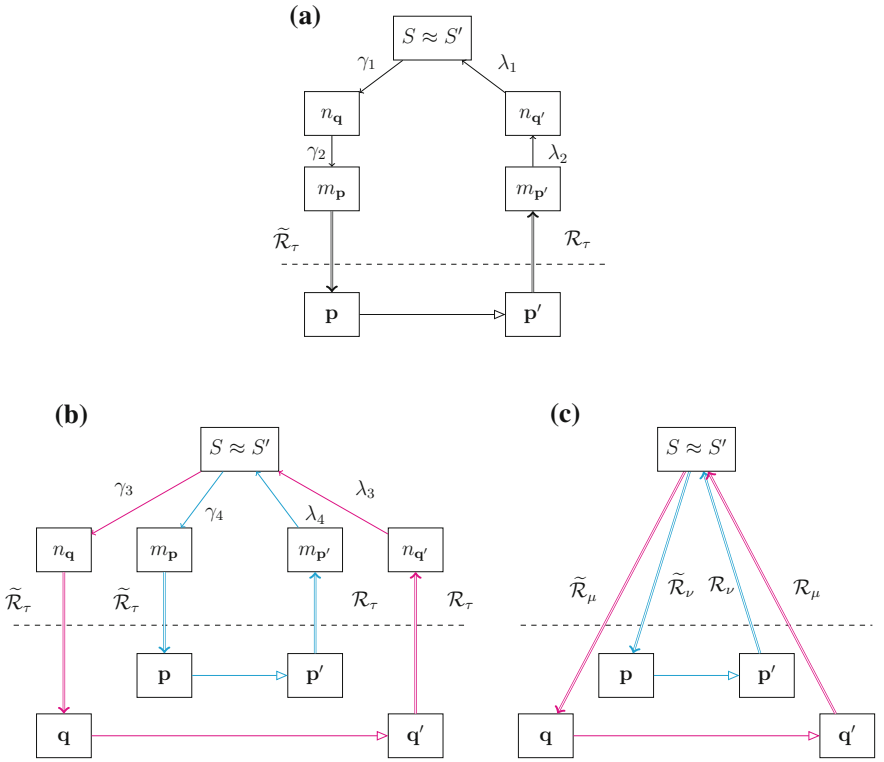


**Fig. 6** Multi-valued representation and instantiation of a signal $S$: **a** one signal carrier with two different abstract representations connected by a refinement; **b** two different refinements of the same signal instantiated in two different physical carriers (the different colours indicate the distinct signalling cycles); **c** (**b**) redrawn as two representation relations connecting the signal carriers

which in turn is an encoding of the signal $S$ through the function $\gamma_1$ (and similarly for the decoding stage). This is a single carrier being represented by different abstract models, both of which represent the same 'refined' signal $S$. Note that it may require considerable computation to implement these steps, including calculation and compilation (for encoding), and rendering (for decoding).

In comparison, Fig. 6b has the same signal, $S$, encoded in two different abstract models $n$ (here given a subscript to form $n_{\mathbf{q}}$), via refinement $\gamma_3$, and $m_{\mathbf{p}}$, via the refinement $\gamma_4$. Each of these in turn is instantiated in a separate physical carrier, **p** and **q**. This is a single signal being transmitted by two different carriers. They are equivalent if $\gamma_3 \equiv \gamma_1$ and there is a composition of the refinements such that $\gamma_4 \equiv \gamma_1 \circ \gamma_2$. It is worth noting that there are situations where this equivalence does not follow, most notably when considering the heterotic systems analysed in [11].

Figure 6b is also equivalent within AR theory to Fig. 6c, where there is only representation between the signal and the abstract layer. As noted in Sect. 2.5, encodings and representation relations can always be combined to form a new representation.

The converse is not, however, always necessarily possible. Some systems may make use of *primitive representation*: the signal is encoded directly into the physical system, rather than via some abstract representation of the physical system. The notion of primitive representation will become important when we consider intrinsic representation in systems.

What all these different situations demonstrate is that there is an arbitrariness to the encoding of signals (and by equivalent, if more complex, diagrams, of computing) in physical carriers. There is a strong sense in which arbitrariness, of both signal representations and carriers, is a *hallmark* of these forms of representational activity happening within physical systems. The multi-valued nature of abstract *versus* physical is well-known within Computer Science (where, for instance, the same computation may be performed on both a standard laptop and a computer constructed from beer cans and string); we see below that it also forms the key to determining the presence of computing, signalling and other representational activity in biological systems.

## 3 Intrinsic Representation

We have seen how AR theory locates computing in a physical system within the broader category of *representational activity*; including science, technology/engineering and signalling. We now turn to the question of when, and indeed if, a biological system can demonstrate such activity, how it can be recognised, and what considerations are needed to extend the framework that was developed in the context of human-centric representational activities to systems without such organisms present. This requires us to extend the range of key concepts within the framework. In this section we give the theoretical extensions and insights necessary for the identification of representational activity in biological systems, giving the framework within which discussion of specific examples then takes place in section Sect. 4.

### *3.1 Representational Entities*

The first consideration when analysing computing in biological systems is the ability of the system to be performing any representational activity at all. In AR theory, the ability to represent depends on there being a representational entity. Originally termed a 'computational entity' in [13], this is an entity capable of establishing a representation relation, and capable of encoding and decoding abstract information in physical systems. It physically locates the representation relation, and is the entity that performs abstraction, and uses the output of representational activity (including computation). If there is representational activity then there is always *something* that is performing it.

When considering standard computing, the representational entity is almost always a human being: the computer designer, or programmer, or user, given the representational cycle being determined. It locates and generates $\mathcal{R}_\mathcal{T}$, bridging the gap between abstract and physical. We now introduce some new terminology that is necessary for discussing computing outside this scenario. If a physical object $\mathbf{p}$ participates in a representational cycle (science, technology, computing, signalling) with a representation $\mathcal{R}_\mathcal{T}$ given by representational entity $\mathbf{e}$ (bold font as the representational entity must be physical), then we say that the system comprising $\{\mathbf{p}, \mathbf{e}, \mathcal{R}_\mathcal{T}\}$ forms a *closed representational system*. If the cycle is a compute cycle, then the set forms a *closed computational system* (and similarly for signalling, etc.). If the computational system $\{\mathbf{p}, \mathcal{R}_\mathcal{T}\}$ does *not* include the physical representational entity $\mathbf{e}$, we say that it is *open under representation*.

In standard computing scenarios, the computer (e.g. a laptop) does not form a closed representational system: the representational entity is separate from the computing device, and not even necessarily co-located with it. This is an example of *designed* computing. Biological systems can compute/signal in exactly this way when they are used by a human (or otherwise) entity to perform a computation. Examples include DNA computing [2] and the use of slime molds to compute shortest paths [1]. In these cases, $\mathbf{p}$ is the biological system, $\mathbf{e}$ the human experimenter/programmer, and $\mathcal{R}_\mathcal{T}$ the representation they have predetermined. This is another example of designed computing, fitting entirely within the field of non-standard or unconventional computing devices, and whose analysis will exactly mirror those given for, e.g. chemical and quantum computers [13].

The core of the present investigation, however, is whether, and if so *how*, a biological system can be said to be computing *intrinsically*. Is there a meaningful way within AR theory to describe a biological process in the absence of any human computer users or experimenters as computing, or indeed performing any other representational activity? AR theory gives us a straightforward way to phrase this question. We require a closed computational system to be present in order for computing to be occurring, $\{\mathbf{p}, \mathbf{e}, \mathcal{R}_\mathcal{T}\}$. In designed computing, $\mathbf{p}$ and $\mathbf{e}$ are separate physical objects; for example, a biological system is being used as a computer. If, however, the biological system itself is closed under representation then the system is performing *intrinsic* computation. In such a case, the physical system $\mathbf{p}$ (for example 'a leaf' or 'a bacterium') includes within it the representational entity, and the system $\{\mathbf{p}, \mathbf{e} \subseteq \mathbf{p}, \mathcal{R}_\mathcal{T}\}$ forms a closed computational system. The first step in identifying intrinsic computing is then to determine whether a biological system under consideration is acting as its own representational entity.

It is important to note immediately that AR theory itself gives no requirements as to the level of complexity that a representational entity must have; only that it is a physical and not an abstract object. Importantly, there is no requirement that representation activity needs to take place in the presence of conscious, sentient, or intelligent agents. This is crucial if we are to investigate the presence of computing (or other representational activity) in low-level biological systems. The presence of representational entities, and of representation itself, is the key element in AR theory's ability to give a meaningful answer to the question of when a biological

system is computing. Without an understanding of the crucial role of representation, answers given previously have been of only two types, neither particularly interesting: either that computation is an activity purely of conscious beings, and therefore any resemblance to biological processes is entirely misleading, or else that everything is computing at all times, not only human beings and bacteria, but rocks and subatomic particles. By focussing on the presence or otherwise of representation and representational entities, AR theory enables us to make important and meaningful distinctions between when a system is computing (or otherwise representing) and when it is merely 'going about its business'.

Non-intelligent representational entities do, however, pose a particular problem that rarely arises in human-computer interactions. When the representational activity is being performed by humans (engineering, technology, computing, etc.), the representational entity is generally both obvious and articulate. It is therefore usually straightforward to see that representation is happening (for example, a computer user can say that they are using a computer to perform a certain calculation), and to determine the representation relation (for example, the computer user tells us how they are encoding data on their laptop). In the absence of the ability to interrogate the representational entity, however, some other method is needed to determine whether representation is occurring within the processes under consideration and, if so, in what way it is being put to use. This requires careful analysis to avoid the most obvious pitfall: it is always possible for an external observer (e.g. us) to impose a *post hoc* abstract representation on the physical system that is not itself part of how the system is operating. We must avoid smuggling ourselves in as representational entities 'through the back door'; this results in a situation where everything, including rocks, compute. Computation then becomes a word without meaningful content. Instead, we must look at the representation happening within the system itself: do we as scientists and external observers represent that system itself as participating in representational activity intrinsic to itself, forming its own representational entity; and, if so, what is being represented?

## *3.2 Signatures of Representation*

The key to determining the presence of intrinsic representational activity is the understanding that when a system is 'going about its business' as normal, its dynamics are given by the physical system itself. The physical system **p** is the important element. If, however, it includes representation, then at some point the important element will be an abstract object $m_{\mathbf{p}}$. When considering intrinsic computation or signalling, abstract representation will always be an intermediary process: it always results in *physical* behavioural changes. What is seen from the outside is always some physical process: all data and information are embodied. In the absence of entities (such as conscious humans) that can use an abstract result (as in the human use of computers), the use of representation in biological systems will be to produce physical output. For example (see Sect. 4.1), consider the system comprising

a bacterium in a chemical gradient. The output of interest might be the observed part of this system that describes the bacterium's movement: swimming versus tumbling. As another example (see Sect. 4.2), consider a system comprising a cell containing DNA and other molecules. The output of interest might be the observed part of this system that is the protein expressed from a gene. The question is whether that output has been produced via the irreducible use of compute/signal cycles along the way.

Both representation (from physical to abstract) and instantiation (abstract to physical) are one:many maps. That is, a given physical system has many possible abstract representations (under different relations), and a given abstract object has many possible physical instantiations. If a physical object *qua* physical object is important in a biological process, then no other object will do: if **p** itself is needed in a process, then some other $\mathbf{p}_1$ will not do. However, if it is the *abstract* object and evolution that matters, then it is $m_{\mathbf{p}}$ that is needed; and there will in general be multiple ways in which this abstract object can be instantiated in a physical system.

It is precisely this multiplicity of mappings that is described in Sect. 2.7 in the context of computing and signalling. They are complex types of representational activity, but the arbitrariness of encoding and decoding identified there has its roots in the arbitrariness of representation, and the more broad type of one:many mapping that is present in all representational activity. In computing, it allows for the possibility of forming multiple alternative compute cycles. More generally, it gives many implementations of a single abstract process.

This is the key to analysing when a biological system is computing intrinsically: if a certain process can or could occur in multiple different ways, all of which instantiate a single abstract object or evolution, then this is the signature that it is the abstract and not the physical operation that is important. Can an arbitrariness of encoding and decoding within a given process be identified; and, from this, can it be seen that the given cycle could be implemented in multiple different physical systems with the same physical outcome at the end?

In order to make this precise, we here concentrate on the situation of intrinsic signalling within systems. As noted above, signalling can be viewed as a simplified form of computing; and within AR theory they share almost all their key elements. An ability to identify signalling and communication within biological systems is an important stepping-stone to an eventual understanding of when such systems compute. If signalling is present in a biological organism, then according to AR theory it has almost all the relevant components in order also to compute intrinsically.

Consider again the arbitrariness of encoding in signalling in designed representational systems, Fig. 6. This figure starts and ends in the abstract domain with the abstract signal *S*. By extending this to the intrinsic case, we give the equivalent signatures of signalling happening where the representational entity is not available for comment. In this situation, what is observed external to the system is an evolution of the whole system (e.g. a plant or a bacterium) from state **p** to $\mathbf{p}'$. By itself there is no way to determine if abstract objects or processes are necessary to this process. However, if there is a part of this evolution that is being implemented by the use of a signal, then there will be an identifiable use of representation within the system, as shown in Fig. 7. Let **q** be some physical subsystem of the overall physical system **p**;

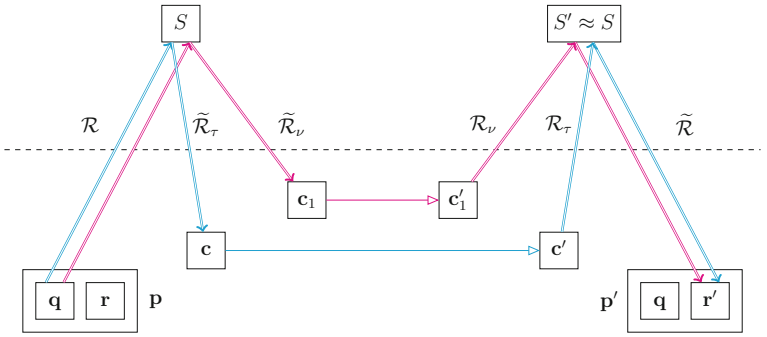


**Fig. 7** An intrinsic use of signalling, with two possible signal carriers $\mathbf{c}$ or $\mathbf{c}_1$ as an intermediary in the physical evolution $\mathbf{p} \to \mathbf{p}'$. The physical elements $\mathbf{q}$ and $\mathbf{r}$ are subsystems of the system $\mathbf{p}$. The different colours denote the two separate signalling pathways

this is the part of the system that is responsible for initiating the signal. The intrinsic use of signalling involves an initial representation $\mathcal{R}$ of $\mathbf{q} \subset \mathbf{p}$ as the abstract signal to be sent, $S$. This is an instance of what is noted in Sect. 2.7 as *primitive representation*: the organism itself has no 'abstract model' of itself, so there is no intermediate step between the physical system and the abstract signal that is being encoded.

In order for the signal $S$ to be transmitted, it is instantiated via $\widetilde{\mathcal{R}}_\tau$ in the physical signal carrier $\mathbf{c}$. This physical carrier evolves to $\mathbf{c}'$: standardly, it is transmitted with no change in its state other than time or space coordinates. $\mathbf{c}'$ is represented as signal $S'$ via $\mathcal{R}_\tau$. If the transmission is faithful, $S' \approx S$. Now consider $\mathbf{r}$, the part of the system $\mathbf{p}$ that is receiving the signal. By instantiating $S'$ in the new state $\mathbf{r}'$, the final state of the full system $\mathbf{p}'$ is produced.

Figure 7 also shows an alternative pathway for the signal; it is the identifiable possibility of these alternatives that gives us the signature of signalling happening in these systems. The signal carrier $\mathbf{c}$ is not the only possible transmission system: precisely as in Fig. 6, the signal $S$ could also be instantiated using $\widetilde{\mathcal{R}}_\nu$ into a different physical system: here, $\mathbf{c}_1$. By decoding at the other end using $\mathcal{R}_\nu$, the same change to subsystem $\mathbf{r}$ is instantiated, and the same physical evolution $\mathbf{p} \to \mathbf{p}'$ is effected. The system has evolved this way through the fundamental use of representation: the different ways in which signal has been transmitted have in common only that they are instantiations of a given abstract object $S$.

The key then is that the semantics of the physical process that uses signalling as an intermediary are given by the abstract signal $S$, not the specific physical carrier $\mathbf{c}$, which is just one of many that could have been used to perform the critical abstract operation. The presence of other signal carriers is a strong sign that representation is occurring fundamentally. This is not however an immediately sufficient criterion. In any given situation a close analysis of the actual signalling pathways will need to be done to back up this hypothesis. This requires a detailed understanding of the biology of the systems under consideration, and a close interrogation of the best scientific understanding of each individual system in order to identify all the separate levels of

encoding, decoding and representation that must be present for such representational activity. We turn to examples of such a detailed analysis in the next section.

This is then our challenge for identifying representation in biological systems: determining where there is an arbitrariness in the physical process (another type of physical system could perform the same operation), but where all the possible processes are physical instantiations of a single abstract object or process. In such a case, the physical objects and processes are functioning to instantiate abstract processes, and we argue that the biological system is engaging in signalling; a stepping-stone towards a further analysis of systems where, in certain cases, the biological system is *computing*.

## 4 Representation in Biological Systems

We now consider in detail three example biological systems. We will look specifically at them as candidates for the presence of the intrinsic use of signalling in biological organisms. This is a first stage towards, and proof-of-principle of, how to consider computation (the most complex of the representation activities considered here) in biological systems. As noted in the introduction, we focus on relatively low-level systems far removed from the degree of organisation and complexity needed for a neural/conscious organism. We consider specific processes within the candidate systems, and interrogate them for the presence of representation via the multiple realisability of information. We identify both representation and computation present even at these low levels and contrast it with an example where representational activity is not present.

### *4.1 Bacteria*

Our first example is of bacteria that possess a motor-driven propellor called a flagellum that enables them to move around in a watery environment. The biological detail in this section is taken from the review article [21].

Such bacteria will swim towards food using their flagellum. The control of the flagellar motor provides a clear example of computation by a biological system, in which signals are processed and integrated through a series of interactions between proteins. Receptor proteins that protrude through the cell membrane detect the level of nutrient outside the cell. This information is passed via the linker protein CheW to CheA. If the nutrient level is declining, CheA transfers a phosphate group to CheY, which moves through the cell and binds to a component of the flagellar motor. If enough CheY is bound, the motor will switch direction from anticlockwise, which drives the cell forward, to clockwise, which causes the cell to tumble, changing its orientation (See Fig. 8).

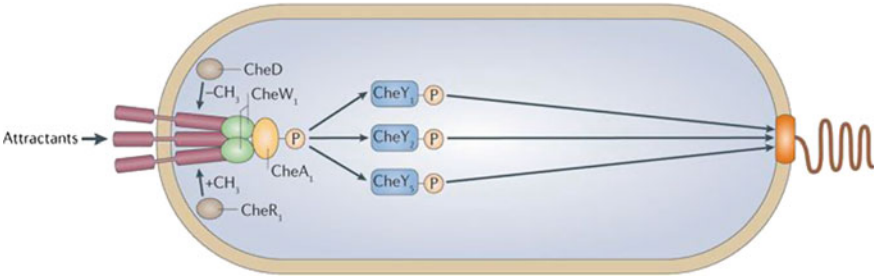


**Fig. 8** Some signalling pathways in *E. coli*. Taken from [21]

In order to determine whether nutrient levels are declining, the system requires a memory, so CheA also adds a phosphate to another protein CheB, which takes methyl groups off the receptor proteins, reducing their ability to activate CheA. This happens on a slower timescale, providing a 'memory' of the average of recent conditions. The net result of all this protein chemistry is that a bacterium that experiences steady or increasing nutrient concentrations keeps swimming in the same direction, but if it is heading into a region with less nutrient, it stops, spins round and tries a different direction at random.

The system just described has been investigated in detail in the well-known bacterium *Escherichia coli*, but many other bacteria have variants of the same system. In *Rhodobacter sphaeroides*, there is an additional set of internal receptors that monitor the level of nutrients inside the cell and modulate the system appropriately. In other words, before chasing food, it asks itself 'am I hungry?' and integrates that information with the external nutrient situation to determine its behaviour. Although these two bacteria use related systems to process the information, the actual nature of the nutrients detected is different: *E. coli* senses amino acids and sugars, whereas *R. sphaeroides* is attracted to acids such as acetate. Hence, the same internal protein-based process represents different external realities. The arbitrariness of the representation is further emphasised by a consideration of more distantly related bacteria such as *Bacillus subtilis*, in which the meaning of the pathway is inverted because it has a different 'memory' mechanism, so that the phosphate-bound form of CheY promotes smooth swimming, rather than tumbling. Furthermore, the same basic information processing system has been adapted to handle different kinds of input and output. Besides attractants and internal nutrient status, inputs can include repellants, oxygen and light, while the system can be used to control other complex behaviours such as developmental gene expression, cyst or biofilm formation.

This bacterial control system makes complex and integral use of representation through signalling, and even some signal processing. The receptor proteins serve as transducers that convert various external inputs into an arbitrary internal representation. This representation is then manipulated, integrating information (including a memory of past states) to generate an output signal that then leads to action by the flagellum or other transducers. Each bacterium has multiple receptors—sometimes

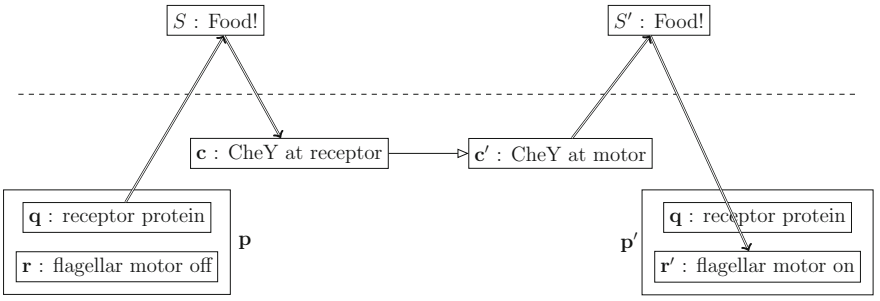


**Fig. 9** Signalling in the *E. Coli* bacterium

dozens—and may have several pathways operating in parallel with different inputs and possibly different outputs. By comparing different signalling pathways across different bacteria, as well as within a single instance, we can see the arbitrariness of the signal carrying substrate in action.

To see in detail how this system satisfies the AR category for signalling we consider again the specific example of *E. coli*, comparing the schematic diagram of its signalling pathways, Fig. 8, with that for intrinsic signalling in AR theory, Fig. 7. The bacterium itself is the full system **p**, and we isolate the three subsystems needed for an AR description of signalling. The receptor proteins form the subsystem initiated the signal, **q**. The CheY proteins are the signal carriers **c**, and finally the flagellar motor forms the subsystem receiving the signal, **r**.

The AR signalling schematic for the system is shown in Fig. 9. The part of the full bacterial system **p** that senses the external environment is **q**, the receptor. This is represented as a signal ("nutrient level rising", or, more simply, "food!"). The signal to be sent is instantiated as the CheY proteins, sent from the receptor to the flagellum. The protein at the flagellum represents "food!", which is then instantiated as the required action of another subsystem of the bacterium: the flagellar motor **r**.

We have drawn Fig. 9 using single instantiation and representation steps for simplicity (see Fig. 6b, c). However, in this system these steps are non-trivial. First, a single signal $S$, indicating rising nutrient levels, is instantiated into multiple carrier protein molecules: it is the number of CheY which determines the signal. Then representation back to the signal $S'$ combines the multiple molecules of the signal carriers into a single signal.

The multi-instantiation within this system is seen in the different ways in which the signal can be instantiated in different carriers. Different proteins could be used, as seen in different bacteria, and the essential role of the instantiation and representation steps make plain the central role of the CheY proteins as carrying the signal rather than anything else intrinsic to their physical state. The proteins themselves do not 'carry information' in isolation; it is their concentration that determines whether a signal is transmitted and received. Only in the context of the flagellar motor receptors does the number of CheY proteins form a representation of a signal. Without that representational context, the signal would not be transmitted. As it is, the bacterium

forms a closed representational system with the use of signalling integral to its final physical output: it is signalling intrinsically.

We have concentrated here on the signalling pathways; however, as noted above, there is also an amount of signal processing going on, in particular during the steps we have indicated as representation/instantiation (the signal involves multiple protein molecules). It is still to be determined (and in fact pre-dates AR theory as an open question in computer science) what minimal degree of signal processing is required before a system is performing intrinsic *computation*. This is an important open question for further research.

## *4.2 DNA*

We now turn to the area of biology where the language of information, encoding and decoding is arguably the most used: DNA systems as they store and transmit the data used for the construction of cellular life. There are a number of redundancies in the way information is coded in DNA, demonstrating a key instance of the use of intrinsic representation in biological systems.

DNA is a heteropolymer, a sequence comprised of four different nucleotide bases: adenine, cytosine, guanine and thymine. Triplets of bases (called codons) encode, or represent, particular amino acids, and sequences of triplets represent sequences of amino acids, or proteins. The genetic code is the particular mapping between base triplets and the corresponding amino acids; see Fig. 10.

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| UTT | Phe | TCT | Ser | TAT | Tyr | TGT | Cys |
| TTC | Phe | TCC | Ser | TAC | Tyr | TGC | Cys |
| TTA | Leu | TCA | Ser | TAA | Stop | TGA* | Stop |
| TTG | Leu | TCG | Ser | TAG | Stop | TGG | Trp |
| CTT | Leu | CCT | Pro | CAT | His | CGT | Arg |
| CTC | Leu | CCC | Pro | CAC | His | CGC | Arg |
| CTA | Leu | CCA | Pro | CAA | Gln | CGA | Arg |
| CTG | Leu | CCG | Pro | CAG | Gln | CGG | Arg |
| ATT | Ile | ACT | Thr | AAT | Asn | AGT | Ser |
| ATC | Ile | ACC | Thr | AAC | Asn | AGC | Ser |
| ATA* | Ile | ACA | Thr | AAA | Lys | AGA* | Arg |
| ATG | Met | ACG | Thr | AAG | Lys | AGG* | Arg |
| GTT | Val | GCT | Ala | GAT | Asp | GGT | Gly |
| GTC | Val | GCC | Ala | GAC | Asp | GGC | Gly |
| GTA | Val | GCA | Ala | GAA | Glu | GGA | Gly |
| GTG | Val | GCG | Ala | GAG | Glu | GGG | Gly |

**Fig. 10** The genetic code. How the 64 DNA codons map to 20 amino acids and a stop signal. This code is almost universal. But in vertebrate mitochondria, the starred codons code differently

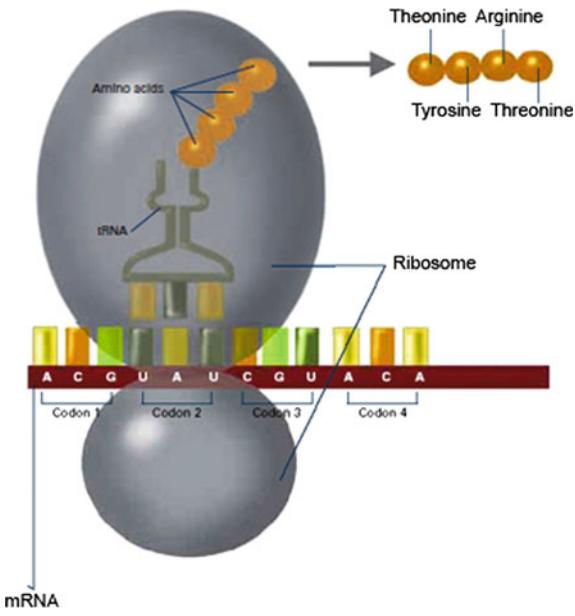


**Fig. 11** Production of amino acids through translation from mRNA codons via tRNA within a ribosome. Taken from [17]

The transfer of information from DNA to protein happens in two stages. First, a copy of the DNA is made in RNA, a molecule that is chemically different from DNA but has the same structure. This messenger RNA (mRNA) represents the DNA sequence base for base. It is the same text, just in a different font, so the process is 'transcription'. The translation from codons to amino acids is realised by a small molecular machine called a tRNA (transfer RNA). This has two 'ends', one of which binds to a codon of the mRNA, and the other to an amino acid, Fig. 11. The tRNAs are the dictionary entries that map the 'words' in the language of DNA into the protein language, so this is aptly called 'translation'. This translation exhibits the first of the levels of redundancy: in most cases, multiple codons are translated to a given amino acid.

As with the case of bacteria, we can analyse this system from within the framework of AR theory. As in the previous example, the output of the AR cycle is a physical system: the amino acid produced. This is comparable with the bacterial output, which was the specific motion of the flagellum in the direction of food. While this physical output is specific, much of the intermediate stage includes the arbitrariness that we see in the use of intrinsic representation. The system begins with the "physical encoding" step, where the structure of the codon is instantiated in the tRNA. This is similar to the first stage of signalling within a bacterium.

Figure 12 illustrates this situation in terms of signalling in AR theory. The signal $S$ is "make a given amino acid". A codon represents an amino acid; this representation is instantiated in a tRNA. Different codons, with their associated tRNAs can represent and instantiate the *same* amino acid. In this system, there are multiple pairs of

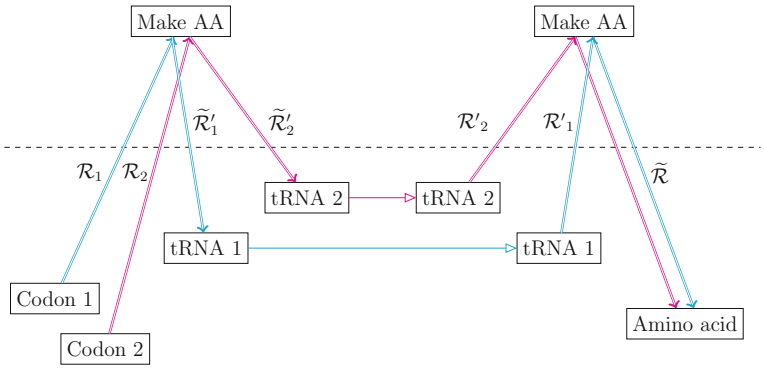


**Fig. 12** Signalling in the genetic code

subsystem **q** (codon) and **c** (tRNA), but only a single signal $S$ and the physical output result $\mathbf{r}'$ (amino acid).

There are further places in the system where multiple representations and instantiations can occur. For example, if the tRNAs were different, then the genetic code would be different. In nature, the genetic code is almost universal across organisms, but there are some naturally occurring minor differences in the genetic code. In vertebrate mitochondria, TGA is not a stop codon, but codes for tryptophan (Trp); ATA does not code for isoleucine (Ile), but for methionine (Met); AGA and AGG become stop codons [19]. That is, these codons represent different information in different contexts. Other organisms have reassigned codons in many other ways. Similar styles of changes have also been engineered within synthetic biology into the genetic code, for example in order to add an extra amino acid [25]. One could also, in principle, have a very different code, by altering all the tRNAs, and then altering the DNA so that these new tRNAs still produce the original proteins.

Even without changing the genetic code itself, there is a certain arbitrariness to the representation. For example, the DNA has to be read three bases at a time to form codons, but where to start? This is defined by the *reading frame*. Some DNA in viruses and mitochondria supports multiple reading frames [6]. So the same piece of DNA can represent different proteins, corresponding to different reading frames.

The underlying nucleotide bases structure of DNA has been extended with further 'unnatural' base pairs, extending the A-T C-G pairs [27]. Such extended DNA has been included in a plasmid, inserted into *E. coli*, and successfully replicated [15]. Potentially, new nucleotides can expand the codon dictionary and be translated into novel amino acids [4].

### *4.3 Photosynthesis*

As a final example, we consider the energy transport processes in photosynthesis. It is a complex process involving specialised cell structures in a cascading sequence of excitations until the electromagnetic energy of the incoming light has been converted into chemical energy in molecules in the cells. This is accomplished by using an incoming photon (a quantum unit of light) to create an exciton (a quantum quasiparticle) in the light-harvesting molecules. Roughly speaking, an exciton is what you get when several electrons are excited into higher energy levels in a collective way. This then needs to be transported to the molecular reaction centre, where the chemical reactions can take place to store the energy. During transport, the exciton moves through the molecules, losing a fraction of its energy to vibrations in the molecules. This energy loss allows the dynamic process of energy transfer to proceed. A lower energy exciton will fit into neighbouring electron excitations better than the current ones, so the excitations transfer. This clearly requires exquisitely tuned molecular structures, and these have been intensively studied to understand the process in detail [7, 9].

Photosynthesis has caught the attention of the quantum information community, who have contributed insights to our understanding of how exactly this process works so well [16, 20]. As anyone who has ever suffered sunburn will know, photons from the sun can be dangerous. They are capable of breaking chemical bonds and thus damaging essential cellular machinery. It is crucial that the captured incoming photon is safely transferred to the reaction centre without causing unwanted rearrangement of the cell's molecules; energy is conserved, so it has to go somewhere once inside the cell. Achieving this high transfer efficiency is best explained using quantum mechanics to describe how the collective excitations maintain *coherence* to transfer the energy step by step. These processes are well-studied in the quantum information context, enabling fragile quantum information to be transported and stored, or processed in quantum computers.

Within the context of photosynthesis, however, these collective excitations are not being *used* by a leaf (or other photosynthesising system) to store or process quantum information. There has been an element of confusion present in the quantum information community since the discovery that this transmission can be modelled as a quantum walk [16], which is a process that can be used in quantum computing, and which can find the shortest path between two points faster than a classical random walk. Problems arise when these results are mistakenly seen to demonstrate that an exciton is 'quantum-computing' the shortest path to the reaction centre.

We can see within AR theory why it is not the case that a light-harvesting complex is computing, or indeed engaged in any other representational activity. The aim of photosynthesis is the transmission of the exciton: the semantics of this process can be understood entirely 'below the line' as a physical process transmitting energy, without needing to describe the system as intrinsically representing anything. There are multiple different ways that this process could be implemented, with different energy carriers, but they all involve the carrying of the energy in the exciton, not storing or

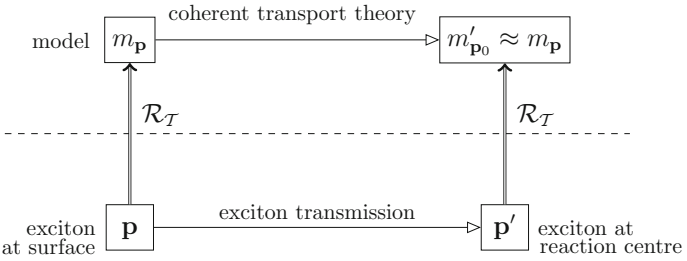


**Fig. 13** A commuting 'science' diagram (c.f. Fig. 2d). Note we have a representation relation at both the beginning and end of the physical evolution. There is no instantiation relation, and hence no computing occurring here

processing any information. There is no multiplicity of instantiation of an abstract object (a signal or other information) created intrinsically by the light-harvesting complex itself. We can draw a 'science' diagram of our external representation of the process (Fig. 13), but not a signalling diagram involving intrinsic representation and instantiation of a signal.

As always, the key is to look to our best theories of the process under consideration: do they represent the system as itself representing information, or is that a description that can be given only *post hoc*, in the presence of external representational entities? In photosynthesis, a full description of the process of energy transmission requires only the physical elements of the exciton. If there is some information being computed that is essential for the process (such as a shortest path), then we would expect multiple ways this could be instantiated for a given exciton. However that is not the case: there is no mechanism for a given exciton to get the 'information' about the shortest path to the reaction centre by using a different physical mechanism. The specific exciton being transported has to undergo quantum coherent transmission, and nothing else. There is no signature of representational activity present intrinsically.

Photosynthesis is an example of the problems that arise with a failure to distinguish between intrinsic and imposed representational activity in natural systems. A light-harvesting complex plus photon/exciton considered in terms of computing a shortest path is not a closed representational system: $\mathcal{R}_{\mathcal{T}}$ for this system would need to be set up by external representational entities (e.g. physicists or biologists looking at the system in this way). There is no evidence of the system itself using this representation of the process in order to complete the energy transport. In fact, even with an external entity imposing a representation in terms of information processing, a single instance of photosynthesis is still not computing: the representation is imposed *post hoc* as a parallel description of the physical process, but the physical system is not being used to predict the outcome of any abstract evolution. In the absence of a prediction element, the system does not satisfy the conditions for computation. Only if a photosynthesising organism were engineered into a larger computing system, and the path of the exciton used to encode some other abstract object, would the process

of photosynthesis be computing. As it stands, it is a purely physical process, with no evidence for any intrinsic representational activity.

## 5 Summary and Conclusions

We have described in some detail our abstraction-representation framework in which representational activity such as signalling or computation can be identified in simple living systems that are far below the sophistication of nervous systems or consciousness. Crucial to identifying representation is the arbitrariness of the systems used to represent the information concerned, information such as 'hungry' or 'food molecules detected'. However, since in many cases the same molecules can be processed both as a source of energy (food) and as triggers for behaviour modification (swim towards food), identifying such systems as representational requires a detailed analysis of the processes. Often this means studying a whole class of organisms to draw out common features that are united by the representational activity. That life uses arbitrary representations at a fundamental level is most clearly illustrated by the variations in the encoding of amino acids by DNA. Conversely, there are many systems that are dedicated to processing energy, to power the cellular machinery, and these do not *a priori* need to include any representational activity. There is clearly much further work to be done to elucidate the conditions that identify representation in biological systems, including determination of the simplest possible representational system, and the ubiquity of representation in living organisms. Our framework is designed to bring clarity to this endeavour, by clearly defining what it means to be representational and how to identify when it is, and is not, occurring.

## References

1. Adamatzky, A.: Physarum Machines: Computers from Slime Mould. World Scientific (2010)
2. Amos, M.: Theoretical and Experimental DNA Computation. Springer (2005)
3. Brent, R., Bruck, J.: 2020 computing: can computers help to explain biology? Nature **440**, 416–417 (2006)
4. Bain, J.D., Switzer, C., Chamberlin, A.R., Benner, S.A.: Ribosome-mediated incorporation of a non-standard amino acid into a peptide through expansion of the genetic code. Nature **356**(6369), 537–539 (1992)
5. Cousot, P., Cousot, R.: Abstract interpretation: a unified lattice model for static analysis of programs by construction or approximation of fixpoints. In: Proceedings of the 4th ACM SIGACT-SIGPLAN Symposium on Principles of Programming Languages, pp. 238–252. ACM (1977)
6. Chirico, N., Vianelli, A., Belshaw, R.: Why genes overlap in viruses. Proc. R. Soc. B: Biol. Sci. **277**(1701), 17–3809 (2010)
7. Cho, M., Vaswani, H.M., Stenger, J., Brixner, T., Fleming, G.R.: Exciton analysis in 2D electronic spectroscopy. J. Phys. Chem. B **109**(21), 10542–10556 (2005)
8. De Marse, T.B., Dockendorf, K.P.: Adaptive flight control with living neuronal networks on microelectrode arrays. In: Proceedings of the 2005 IEEE International Joint Conference on Neural Networks. IJCNN'05, vol. 3, pp. 1548–1551. IEEE (2005)

9. Engel, G.S., Calhoun, T.R., Read, E.L., Ahn, T.-K., Cheng, Y.-C., Mancal, T., Blankenship, R.E., Fleming, G.R.: Evidence for wavelike energy transfer through quantum coherence in photosynthetic systems. Nature **446**, 782–786 (2007)
10. He, J., Hoare, C.A.R., Sanders, J.W.: Data refinement refined resume. In: ESOP 86, pp. 187–196. Springer (1986)
11. Horsman, D.C.: Abstraction/representation theory for heterotic physical computing. Philos. Trans. R. Soc. A **373**, 20140224 (2015)
12. Horsman, C., Stepney, S., Kendon, V.: When does an unconventional substrate compute? In: UCNC 2014 Poster Proceedings, University of Western Ontario Technical report #758 (2014)
13. Horsman, C., Stepney, S., Wagner, R.C., Kendon, V.: When does a physical system compute? Proce. R. Soc. A **470**(2169), 20140182 (2014)
14. Kendon, V., Sebald, A., Stepney, S.: Heterotic computing: past, present and future. Philos. Trans. R. Soci. A **373**, 20140225 (2015)
15. Malyshev, D.A., Dhami, K., Lavergne, T., Chen, T., Dai, N., Foster, J.M., Correa, I.R., Romesberg, F.E.: A semi-synthetic organism with an expanded genetic alphabet. Nature **509**(7500), 385–388 (2014)
16. Mohseni, M., Rebentrost, P., Lloyd, S., Aspuru-Guzik, A.: Environment-assisted quantum walks in photosynthetic energy transfer. J. Chem. Phys. **129**, 174106 (2008)
17. National Institute of General Medical Sciences. The New Genetics. NIH Publication No.10-662 (2010). http://www.nigms.nih.gov
18. Navlakha, S., Bar-Joseph, Z.: Distributed information processing in biological and computational systems. Commun. ACM **58**(1), 94–102 (2015)
19. Osawa, S., Jukes, T.H., Watanabe, K., Muto, A.: Recent evidence for evolution of the genetic code. Microbiol. Rev. **56**(1), 229–264 (1992)
20. Plenio, M.B., Huelga, S.F.: Dephasing assisted transport: quantum networks and biomolecules. New J. Phys. **10**, 113019 (2008)
21. Porter, S.L., Wadhams, G.H., Armitage, J.P.: Signal processing in complex chemotaxis pathways. Nat. Rev. Microbiol. **9**(3), 153–165 (2011)
22. Regev, A., Shapiro, E.: Cellular abstractions: cells as computation. Nature **419**, 343 (2002)
23. Wooley, J., Lin, H. (eds.): Catalyzing Inquiry at the Interface of Computing and Biology. National Academies Press (2005)
24. War, A.R., Paulraj, M.G., Ahmad, T., Buhroo, A.A., Hussain, B., Ignacimuthu, S., Sharma, H.C.: Mechanisms of plant defense against insect herbivores. Plant Signal. Behav. **7**(10), 1306–1320 (2012)
25. Wang, Q., Parrish, A.R., Wang, L.: Expanding the genetic code for biological studies. Chem. Biol. **16**(3), 323–336 (2009)
26. Wang, B., Kitney, R, Joly, N., Buck, M.: Engineering modular orthogonal genetic logic gates for robust digital-like synthetic biology. Nat. Commun. **2**(508) (2011)
27. Yang, Z., Hutter, D., Sheng, P., Sismour, A.M., Benner, S.A.: Artificially expanded genetic information system: a new base pair with an alternative hydrogen bonding pattern. Nucleic Acids Res. **34**(21), 6095–6101 (2006)