Epistemic Computation and Artificial Intelligence

Jiří Wiedermann[1(✉)] and Jan van Leeuwen[2(✉)]

[1] Institute of Computer Science of AS CR, Prague, Czech Republic
jiri.wiedermann@cs.cas.cz
[2] Department of Information and Computing Sciences, Utrecht University, Utrecht, The Netherlands
J.vanLeeuwen1@uu.nl

Abstract. AI research is continually challenged to explain cognitive processes as being computational. Whereas existing notions of computing seem to have their limits for it, we contend that the recent, epistemic approach to computations may hold the key to understanding cognition from this perspective. In this approach, computations are seen as processes generating knowledge over a suitable knowledge domain, within the framework of a suitable knowledge theory. This, machine-independent, understanding of computation allows us to explain a variety of higher cognitive functions such as accountability, self-awareness, introspection, free will, creativity, anticipation and curiosity in computational terms. It also opens the way to understanding the self-improving mechanisms behind the development of intelligence. The argumentation does not depend on any technological analogies.

1 Introduction

Computation has proved to be a powerful framework for understanding cognitive processes. The all-important question is how computation can explain and model them, whether our understanding of computation is sufficient for it, and how a sufficiently general theory might be developed that transcends the realm of concrete algorithmic models.

In computer science, computations are traditionally seen as processes performed by computers. In order to make this definition more precise we must fix the notion of a computer. In theory, we commonly use the Turing machine for it since we know that, in principle, this model captures the computational ability of a large class of contemporary digital computers. Nevertheless, in practice, especially in artificial intelligence, biology and physics, one often considers computations in broader terms, not only including the processes taking place in computers and related artifacts, but also those in cells or plants, in the brains of people or animals, or even in the Universe as a whole.

The research of the first author was partially supported by the ICS AS CR fund RVO 67985807 and the Czech National Foundation Grant No. 15-04960S.

© Springer Nature Switzerland AG 2018
V. C. Müller (Ed.): PT-AI 2017, SAPERE 44, pp. 215–224, 2018.
https://doi.org/10.1007/978-3-319-96448-5_22

Hence, in order to see how the computational framework might apply, we have to investigate the nature and limitations of computations in the diverse contexts in which they emerge and by the devices that seem to perform them. In particular, much has been learned about the problem, *how* computations are realized by the various kinds of devices. This view is theoretically and technically interesting and has led to the development of e.g. classical computability theory and to important approaches to the understanding of computational complexity, indeed. However, these theories are dependent on the chosen model and, even worse, they tend to lack the potential to capture the meaning, or aim, of the computations that are being modeled. In other words, the problem *what* computations should do or be for us, or for the device realizing a computation, remains largely unanswered by it.

When asking what computations do, we contend that the only reasonable answer is that they produce *knowledge* of some kind. Here we take knowledge to be knowledge in the general sense, be it declarative or procedural or otherwise. For example, we see the causing processes of actions or behaviors as computational, as they generate the knowledge needed for an agent to trigger or perform them. This view has been the starting point of the so-called *epistemic theory of computation* proposed by the authors some years ago (cf. Wiedermann and van Leeuwen 2013, 2014, 2015a,b, 2017). Following this theory, computations are seen as processes generating knowledge over a suitable domain, in the framework of a suitable epistemic theory.

In this paper we show that the epistemic approach to computations has great potential especially for Artificial Intelligence. We will argue that it gives us a natural way to define and explain non-trivial cognitive functions like accountability, self-awareness, introspection, knowledge understanding, free will, creativity, anticipation, and curiosity as specific ways of computation, i.e. of particular kinds of knowledge generation. This brings new insight into the surmised algorithmic mechanisms behind intelligence, in a way that does not depend on any specific model of computation.

The latter feature is a considerable advantage of the epistemic model over other approaches to computational cognitive systems as described in the literature, which often use far more complex models of cognition. These models, usually based on concrete algorithmic mechanisms, tend to lead to cumbersome definitions, with descriptions depending more on the architecture and the special properties of the underlying model than on the general properties of the cognitive functions under consideration. A detailed account of the realized cognitive architectures can be found in Samsonovich (2010) and in the related on-line catalog. For the introduction to the theory of epistemic computation, its justification and examples of its viability, see Wiedermann and van Leeuwen (2013).

The structure of the paper is as follows. In Sect. 2 we present the main ideas of the epistemic approach to computation. In Sect. 3 we explain how this approach leads to machine- and algorithm-independent definitions of important higher cognitive functions, including also the notion of self-improving epistemic theories. Finally, in Sect. 4 we give some conclusions.

2 Computation as Knowledge Generation

Before digressing on the epistemic approach to computation, we need to be more specific about the way we view *knowledge*. Informally, knowledge in our framework is knowledge in the usual sense of this word. For our purposes we allow both declarative and procedural knowledge as they are commonly distinguished, and any other form of knowledge that might be implicitly or explicitly acquired somehow. Thus, knowledge could be facts or specific information of some kind but also knowledge of the 'know-how' kind, including the underlying rules of actions or skills and even of behaviors. For the treatment of skills in our approach, see the short remark at the end of this section.

When defining knowledge in accordance with the epistemic theory of computation, we need to be somewhat more specific. We will see knowledge as the result of a certain computational process, working over a given *knowledge domain*. This process combines known elements of the domain—so-called *elementary knowledge*—into derived, often more complex constructs that represent new knowledge over the given domain. For combining elements from the domain, the computation makes use of *derivation rules* which are either embedded in the process or proceed in interaction with the environment. The rules can be known beforehand or may be learned, in the latter case through the potentially endless processing of many computations over the given domain. Hence, a computation works with a more or less formal theory, capturing the given knowledge elements of the underlying epistemic domain as well as the ways of inferring new elements within that domain.

2.1 Computational Processes

As mentioned above, we are not interested in *how* a computation proceeds by means of a mechanism of some kind, but rather in *what* it does, i.e., in what knowledge is generated in the course of a computation. Under this viewpoint, the ability to generate knowledge becomes the hallmark of those processes that we will call computational. *Intelligent systems* are special instances of computational processes or sets thereof, which are able to generate knowledge over knowledge domains that model large parts of the real world, of various sciences, or of any other specific areas that are amenable to knowledge generation. This contrasts with the practice of contemporary AI systems, or "ordinary" computations that, as a rule, are specialized to, in most cases, considerably restricted knowledge domains. Within the epistemic theory of computation, the processes that do not generate knowledge are not considered as computations.

The question how (generated) knowledge can be identified or recognized is a difficult philosophical problem. From a practical point of view, one normally agrees that it is an *observer-dependent* matter - what constitutes knowledge can be obvious for one person and completely unclear to an other. That is, what is knowledge depends on how knowledgeable a person (or an AI system) already is in the domain under consideration. In the epistemic approach to computation we therefore always define knowledge in the framework of the *knowledge domain* over

which a computation operates. All knowledge about (a subset of) the knowledge domain at hand is to be captured by a corresponding *knowledge theory* which can be more or less formal, or completely informal.

In the knowledge theory of a domain, *axioms* describe the available *elementary knowledge* corresponding to the (representations of) objects in the domain and their properties. The ways in which new, derived knowledge can be constructed from elementary knowledge are described by *inference* or *derivation rules*. In fact, there are also other constructive methods for knowledge generation, used e.g. in ancient mathematics (cf. Sloman 2018), but here we concentrate solely on theories that are based on logic-based inference. Then, computational processes are bound to their knowledge domain through the corresponding knowledge theory, via the following condition:

> *whatever can be derived within the given theory must be supported by the corresponding computational process (and vice versa).*

Furthermore, for any computational process, there must be evidence concerning the validity of the latter condition. If the condition holds, then what knowledge can or cannot be generated over the given knowledge domain, and the "quality" of this knowledge (e.g., its agreement with observations), depends solely on the properties of the underlying knowledge theory.

In other words, in order for a computation to generate knowledge there must be evidence that explains that the computational process works as expected. This evidence must establish two facts: *(i)* that the generated knowledge can be derived within the underlying epistemic theory; this is provided via a 'proof' in that theory, and *(ii)* that the computational process generates the desired knowledge; this is also done via a proof, in a formalism capturing the actions of the computational process. The latter is the key to the following more formal definition (cf. Wiedermann and van Leeuwen (2014)). In this definition we assume that the input to a computation is part of both the underlying epistemic domain (and thus of the theory) and the initial data of the process.

Definition 1. *Let T be a theory, let ω be a piece of knowledge serving as the input to a computation, and let $\kappa \in T$ be a piece of knowledge from T denoting the output of the computation. Let Π be a computational process and let E be an explanation. Then we say that process Π acting on input ω generates the piece of knowledge κ if and only if the following two conditions hold:*

- *$(T, \omega) \vdash \kappa$, i.e., κ is provable within T from ω, and*
- *E is the (causal) explanation that Π generates κ on input ω.*

We say that the 5-tuple $C = (T, \omega, \kappa, \Pi, E)$ is a computation rooted in theory T which, on input ω, generates knowledge κ using computational process Π with explanation E.

Although the notation used in the definition resembles the one used in the formal theories, we will be using it equally in the case of informal epistemic domains and theories.

Note that the epistemic approach to computations is *machine independent*, since it holds regardless of the mechanism supporting knowledge generations in the theory at hand. It is also *algorithm independent* since we do not care how the computational process operates. Last but not least, the approach is *representation independent* since we do not need to assume any particular knowledge representation. For a more detailed account of this approach to computation and knowledge generation, see Wiedermann and van Leeuwen (2014), (2015a).

2.2 Intelligent Systems and Their Knowledge Theories

Thanks to its generality, the epistemic approach is well suited to be applied both in well formalized, so-called *theory-full knowledge domains,* and in knowledge domains with inference rules that resist any formalization efforts (so-called *theory-less domains*). The prototypical example of a theory-less domain with informal derivation rules is the 'real world'. Its objects, phenomena, and actions, and the relations among them, are described in a natural language. Knowledge about such a domain is captured by sentences in a natural language again. In this case, derivation rules are the rules of rational thinking and behavior. These rules are based on facts and arguments that can be described in natural language. In typical cases, theory-less domains have large knowledge bases (think, e.g., of the Internet) and relatively short derivation chains.

A prime example of an intelligent system is the human brain. In the brain, knowledge is generated by processes following an informal theory which can be described, however, in a natural language. In principle, in stead of the brain one may consider any other kind of intelligent artifact with similar properties, even a not-yet-existing one. The result would be the same: a general artificial system on par with human intelligence. The fact that the epistemic approach allows one to work with poorly formalized notions is a strong point of our modeling. It opens the door to an computational understanding of the mechanisms of knowledge generation that so far was not possible by other means.

Note that the idea of epistemic computation surely works well for systems whose main task is "thinking", i.e., solving purely intellectual problems. However, when we want to apply our approach to the functions of a robot whose main purpose is to perform some prevailingly physical tasks, in addition to speech recognition and interaction, we are in a more difficult situation. This is because such tasks require sensory-motor skills which use the 'physics' of the robot and are not clearly 'knowledge-driven' only. Thus, these skills cannot, probably, be adequately captured in a sufficiently formalized and manageable theory that would drive the behavior of the robot at hand. A solution of this problem may depend on our ability to invent new types of languages similar to how languages were 'invented' a long ago by biological evolution and used by our brains. It may not be formalizable using current logical and grammatical notations, but those are relatively recent discoveries. In the future, one may discover what evolution has achieved and how it was done. We may then be able to replicate the mechanisms in future, more intelligent machines.

A more immediate solution could be based on a hybrid approach, delegating the realization of skills to specialized, pre-trained, "trusted" specialized (deep) neural nets whose correctness is based on statistical evidence. Their ability to perform the designated tasks follows from their construction, after they were trained on large sets of data. The cooperation of these nets is then governed by some epistemic theory with checkable proofs (cf. Shanahan (2016) for a similar approach).

A more formal explanation of our approach, presented only informally in this paper, can be found in van Leeuwen and Wiedermann (2017).

3 Epistemic Computation and Higher Cognitive Functions

Let us return to the condition (cf. Sect. 2.1) that binds a computational process to a specific epistemic theory of its domain. We also required that for any (piece of) knowledge that can be derived in the epistemic theory, there must be evidence that the computational process supports (realizes) the resulting knowledge. This evidence need not necessarily be a logic-based proof. It could use a wider range of forms of representation, as used in ancient mathematics (cf. Sloman 2018), or in the reasoning about chemical or physical processes.

In this section we will show that a cognitive system working in this way, can generate knowledge corresponding to the definition of many non-trivial cognitive functions. That is, within the epistemic framework we can naturally define and explain many higher cognitive functions as knowledge generating and, hence, computational processes, which would be cumbersome otherwise.

In what follows we describe the respective higher cognitive function informally, but this will be sufficient to bring them into the framework of epistemic computations. There are a least three reasons for doing so. First, we do not aim at a descriptive model of the cognitive functions, as these are amply considered in the literature of the cognitive or brain sciences. Secondly, there is often no generally accepted definition of these functions. Finally, our goal is primarily to demonstrate that these functions are all of a computational nature, in the context of artificial intelligence. It is an advantage of epistemic approach that it allows us to concentrate on the knowledge generating aspects of these functions, without having to take the specificity of their definition into account.

We now briefly characterize and discuss nine higher order cognitive functions in turn, to illustrate the epistemic approach.

(a) Accountability: an ability of a system to generate knowledge justifying its own decisions. This means e.g. that a cognitive agent can, upon request or otherwise, issue a substantiation of his doings, i.e. an explanation of how he made out his findings, cf. Kroll *et al.* (2016). To this end it is enough for the agent to generate, together with the resulting knowledge, also the proof for its derivation. Of course, this proof should be presented in a formalism accessible to the user. This enables the user to review and check whether the agent's justifying information is correct and complete, within the framework of the underlying epistemic theory.

(b) Awareness: an ability of a cognitive system to have knowledge about the problem being solved. Given accountability, this knowledge is available and an agent can report what is it doing, if asked to do so. Self-knowledge can be a part of the epistemic theory controlling the activities of the system.

(c) Introspection: an ability of a system to recall knowledge about its previous actions and their derivation. A cognitive system can store its previous tasks and the way of their solutions. Doing so, it can return to them, to re-inspect them and make use of them when solving new problems by means of analogy. It can also improve upon previous solutions w.r.t. new findings that the system could have accumulated in the mean time.

(d) Epistemic understanding. Accountability, (self-)awareness and introspection together give rise to understanding the knowledge domain over which a system operates. A system is able to explain the meaning of the terms it works with and, based on its previous experience (recorded in its knowledge base), to apply them in new contexts. For a full understanding of the real world, one has to consider embodied cognitive systems.

(e) Free will. We say that cognitive system A has free will with respect to cognitive system B if and only if, based solely on the observation of A's actions, B is not able to always predict (in the form of generated knowledge) A's future actions in concrete situations. This definition differs from numerous definitions of free will (cf. Wikipedia) that see the concept from an inner (subjective) view of a system. For instance "in a given situation, free will is the ability to choose from several alternative behaviors", or "an ability to behave differently than in the past under the same conditions". Only a cognitive system itself has information whether it has chosen its behavior from several possibilities, or whether it has "invented"a new behavior. Assuming that this information is inaccessible for an external observer, there is no way for such an observer to decide whether the system at hand possesses free will or not. This fact makes our definition of free will observer dependent. However, the advantage is that it identifies the problem of free will as computational w.r.t. the given observer.

(f) Creativity: a manifestation of a creative process, which is any process generating a solution to a problem (in the form of knowledge) that is new for the given cognitive system. Its counterpart is a routine process, which solves a known problem with the help of known procedures. In general, a creative process seeks explicit knowledge that is given implicitly via conditions that the knowledge to be found must satisfy. Deferring efficiency aspects, in our approach the basic strategy for a creative process is the exploitation of "brute force"—the systematic examination of all knowledge that can be generated in the framework of a given theory and checking whether the generated knowledge satisfies the given conditions. This looks like an extremely inefficient, even naive approach, but it seems that such an approach is the basis of any creative process. In fact, it is a special case of *knowledge discovery.* This initially inefficient, but universal process of knowledge discovery is cultivated in the course of its repeated use. Knowledge discovery is then seen as a potentially never ending, evolutionary

self-improving learning process whose goal is to make its creative abilities more efficient. In Wiedermann and van Leeuwen (2015b) we have described several basic techniques that can be used in the cultivation of creative processes:

- *interactive refinement:* this entails a modification of the search criteria, based on the experience from previous, unsatisfactory or even failed searches.
- *automatic extraction and modification of user preferences:* this narrows the search space. It is based on the observed preferences of the user (in a similar way as done by Google+) and/or on the basis of user's emotions and subjective experience (if this is accessible to the system). This activity of collecting and shaping user preferences goes on at every occasion when solving any creative problem. The extracted preferences are exploited for streamlining any discovery process and for ordering the discovered solutions.
- *guided interaction with the environment:* its aim is to gain new, additional knowledge that can help in solving the given concrete problem (in much the same way as we do when checking the Internet for additional information).

As a result of such cultivation procedures, the knowledge base of the creative process as well as the mechanisms of its exploitation are modified or supplemented.

Note that the mechanisms described above not only answer the question *what* knowledge must be generated in order to solve a problem, but at the same time they also answer the question *how* such knowledge can be found, i.e., in fact, *how* to solve the given problem. A special case is the question *how* to improve some aspects of a known solution of a problem. This question can be transformed into the question of *what* knowledge must be found solving a given problem and, at the same time, satisfying a new condition referring to any aspect of a solution that must be improved.

(g) Anticipation: an ability of a system to generate knowledge in the form of predictions about the future occurrence of events or conditions in an epistemic domain. It is seen as the result of a "wired" creativity, a limiting result of creativity cultivation where no search is necessary in order to solve the problem. Consequently, anticipation becomes a routine process working as the first choice alternative or an efficient substitute of an originally creative process.

(h) Epistemic curiosity: a perpetual need of a system to discover new knowledge. It is intimately related to creativity and anticipation. Similar to creativity, curiosity is a life-long learning process whose cultivation causes that not everything is explored and exploration is not made randomly. Curiosity is often invoked when anticipation fails.

(i) Epistemic self-improvement. The mere ability to derive new knowledge in the framework of a given epistemic theory cannot be considered to be the main attribute of intelligent systems. Namely, the main attribute of an intelligent system is its ability to improve its own epistemic theories through which it generates its knowledge. If this is the case, then the intelligence of such a system keeps provably increasing.

When using cultivation procedures as described above, it may happen that a system gets new knowledge that contradicts the knowledge already possessed by the system. Such contradictory knowledge can be derived by the system itself or it can enter the system from "outside" (e.g., from the Internet), or when the system reveals a discrepancy between its own observations with its epistemic theory. Such a flaw can only be cured by a change of the underlying theory.

Systems that have mechanisms for discovering and repairing logical inconsistencies in their theory obviously can increase their intelligence under any reasonable definition of this notion. This process can continue as long as there exist contradictory facts within the theory and the system at hand can find them, and as long as there exist unexplored objects and phenomena in the underlying knowledge domain. As a result, such systems can potentially, at least in some domains, overcome human intelligence (Wiedermann and van Leeuwen 2017). Unlike the popular idea of software self-improvement that aims at streamlining derivation procedures in a cognitive system (cf. Bostrom 2014), self-improvement of knowledge theories aims at the heart of the intelligence—viz. the quality and quantity of the epistemic data.

4 Conclusions

In this paper we have applied the framework of epistemic computation to the definition of various higher cognitive functions. The aim was to present them as knowledge generating functions, i.e., in fact, as computational processes. We have also elucidated the mechanisms of self-improving epistemic theories that lie behind the development of intelligence. In doing so we have focused, in accordance with the philosophy of the epistemic approach to computation, on the question *what* the cognitive functions under consideration do, i.e., what knowledge they are producing, rather than on *how* they do what they do. This leads to a new approach to understanding these functions. They cannot be described in a simple and elegant way using the classical view of computations, generated by various models of computers. This is because such a view is necessarily machine dependent and therefore cannot offer a sufficiently abstract and general framework for defining and understanding the functions at hand. Contrary to this, with the help of an elementary, abstract model of cognitive systems that is not burdened by any technical details, our approach clearly points to the conclusion that all cognitive functions under consideration are related to specific forms of knowledge generation, within an appropriate epistemic theory.

Viewing computations as knowledge generating processes has great potential for AI. In future work, we intend to apply the epistemic approach also to the problem of consciousness. This would present an essential contribution to the philosophy and theory of computational cognitive systems.

Acknowledgment. The authors thank Jodi Guazzini and Aaron Sloman for comments and suggestions that greatly helped to improve the manuscript.

References

Bostrom, N.: Superintelligence: Paths, Dangers, Strategies. Oxford University Press, Oxford (2014)

Garnelo, M., Arulkumaran, K., Shanahan, M.: Towards deep symbolic reinforcement learning. arXiv:1609.05518 (2016)

Kroll, J.A., et al.: Accountable algorithms. Univ. PA Law Rev. **165**(3), 633–705 (2016). Available at SSRN. https://ssrn.com/abstract=2765268

Samsonovich, A.V.: Toward a unified catalog of implemented cognitive architectures. In: Proceedings of BICA 2010. Frontiers in Artificial Intelligence and Applications, vol. 221. pp. 195–244. IOS Press Ebooks (2010). http://bicasociety.org/cogarch/

Sloman, A.: Huge but unnoticed gaps between current AI and natural intelligence (2018). This volume

van Leeuwen, J., Wiedermann, J.: Knowledge, representation and the dynamics of computation. In: Dodig-Crnkovic, G., Giovagnoli, R. (eds.) Representation and Reality in Humans, Other Living Organisms and Intelligent Machines, pp. 69–89. Springer, Cham (2017)

Wiedermann, J., van Leeuwen, J.: Rethinking computation. In: Proceedings of 6th AISB Symposium on Computing and Philosophy: The Scandal of Computation - What is Computation?, AISB Convention 2013, Exeter, UK, pp. 6–10. AISB (2013)

Wiedermann, J., van Leeuwen, J.: Computation as knowledge generation, with application to the observer-relativity problem. In: Proceedings of 7th AISB Symposium on Computing and Philosophy: Is Computation Observer-Relative?, AISB Convention 2014, Goldsmiths, London. AISB (2014)

Wiedermann, J., van Leeuwen, J.: What is computation: an epistemic approach. (Invited talk.) In: Italiano, G., et al., (eds.) SOFSEM 2015: Theory and Practice of Computer Science. Lecture Notes in Computer Science, vol. 8939, pp. 1-13. Springer (2015a)

Wiedermann, J., van Leeuwen, J.: Towards a computational theory of epistemic creativity. In: Proceedings of 41st Annual Convention of AISB 2015, London, pp. 235–242 (2015b)

Wiedermann, J., van Leeuwen, J.: Understanding and controlling artificial general intelligenct systems. In: Proceedings of 10th AISB Symposium on Computing and Philosophy: Language, Cognition and Philosophy, AISB Convention 2017. University of Bath, UK, AISB, pp. 356–363 (2017)