What kinds of natural processes can be regarded as computations?

Colin G. Johnson.
Computing Laboratory.
University of Kent at Canterbury.
Canterbury, Kent, CT2 7NF, England.
Email: C.G.Johnson@ukc.ac.uk

Abstract

This chapter is concerned with how computational ideas can be used as the basis for understanding biological systems, not by simulating such systems, but by taking a computational stance towards the way such systems work. A number of issues are addressed. Firstly the question of what kinds of computer science are needed to help understand computational processes which happen outside of conventional computing machines. The second issue addressed places computational constraints on how the world can act into Dennett's framework of grades of possibility. The final main section considers the issue of changes in the world, and when it is meaningful to regard such changes as carrying out computations.

1. Introduction

In recent years the idea of using *computational concepts* as a way of understanding biological systems has become of increasing importance; this conceptual use of computational ideas should be contrasted with the equally valuable activity of using computers as tools for interpreting biological data and simulating biological systems. This computational attitude towards biological systems has been valuable in computer science itself, too; by observing how biological systems solve problems, new algorithms for problem-solving on computers can be developed.

The aim of this chapter is to tease out some details of how ideas from computing can be used to inform thinking about biological questions, and *vice versa*. In keeping with the theme of the book an attempt is made to use ideas from cellular and tissue-level biology. The following questions indicate the main issues addressed:

- What kind of computer science is needed to answer biological questions?
- What does computational complexity mean when computing is grounded in the physical world?
- Does computation place limits on what sort of thing is possible in the world, and how does this fit in with other ways of assessing possibility?
- What does the ability of computers to simulate or not be able to simulate a system say about those systems?
- What kinds of transformations in the world can be regarded as being computations; and which transformations can be thought of as *not* being computations?

2. *Computer* science or computer *science*?

"Who could believe an ant in theory?
a giraffe in blueprint?
Ten thousand doctors of what's possible
could reason half the jungle out of being."
—John Ciardi [Cia97]

It is a tired cliché of popular psychology that we only use 10% of our brains. It is unlikely that this is true, but it is interesting to consider how a statement like this might be interpreted. Is this a *physical* statement? Could we cut out the 90% that isn't being used, throw it away, and still function normally? Is this a biological question, meaning that we are only using 10% of the available neuronal pathways or only activating 10% of the signals that we could? This is perhaps closer, but still not ideal. Does it mean that we could store 10 times as much "stuff" if we were working at full capacity? Think 10 times as quickly? These are still fairly ill-defined questions, but they conform to the intuitions which people have about the brain, and they are at heart *computational* questions. They are questions about the capacity of an entity for the storage of information and its ability to process that information.

The point of this story is to illustrate that we already think about biological processes in terms of computational ideas, even if in an informal way. It is not surprising to find that we think about the brain in this way, given both the popular view of brains as being essentially computers and the view dating back to the early days of computing of computers as "electronic brains". However it is only a short step from this to start thinking about other parts of the body (whether at the tissue level or the cellular level) in computational terms.

Many cellular systems have an *information processing* aspect. The immune system is well studied from this perspective [FH00,PW97], and there is potential to view the signal transduction system in this way. What kinds of computer science are needed to help understand these kinds of system?

2.1 Complexity of natural computation.

One example of a piece of computer science theory that could provide a tool for the understanding of natural computation is a theory of complexity. If we consider computation to be something that is grounded in the world, then how does that influence our view of computational complexity? What kinds of complexity in nature arise out of the presence of computations in natural systems? Clearly we can always define certain formal systems as being what we mean by the term computation, and then derive/define certain measures of complexity with respect to this definition. However if we want to apply computational ideas in the context of analysing transformations of the world then we might want to not ground these in particular axiomatisations of computation, as we might not be able to show that the physical system conforms to that axiomatisation.

An interesting example of this is protein-folding, where a linear string of proteins can form a three-dimensional structure in a very short amount of time [Fra93]. The number of possible configurations that this three-dimensional structure could take, and the problem of calculating this structure is computationally hard (e.g. it has been shown to be NP-complete [BL98,CGP+98]). How, therefore, does the protein "compute" its configuration on a realistic timescale? This is a well known problem in theoretical biology, known in less formal terms as *Levinthal's paradox* [Lev69]. It may be the case that there is little to explain here; whilst the process happens quickly, it may be that the process is simply doing conventional computation very quickly, and if we were to be able to measure the timescales on which the folding was happening with accuracy there is sufficient time to do enough operations. However if it is not, we are presented with the interesting question of how the system processes the information sufficiently quickly to produce the result. Is it exploiting some property of the world which we do not use in building conventional computers, and which we do not therefore incorporate into our conventional models of computing? Is it exploiting computation in some way which means that we cannot on conventional computers (e.g. using some form of high-density parallelism in which small parts of the system can be considered as doing local computations from which the global structure emerges)? Or are we wrong to assert in the first place that if something changes in a way such that we can measure its computational complexity then it is necessarily doing the problem in a computational way. A similar problem occurs with mathematical models. We can demonstrate that a certain complex set of differential equations models the turbulent motion of a seed blowing around in the wind. However a bird moving to catch such a seed doesn't need to solve those equations in order to catch the seed, nor does the seed need to be aware of the equations in order to carry out the movement. Perhaps the problem is simply a confusion between a description of the system and the system itself.

2.2 Simulation of natural systems

An interesting perspective on the relationship between natural systems and computational systems is considering the idea of simulating the system in question on a computer. One of the most interesting results to come out of such a thought is the original work on quantum computing by Feynman. His original idea about quantum computing came from considering the idea of simulating quantum physics on computers [Fey82]:

> *"[...] the full description of the quantum mechanics for a large system [...], because it has too many variables,* cannot be simulated *with a normal computer [...]. And therefore, the problem is, how can we simulate the quantum mechanics? There are two ways we can go about it. We can give up on our rule about what the computer was, we can say: Let the computer itself be built out of quantum mechanical elements which obey quantum mechanical laws. [...]"*

We can generalize this idea to all natural systems as follows. Given any system in the world and some idea of what we mean by a computer either we can simulate it on the computer or not. There are two variants of this. In the first we consider what can be simulated at all. For example we cannot accurately simulate most non-discrete systems using a computer with a finite memory (we can clearly simulate *some* such systems, such as relationships between two intervals on the real line which can be described by a finitely-describable function). This is regardless of the amount of time we take; even given any finite number of timesteps we cannot even represent the initial state of the system exactly.

Other systems admit an inefficient simulation. For example a problem like factoring composite integers is hard (in the technical sense) on conventional computers, yet proposed quantum computers [WC98] provide a technology on which polynomial-time algorithms for the same problem can be executed.

The consequence of this is that if we cannot simulate the system (efficiently, or at all) on the computer then theoretically there is a property of the world which can be used as a substrate for computation. Clearly whether the particular property admits its use for computing artificially created problems will vary from case to case. In particular a significant feature is whether the system admits control over its inputs; many computations are happening in the natural world which cannot take inputs other than the ones which they receive as part of a larger system. Therefore we cannot say that merely observing the act of computation in a natural system provides the core of a practical computation system.

3. Grades of possibility

"There seem to be at least four different kinds or grades of possibility: logical, physical, biological, and historical, nested in that order." Daniel Dennett [Den95]

Does computation have a role to play in explaining what is possible in the world? It has been suggested by Dennett [Den95] that there is a *hierarchy* of "grades of possibility" of things in the world. He suggests the following as such a hierarchy (with the possibility of other grades being added):

- Logical
- Physical
- Biological
- Historical

In order for something (a biological something, that is) to actually exist in the world, it has to be possible at all levels. However given a putative object which does *not* exist in the world, that non-existence can be explained at one of the levels. Some things are not logically possible, for example an object which simultaneously exists and doesn't. In order to explain the impossibility of such an object a logical explanation suffices; it is not necessary to go further in the hierarchy to explain the impossibility of such an object. Just because a putative object contains characteristics which associate it with a point on the hierarchy doesn't necessarily place it there; 10 metre high ants would be biological objects, but it is not necessary to go as far as biological in the hierarchy to explain their absence in the world; physics will do that for us. Therefore for every putative object each of those stages can be examined and whether it is possible at that level or not determined. Broadly any object placed at one level must also be possible in the previous level, though there are complexities, in particular the "historical" level can contain objects which are physically possible but which have no biological aspect, so it is impossible to place them meaningfully in or out of the biological category. There are a number of ways to deal with this consistently, e.g. allowing non-biological objects through the biological layer without being classified or branching the hierarchy into "biological" and "non-biological" branches. The final stage in the hierarchy is concerned with what has actually happened; thus it is necessary to fix a particular point in time before it is possible to make statements about that final point in the hierarchy.

It is an interesting thought-experiment to take each of the grades in the hierarchy and think of some putative object which is impossible because of precisely that reason, i.e. the reason "earlier" in the hierarchy admits it, whilst the current reason is sufficient to dismiss it without needing to go further into the list.

3.1 Computational possibility.

Might it be reasonable to introduce a new grade into this hierarchy, *computational* possibility? Such a grade would include items which are possible because they require computation to be carried out in order for them to exist, and they require that that computation would be feasible given the computational resources available in the system. Where would such a grade of possibility go in the hierarchy? To avoid the complications about non-biological objects discussed above, let us restrict ourselves to biological systems only. Firstly let us try to introduce a computational grade between “biological” and “historical”. What might be an example of something which is biologically plausible but computationally impossible (or effectively impossible). One example might be a type of asexually reproducing bacteria (or any other kind of asexually reproducing creature) which are genetically identical from generation to generation. The reason this is implausible is because of the imperfection of the information-transmission process from generation to generation; as mutations occur and get passed onto future generations, so the initial genomic uniformity gets broken down. Clearly this could be regarded as a biological property; but by the same token all the biological properties could be regarded as physical in origin. What we are trying to do is to *refine* the space between the two extremes of the hierarchy.

Where else might “computational” be placed in the hierarchy? If computational is placed between “physical” and “biological” we are concerned with computational systems which can be realized in the physical world yet which cannot be implemented by biological systems. It is hard to think of a nontrivial example. It might be reasonable to assert that biology places constraints on the size of creatures, and therefore on the amount of information which they can store; therefore some computer systems could be physically created which wouldn’t be capable of biological realization. This seems to be an unsophisticated example, however. As we discover more about the mechanisms by which biological systems compute, we may find more things in this category.

Another possibility would be to place “computational” between “logical” and “physical”. This would suggest that there are computational constraints on the laws of physics; such an idea has been occasionally explored by theoretical physicists [Sch97,Sch00]. Exploration of this idea would take us too far from our main discussion here.

It may be that "computational" can be meaningfully placed at a number of points in the hierarchy, and that these different placements give a taxonomy of different kinds of computational phenomena.

4. Can a change *not* be a computation?

"Does a rock compute every finite-state automaton?" David Chalmers [Cha96]

In the above discussion we have been considering the consequences of considering certain actions in the natural world to be computations. In this section the question is reversed. Consider the following question: are there any transformations in the natural world which we can *not* meaningfully regard as being computations?

By considering some action in the world to be a computation, a number of questions about that action and the system in which it occurs: how is information stored in the system? What is the scope of transformations which can be made to that information? How does the complexity of doing that transformation place constraints on what the system can do on a particular timescale?

If we want to stop thinking of computing as just something which happens in machines inside beige boxes containing electrical circuits and consider it to be a property of natural systems such as cellular systems then we need to decide where to stop. There would seem to be a danger in using the term "computation" excessively to the point where it just becomes synonymous with "change" or "transformation". Given the set of all possible transformations which can happen in the world (or the particular part of the world which we are interested in, e.g. the cellular world), to which of them do we want to ascribe the label "computation". On a trivial level we are free to use this work in any way we want, so perhaps we should refine the question somewhat. A better version might be this: given the set of transformations which can happen in the world, how can they be divided into "computations" and "non-computations" in a way which respects the essential properties of computation in machines. The difficulty here is with the word "essential"; given that we are attempting to extend a concept away from the domain in which it was originally defined, we must let go of some ideas, otherwise there would be no problem. In the rest of this section I would like to consider a number of features which might help to make a useful distinction.

4.1 Observability

Is the notion of a change in the world being *observed* essential to the idea of including it in the set of computations? This idea can be unpacked in two directions.

Firstly it doesn't seem essential that the computation itself be observable, only that the inputs and outputs be. In normal computing, we are happy with the idea that a user of a system (whether that user is a human or another computer system making an automated enquiry) interacts with the system by specifying input and in turn receives output; they do not need to see the states which the machine takes on in between. Indeed it seems natural to extend the idea of computation to those systems where the changing state *cannot* be observed without disturbing the process, as in quantum computing.

Secondly we can concentrate on the question of what is doing the observing. It does not seem necessary to restrict the observer to being a conscious entity; it would seem reasonable to suggest that in a multi-component system, one component can carry out a computation and pass its output onto another. It may be the case that a system can be self-observing.

The aim of considering observability is to attempt to exclude those parts of the world which are changing but not affecting other parts; however this doesn't seem to be a significant part of "computing". Whilst transformations happen in the world without being observed (in the broad sense of passing their data onto another system), it does not seem that we should exclude these from what we regard as computations, or that this is a significant distinction (it is akin to "when a tree falls in the woods, does it make a sound?"—entirely dependent on definition).

4.2 Consistent ascribing of symbols

An important characteristic of computing is that symbols within the system have a consistent interpretation throughout the computation, or at least if they do not there is a component of the system which explains how the interpretation of the symbols changes as the computation progresses. That is, any external system which observes and/or initiates a computation must declare in advance how it is going to interpret those symbols. This seems to be a key characteristic of computing which can be applied to natural systems.

If there is not a consistent allocation of symbols then transformations are meaningless. In particular if we are completely free to assign any symbol to any meaning at any point in the computation then we can say that *any* transformation is doing any computing (subject to certain restrictions on the number of bits being transformed). This is akin to the "can a rock implement every finite-state automaton" argument [Put88,Cha96]. If we take a trivial "transformation" of a system (i.e. one in which nothing changes as a result of the transformation) and we are free to change the interpretation of the symbols, then we can just "relabel" the unchanged symbols in terms of the desired output; we would presumably not want to ascribe the property of computation to that trivial non-transformation.

It seems that many biological systems to which we want to ascribe the idea of computation support this idea. The output from a computation on a traditional computer passes a stream of bits to a screen or printer which are interpreted in a consistent way so as to display a particular text or image. In biological cells the end result of a sequence of signal transduction steps on receipt of a particular receptor is a particular protein; in protein folding a particular amino acid sequence gives rise to a particular three-dimensional structure (or one drawn from a particular probability distribution of structures). It is important to make a distinction between *consistent* and *deterministic* here; this property does not exclude probabilistic actions being included in computations.

4.3 Digital encoding

Is a digital encoding of information necessary in order to call some transformation a computation? Many discussions of computing assume that digital information is at the heart of computing, and the fact that the genetic system is digital is often seen as one of the core arguments for evolution and development having computational aspects to them. It is possible, however, to construct computational devices out of non-digital components, and to construct algorithms which make use of analogue representations of information; indeed in the early development of computing digital and analogue approaches to computation were developed alongside each other. If we are to think of computing as something which occurs in a wider variety of systems, it would seem that we shouldn't take the presence of a digital representation of information as a key factor in deciding whether a system is computational or not. Indeed in many cellular systems the structures for representing information digitally do not seem to exist.

4.4 Flexibility of inputs

Another factor which we may want to take into account in developing a distinction between computation and non-computation is the flexibility that an external system has to change the input. An important characteristic of computing is that computers act on different data; they don't just do the same action all the time. Still important, though perhaps less core to the idea of computing, is the idea of programmability. The ability to influence the system by adding new information would seem to be a core idea in ascribing an idea of "computing" to an action in the world. Again this is well illustrated by the protein folding problem; one of the reasons that we can easily apply computational reasoning to understanding that problem is that we can put information into the system as symbol strings, and the range of inputs is vast.

4.5 Intention to initiate a change

A final property we shall consider here is whether the *intention* to do a computation is a significant factor in deciding which natural transformations should be regarded as computations. As with the idea of observability above, intention here need not mean conscious intention. However it seems important when ascribing the notion of computation to an action that it be triggered by some other system (or by itself, but in a more sophisticated way than just existing in a changing state) with the end result that the output from the system will act in the world. By making this distinction we draw a line between those transformations which are part of some system which is acting to effect a deliberate change on the world, and those which are just happening because the laws of physics are acting on certain pieces of matter.

4.6 Summary.

Clearly we could consider other properties. However it seems that we are beginning to tease out what we might mean by a "computation", and what transformations we might ascribe to the category "not a computation". Clearly this is a topic around which much future discussion could revolve.

In particular it is interesting to speculate as to whether it is only a historical happenstance that our first encounter with the concept of computation was through the synthetic creation of computing devices? Could we instead have come across some of the important ideas in computing by an analytic study of biological systems? If so, which concepts would have most easily been discovered through such an analytic study? What systems, and what commonalities between systems, might have suggested these concepts? Are there other concepts, now regarded principally as analytic scientific concepts, which were originally discovered as synthetic engineering ideas? If so, how did the transition occur from the concept being seen as purely synthetic to it being seen as a scientific concept which could be applied in an analytic fashion to understand objects in the natural world? If we believe that this is important, how do we go about encouraging such an enrichment of ideas in the context of concepts from computing?

How can we revisit computational ideas without being overly distracted by the kind of computation that we see on computational devices? What would computer scientists be studying if the computer had not been invented?

References

[BL98] B. Berger and T. Leighton. Protein folding in the hydrophilic-hydrophobic (HP) model is NP-complete. *Journal of Computational Biology*, 5(1): 27–40, 1998.

[CGP+98] Pierluigi Crescenzi, Deborah Goldman, Christos Papadimitriou, Antonio Piccolboni, and Mihalis Yannakakis. On the complexity of protein folding. *Journal of Computational Biology*, 5: 423–465, 1998.

[Cha96] David J. Chalmers. Does a rock implement every finite-state automaton? *Synthese*, 108: 309–333, 1996.

[Cia97] John Ciardi. *The Collected Poems of John Ciardi*. University of Arkansas Press, 1997. edited by Edward M. Cifelli.

[Den95] Daniel Dennett. *Darwin's Dangerous Idea: Evolution and the Meanings of Life*. Penguin, 1995.

[Fey82] Richard P. Feynman. Simulating physics with computers. *International Journal of Theoretical Physics*, 21: 467–488, 1982.

[FH00] S. Forrest and S.A. Hofmeyr. Immunology as information processing. In L.A. Segel and I. Cohen, editors, *Design Principles for the Immune System and Other Distributed Autonomous Systems*. Oxford University Press, 2000.

[Fra93] A. Fraenkel. Complexity of protein folding. *Bulletin of Mathematical Biology*, 55(6): 1199–1210, 1993.

[Lev69] Cyrus Levinthal. How to fold graciously. In J. T. P. DeBrunner and E. Munck, editors, *Mossbauer Spectroscopy in Biological Systems: Proceedings of a meeting held at Allerton House, Monticello, Illinois*, pages 22–24. University of Illinois Press, 1969.

[Put88] Hilary Putnam. *Representation and Reality*. MIT Press, 1988.

[PW97] Alan S. Perelson and Gerard Weisbuch. Immunology for physicists. *Reviews of Modern Physics*, 69(4): 1219–1267, 1997.

[Sch97] Jürgen Schmidhuber. A computer scientist's view of life, the universe, and everything. In C. Freksa, M. Jantzen, and R. Valk, editors, *Foundations of Computer*

Science: Potential - Theory - Cognition, pages 201–208. Springer, 1997. Lecture Notes in Computer Science.

[Sch00] Jürgen Schmidhuber. Algorithmic theories of everything. Technical Report 20–00, IDSIA, 2000.

[WC98] Colin P. Williams and Scott H. Clearwater. *Explorations in Quantum Computing*. Springer, 1998.