Tractable embodied computation needs embeddedness

van Rooij, I.; Blokpoel, M.; de Haan, R.; Wareham, T.

DOI
10.12832/94728
10.12832/94728

Publication date
2019

Document Version
Submitted manuscript

Published in
Reti, Saperi, Linguaggi

Tractable embodied computation needs embeddedness

Iris van Rooij
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

Mark Blokpoel
Radboud University, Donders Institute for Brain, Cognition and Behaviour, The Netherlands

Ronald de Haan
Institute for Logic, Language and Information, University of Amsterdam, The Netherlands

Todd Wareham
Department of Computer Science, Memorial University of Newfoundland, Canada

Abstract

It seems that a move from discrete to analog computation demands new types of complexity theory to demarcate the tractability border for embodied computation. We argue, however, that application of complexity theory need not hinge on the analog versus discrete distinction as much as one may think. Instead, we will show that *embeddedness*, one of the core commitments of enactivism, proves more critical for understanding tractable embodied computation.

Keywords: Enactivism, Embodied Cognition, Computation, Tractability

Enactivist approaches to cognition have long had a difficult relationship with computationalism. The relationship seems to be improving as more and more theorists spell out ways in which the embodied, embedded, enactive, and extended (4E) nature of cognition on the one hand, and computationalism on the other, are not incompatible (Milkowski, 2018; van Rooij, 2012; Villalobos & Dewhurst, 2017, 2018; c.f. Abramova, 2019). Isaac (n.d.) contributes to this development, as he proposes analog computation such as embodied in fluid logics as a framework for thinking about enactivist computational explanations. In this commentary, we zoom in on a question raised by Isaac: What, if any, are the implications of this proposal for understanding the conditions that make enactive cognition tractable?

Isaac sees two possibilities. The first possibility is that tasks that are intractable for digital computation are also intractable for analog computation, and vice versa. In other words, in this case the tractable-intractable border is insensitive to whether or not the computation is achieved by a discrete or analog computation and, consequently, the same tractability constraints that have been argued to apply to cognitivist accounts (see, e.g., Frixione, 2001; van Rooij, 2008) would also naturally apply to embodied accounts of cognition (see, e.g., van Rooij, 2012). The second possibility is that there exist computational transformations that can be tractably computed by analog computation but not by classical (Turing machine) computation. In that case, not only would assessing the tractability of embodied computation require its own, non-classical form of complexity analysis, but it may also speak in favour of embodied accounts whenever cognitivist accounts run into intractability but the embodied (analog computation) account does not.

In this commentary, we argue that regardless of which of these two possibilities holds, enactivists can benefit greatly from adopting the tractability constraint, as it will provide a means to explicate the important role that one of its E's plays in understanding the tractability of cognition in the world. The reason is, as we explain, that tractability cannot be achieved without proper embeddedness. Moreover, this is true, regardless of whether the computation performed is discrete or analog, and whether or not those are ultimately equivalent as far as tractability is concerned.

We will first briefly explain the formal notions from computational complexity theory and intractability on which our argumentation builds. We next consider the conditions under which the computations performed by an enactive, embodied agent can be tractable. We will show that this requires either (1) that the environment is structured such that the situations that arise for the agent, and that it can sense, are a proper subset of all logically possible situations; or (2) that the agent is made tractable by design in which case its fit with the environment must have been determined by 'design' (e.g., evolution).

1 Computational Complexity & Intractability

Computational complexity theory provides mathematical tools and concepts for assessing the resources (e.g., time, space, randomness) consumed during computation. We focus here on the resource time, as it is known to be one of the most sensitive resources (e.g., more so than space—see, e.g., Arora and Barak, 2009). One important distinction in computational complexity theory relevant for our purposes is the notion of 'intractability'. Two distinct formalisations of this notion have been put forth, one based on classical complexity theory (see, e.g., Arora & Barak, 2009; Garey & Johnson, 1979) and one based on parameterized complexity theory (see, e.g., Downey & Fellows, 2013; Flum & Grohe, 2006).[1] We next consider each in turn.

1.1 Classical Complexity Theory

According to classical complexity theory (Arora & Barak, 2009; Garey & Johnson, 1979), a computation is intractable if the time that it consumes grows faster than any polynomial function (i.e., faster than n^c, where c is some constant and n is some measure of the problem size, e.g., number of variables or states). Such computations are called non-polynomial time computations. An example is a computation that takes exponential-time, on the order of c^n for some constant $c > 1$. Even for intermediate problem sizes n, exponential time becomes prohibitively large (e.g., for $c = 2$ and $n = 60$ the number c^n is more than the number of

[1] For an accessible treatment of both types of complexity theory applied in cognitive science, we refer to (van Rooij, Blokpoel, Kwisthout, & Wareham, 2019)

seconds that have passed since the dinosaurs went extinct). The embodied computations of enactive agents surely should operate on a time scale relevant for survival. In practice, this will be on the order of seconds, minutes, and in rare occasions hours. An organism that instead takes, say, millions of years to come to a decision about whether or not to eat a given food option will be dead before it can take the first bite.

An example of a classically intractable computation is 3-SAT.[2] In 3-SAT, one is given a logical formula in conjunctive normal form: i.e., the formula is a conjunction of a set of clauses of disjunctions of at most 3 variables, e.g., $(v_1 \vee v_2 \vee \neg v_3) \wedge (v_5 \vee \neg v_1 \vee v_3) \wedge (v_6 \vee \neg v_4)$. The result of the computation is a truth assignment to the variables that makes the whole formula true (e.g., the assignment v_2, v_4, $v_6 = true$ and v_1,v_3, $v_5 = false$ would make the above formula true). There exists no polynomial time procedure for computing 3-SAT. However, the problem 2-SAT, in which each clause is a disjunction of at most 2 variables, is tractable, i.e., it can be computed in polynomial time.

1.2 Parameterized Complexity Theory

According to parameterized complexity theory (Cygan et al., 2015; Downey & Fellows, 2013; Flum & Grohe, 2006; Niedermeier, 2006), computations are tractable when they are so-called fixed-parameter tractable for *small* parameters. A computation is said to be *fixed-parameter tractable* when the time it consumes can be upper bounded by some function $f(k)n^c$, where f can be an arbitrary function of the parameter k. Since the n^c-part is polynomial-time, fixed-parameter tractable computations are tractable even for large n provided only that k is relatively small. Note that classically intractable computations can be fixed-parameter tractable. For instance, we saw that 3-SAT is classically intractable. Yet, 3-SAT is fixed-parameter tractable when parameterized by the number of variables that appear negated in the formula.[3]

1.3 Invariance thesis

Even though both classical and parameterized complexity theory were originally formulated for Turing machines, according to the Invariance thesis the distinction between polynomial versus non-polynomial (for classical complexity) and between fixed-parameter tractable and fixed-parameter intractable (for parameterized complexity) is insensitive to the exact machine model used.[4] As Isaac notes, it is not known if any future analog computer model may show that the Invariance thesis is false. Yet, at present there is no reason to believe that it is. For instance, Siegelmann and Sontag (1994) have shown that analog neural networks, even with infinite precision real valued connection weights, cannot perform intractable (NP-hard) computations in polynomial time. Similarly, Aaronson (2005) discusses various forms of physical computation, ranging from protein folding to quantum computing, and

[2] Formally, 3-SAT is a known to be NP-hard (Garey & Johnson, 1979, Problem L02). This means that it is not polynomial-time computable, unless P = NP. It is famously conjectured, however, that P $\neq$ NP (Fortnow, 2009; Gasarch, 2002).

[3] This is a folklore result, that can be proved by combining a simple pre-processing rule with results mentioned by Niedermeier (2006, Chapter 1).

[4] As Parberry (1986) wrote, the Invariance (or Extended Church-Turing) Thesis "states [...] that time on all 'reasonable' machine models is related by a polynomial."

concludes that the bounds of tractable computation seems to be a universal limit and an inherent property of physical computation generally, not just classical (Turing) machines.

Does this mean that the bounds of intractability are irrelevant for enactivist versus cognitivist debates about the nature of cognition? We think not. But we think that more relevant for understanding how enactivist approaches may be one step ahead of cognitivist approaches is that they view embeddedness of cognitive agents in their life world as a fundamental part of understanding cognitive behaviour. As we will argue, it is also vital for understanding the tractability of cognition.

2 An Enactive Agent Schema

Let's consider an embodied agent $A = (S, E, M)$ with a set of sensors $S = \{s_1, ..., s_k\}$, a set of effectors $E = \{e_1, ...e_m\}$, and an embodied control mechanism M. The agent is embedded in an environment, its *life world* $\mathcal{E}$ (see Figure 1; cf., Di Paolo, Buhrmann, and Barandiaran (2017)). In general, the agent's sensors and effectors may take on continuous values in a certain range (e.g., $[-1, +1]$) or binary values "on" or "off", or some combination of these. Even for medium sized sensor and effector sets, the sensory and action repertoire of our agent can already be quite substantial. Consider for instance, that forthe agent which has six sensors with an accuracy such that each sensor can distinguish about 10 different states in the $[-1, +1]$ range (e.g., light intensity, colours, temperature, pitch), there are already a million sensory states that this agent can distinguish (not necessarily phenomenologically, but in principle effectively). Similarly, if the agent has eight effectors, each with four degrees of freedom, then it can perform, in principle, more than 65,000 actions, where each action corresponds to an effector state. This can pose considerable control problems for an embodied agent: How can it (learn to) select those actions that are adaptive for its life world?

We consider the process of selecting actions for an agent A given sensory information as a state transition function f, denoted as follows:

$$A, \overline{s_t}, \overline{m_{t-1}}, \overline{e_{t-1}} \xrightarrow{f} A, \overline{s_{t+1}}, \overline{m_t}, \overline{e_t} \tag{1}$$

Here, $\overline{s_t}$, $\overline{m_t}$ and $\overline{e_t}$ are the value assignments at time t to the sensors in S, internal states of M and effectors in E respectively.

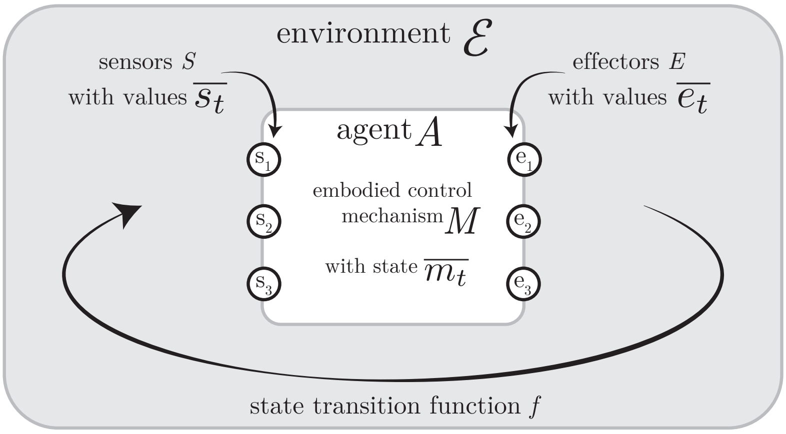
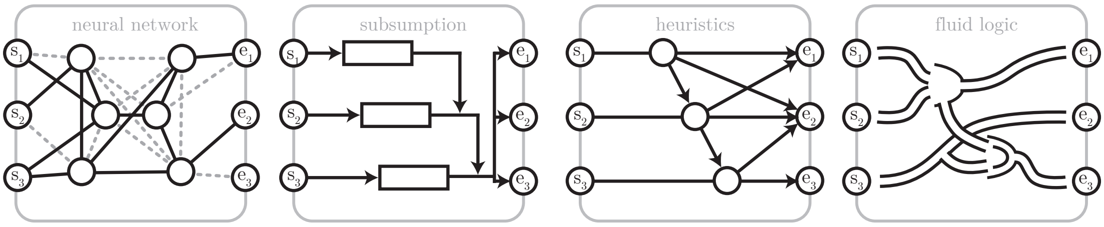


Figure 1. An agent A embedded in its life world $\mathcal{E}$.



(a) (b) (c) (d)

Figure 2. Example types of control architectures for embodied computation. a) Hopfield neural network. b) Brooks-type subsumption. c) Gigerenzer-type adaptive toolbox of heuristics. d) MONIAC fluid logic.

An embodied control architecture will need to control how the sensory states affect the effector states in such a way that agent can behave adaptively in its environment (e.g., minimally move around, find food and try not to die.) Enactivists may envision different types of control architectures M. For instance, a Hopfield neural network that selects those effectors that are most coherent (energy minimum) given the sensory inputs and the internal network structure (Fig. 2a) (Bruck & Goodman, 1990; Thagard & Verbeurgt, 1998; Xu, Hu, & Kwong, 1996), M can be a Brooks-type subsumption architecture that given sensory states trigger the highest effector in the hierarchy that is not inhibited by any of the others (Fig. 2b) (Brooks, 1986), a Gigerenzer-type adaptive toolbox of heuristics that given sensory inputs decides based on cheap and dirty decision rules which effectors to activate (Fig. 2c) (Otworowska, Blokpoel, Sweers, Wareham, & van Rooij, 2018; Schmitt & Martignon, 2006), a MONIAC fluid logic mechanism that determines effector activation from sensor activation based on logic gates and fluid dynamics (Fig. 2d) (Isaac, n.d.) or a so far unknown architecture to be specified in the future. Importantly, though, for the agent to count as an enactive, and not cognitivist, we want the architecture to not involve any explicit planning and reasoning over representations of the world (and none of the abovementioned architectures do).[5]

3 The argument for embeddednes

To claim tractability of embodied computation, enactivists have access to two possible avenues: Either the control structure M is not *tractable-by-design* or it is. We consider each of these options and show that in each case *embeddedness* is central to understanding how an agent can tractably navigate its life world. We start, in Section 3.1, by considering the first option and consider the second option in Section 3.2.

3.1 Embedded tractable control

We say a control mechanism M is *not tractable-by-design* if there exist a logically possible world $\mathcal{E}$ (which need not be A's actual world) such that A could encounter control problems in $\mathcal{E}$ that are NP-hard. This is the case when there can exist value assignments to

[5]Traditionally, the adaptive toolbox of heuristics may be considered cognitivist. Nevertheless, its architecture as formalised in Otworowska et al. (2018) has no internal reasoning over representations other than mapping sensory conditions to actions, not unlike a Brooks-type subsumption architecture.

its sensor, internal and effector states such that the transition function is NP-hard. This would be the case, for instance, if the control mechanism M were a Hopfield neural network and the control problem is conceived as minimizing[6] the energy of the activation pattern (i.e., the value assignments over the internal and effector nodes in the network, given a value assignment for its sensory nodes).[7] Such an agent would have the computational capacity for solving NP-hard problems, because it is possible to so-called polynomial-time reduce known NP-hard problems (like 3-SAT) to the operation of the control system of such an agent.[8] This is not to say that an embodied agent A would explicitly solve such problems in the sense that the agent internationally or mentally represents such problems explicitly or even at all. It is merely to say that the transition function has sufficient degrees of freedom or expressive power such that if we were to have access to such an M then we, scientists or external observers, would be able to read off the states computed by the transition function and use this information for solving NP-hard problems. This proves formally that an agent with such an M is not tractable-by-design, as it can encounter control situations for which no polynomial-time algorithm (of any kind; see Section 1.3) can possibly exist.

How could such an agent manage to nevertheless navigate its life world tractably? There is only one way possible: the agent's life world $\mathcal{E}$ must be structured in such a way that the agent only encounters a proper subset of all the possible value assignments for its sensors *and* the control problem that M computes is tractable *for that subset*. In other words, tractability in this case will have to arise, in one way or another, from the *embeddedness* of the embodied agent.

This means that an enactivist program that wants to explain how enactive computation can be tractable, without enactive architecture being tractable-by-design, will need to characterize the specifics of the embeddedness that make tractable computation possible. This could be achieved if f is either polynomial-time computable or fixed-parameter tractable for the set of sensory inputs that can be induced by A acting in $\mathcal{E}$ (Haselager, van Dijk, & van Rooij, 2008; Wareham et al., 2011).

3.2 Embedded tractable adaptation

We say a control mechanism M is *tractable-by-design* if for all its possible sensory states and environments $\mathcal{E}$ the state transition function f is tractably (polynomial-time) computable.

[6] Nothing in our argument hinges on the assumption of optimality. First of all, non-optimality—be it by satisficing, heuristics, approximation or appeals to 'as if'—does not generally buy one tractability (see van Rooij, Wright, and Wareham (2012), van Rooij, Wright, Kwisthout, and Wareham (2018)). Second, even when it does, that renders the architecture tractable-by-design and then our argument as laid out in Section 3.2 applies.

[7] Another example would be a hybrid deliberative-reactive architecture, as analyzed by Wareham, Kwisthout, Haselager, and van Rooij (2011). Enactivists may object that the deliberative component disqualifies this architecture as an enactive one. However, given that in the analyses of Wareham et al. the deliberative component itself could operate in non-classical non-representational manner, e.g., like a Hopfield network, we think such judgment would be premature.

[8] Computing a minimum energy activation pattern for a Hopfield neural network is NP-hard (Šıma, 2001). This means that we can polynomial-time reduce 3-SAT to this problem. By carrying out this reduction from appropriate 3-SAT inputs (e.g., 3-CNF formulas used in the proof of Theorem 2.10 in Cadoli, Donini, Liberatore, & Schaerf, 2002), one can show that even for a fixed network, the problem of computing a minimum energy activation pattern corresponds to solving different (arbitrary) 3-SAT instances depending on the value assignment to the sensory nodes of the network.

This would be the case, for instance, for a Brooks-type reactive architecture, Gigerenzer-type adaptive toolbox of heuristics, and the MONIAC fluid logic mechanism proposed in Isaac (n.d.).[9] All three are feedfoward mechanisms and, accordingly, the transition function can be computed in polynomial time (in these specific cases, the time required is directly proportional to the size of the mechanism, and hence the transition function would be linear-time computable).

An agent with such a control mechanism purchases tractability at the cost of degrees of freedom or expressive power of its transition function. This means that it can only solve much simpler problems than an enactive agent that is not tractable-by-design (see Section 3.1). Despite its simple control structure, such an agent can still display complex behaviour. We are reminded of Simon's ant:

> "An ant, viewed as a behaving system, is quite simple. The apparent complexity of its behavior over time is largely a reflection of the complexity of the environment in which it finds itself." (Simon, 1996, p. 64)

How could such a simple agent ensure that its behaviour sufficiently fits its life world? Not being a cognitivist agent, it has no access to reasoning to weigh pros and cons of different actions nor can it internalise any non-cognitivist deliberation mechanism, as that would make it not tractable-by-design anymore (Wareham et al., 2011). Hence, its apparent fit must have been the result of an external, intergenerational adaptation process such as evolution by natural selection. However, tractability of that adaptation process cannot be taken for granted. For example, it has previously been shown that adapting Brooks-type reactive architectures and Gigerenzer-type toolboxes of heuristics is NP-hard in general (Otworowska et al., 2018; Wareham et al., 2011), and arguably the same holds for adapting the MONIAC fluid logic mechanism.[10]

Adaptation processes could be tractable if the environments in which adaptation is taking place are structured in such a way that the adaptation process only has to deal with a subset of all possible life worlds *and* adaptation is tractable *for that subset*. In other words, tractability will have to arise, in one way or another, from embeddedness of the adaptation process. Notably, this embeddedness goes beyond the nature of the embeddedness of the embodied agent required for sufficient fitness. This follows from previous complexity-theoretic results that establish that even if it is 'promised' that the environment $\mathcal{E}$ is properly structured such that there exists a way to adapt a fit embodied agent, the process of adapation can still be intractable.[11]

[9] Computing transition functions using MONIAC fluid logic mechanism is computationally equivalent to solving the Circuit Value Problem for Boolean circuits—both are solvable in polynomial time (see, e.g., Arora & Barak, 2009, Theorem 6.30).

[10] The NP-hardness proof of Otworowska et al. (2018) of adapting Gigerenzer-type toolboxes of heuristics can straightforwardly be adapted to work also for the problem of adapting other mechanisms. To make this work, the most important properties are (i) that the mechanisms are required to be upper bounded in size (and that the upper bound is polynomially smaller than the size of the environment, but not trivially small), (ii) that the mechanism's size limits the number of observations it can make about the environment, (iii) that the mechanism is sophisticated enough to compute if-then statements, and (iv) that carrying out the mechanism can be done in polynomial time. The MONIAC fluid logic mechanism satisfies these conditions (i)–(iv), so its adaptation problem is NP-hard.

[11] One can prove, reductio ad absurdum, that so-called 'promise'-adaptation (PA) is intractable (i.e., cannot

This means that an enactivist program that wants to appeal to evolutionary adaptation to not only explain how enactive computation can be tractable-by-design but also to ensure that embodied agents controlled by such architectures are fit in their life worlds will need to characterize the specifics of the embeddedness that make tractable adaptation possible. This could be achieved if the adaptation process is either polynomial-time computable (see Section 1.1) or fixed-parameter tractable (Section 1.2) for the set of environments $\mathcal{E}$ for which the adaptation process yields an agent A that is tractable-by-design

4 Conclusion

Isaac (n.d.) proposed that consideration of computational complexity (in particular, tractability) can be informative for enactive approaches as it can help define the class of cognitive capacities that are tractable for embodied cognitive architectures. This holds regardless of whether or not we assume that enactive computation is analog or digital, and whether or not the tractability border is the same or different for enactivist and cognitive architectures. We wholeheartedly agree with Isaac on this point. The intent of our commentary is to highlight the vital role played by embeddedness in this endeavor.

With our analysis we have demonstrated that, regardless of whether or not one assumes that embodied computation is tractable by design, embeddedness is central to understanding how embodied agents can successfully navigate their life worlds in a tractable manner. Of course, given that embeddedness is one of the E's in the 4E program, this observation is in harmony with this program. Yet, it is our impression that to date it has been underestimated how critical embeddedness actually is for understanding the *tractability* of embodied computation. While the role of embeddeness in successful behaviour has been generally acknowledged, it seems that the 4E program seeks the solution for intractability in the type of control mechanism, e.g., analog, parallel, dynamical and/or non-representational (van Gelder, 1995, 1998). However, 4E may have been looking for a solution for intractability in a place where it cannot be found, and has overlooked that the key solution lies in one of its E's. For understanding tractable embodied computation, embeddedness should be given its proper status. In order to achieve this, enactivists will have to take on the task of explicitly modeling the environment $\mathcal{E}$ and studying those properties of $\mathcal{E}$ that render embodied control and adaptation tractable.

5 Acknowledgements

The authors thank Anco Peeters, Joe Dewhurst and an anonymous reviewer for helpful comments on an earlier version of this commentary.

be solved in polynomial time, unless P=NP). Assume that PA can be solved in polynomial time. It is then possible to solve Satisfiability (SAT; Garey & Johnson, 1979) in polynomial time by first polynomial-time reducing SAT to PA and then solving PA in polynomial time. There are three possible outputs of PA in this situation: (1) the promise holds, PA returns an adapted agent, and hence the solution to SAT is 'satisfiable'; (2) the promise does not hold, PA returns a maladapted agent, and hence the solution to SAT is 'unsatisfiable'; or (3) the promise does not hold, PA returns an uninterpretable solution, and hence the solution to SAT is 'unsatisfiable'. This implies that SAT can be solved in polynomial time. However, this contradicts our assumption that $P \neq NP$, as SAT is known to be NP-hard. Hence our assumption that PA can be solved in polynomial time must be false.

References

Aaronson, S. (2005). Guest column: NP-complete problems and physical reality. *ACM SIGACT News*, *36*(1), 30–52.

Abramova, E. (2019). *Joint action: an enactive mechanistic account* (Doctoral dissertation, Radboud University, Nijmegen, The Netherlands).

Arora, S. & Barak, B. (2009). *Computational complexity – a modern approach*. Cambridge University Press.

Brooks, R. A. (1986). A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, *RA-2*(1), 14–23.

Bruck, J. & Goodman, J. (1990). On the power of neural networks. *Journal of Complexity*, *6*, 129–135.

Cadoli, M., Donini, F. M., Liberatore, P., & Schaerf, M. (2002). Preprocessing of intractable problems. *Information and Computation*, *176*(2), 89–120.

Cygan, M., Fomin, F. V., Kowalik, L., Lokshtanov, D., Marx, D., Pilipczuk, M., ... Saurabh, S. (2015). *Parameterized algorithms*. Springer.

Di Paolo, E., Buhrmann, T., & Barandiaran, X. (2017). *Sensorimotor life: an enactive proposal*. Oxford University Press.

Downey, R. G. & Fellows, M. R. (2013). *Fundamentals of parameterized complexity*. Berlin: Springer.

Flum, J. & Grohe, M. (2006). *Parameterized complexity theory*. Berlin: Springer.

Fortnow, L. (2009). The status of the P versus NP problem. *Communications of the ACM*, *52*(9), 78–86.

Frixione, M. (2001). Tractable Competence. *Minds and Machines*, *11*(3), 379–397.

Garey, M. R. & Johnson, D. S. (1979). *Computers and intractability: A guide to the theory of NP-completeness*. San Francisco, CA: W. H. Freeman.

Gasarch, W. I. (2002). The P =? NP poll. *ACM SIGACT News*, *33*(2), 34–47.

Haselager, W., van Dijk, J., & van Rooij, I. (2008). A lazy brain? Emb0died embedded cognition and cognitive neuroscience. In P. Calvo & T. Gomila (Eds.), *Handbook of ciognitive science: an embiodied approach* (pp. 273–290). Elsevier.

Isaac, A. M. C. (n.d.). Embodied cognition as analog computation. *Italian Journal of Cognitive Science*, (this issue).

Milkowski, M. (2018). Embodied cognition meets multiple realizability. *Reti, saperi, linguaggi, Italian Journal of Cognitive Sciences*, (2/2018), 349–364

Niedermeier, R. (2006). *Invitation to fixed-parameter algorithms*. Oxford University Press.

Otworowska, M., Blokpoel, M., Sweers, M., Wareham, T., & van Rooij, I. (2018). Demons of ecological rationality. *Cognitive Science*, *42*(3), 1057–1066.

Parberry, I. (1986). Parallel speedup of sequential machines: a defense of the parallel computation thesis. *ACM SIGACT News*, *18*(1), 54–67.

Schmitt, M. & Martignon, L. (2006). On the complexity of learning lexicographic strategies. *Journal of Machine Learning Research*, *7*(Jan), 55–83.

Siegelmann, H. T. & Sontag, E. D. (1994). Analog computation via neural networks. *Theoretical Computer Science*, *131*(2), 331–360.

Simon, H. A. (1996). *The sciences of the artificial* (3rd). The MIT Press.

Šıma, J. (2001). Energy-based computation with symmetric hopfield nets. In S. Ablameyko, M. Gori, L. Goras, & V. Piuri (Eds.), *Limitations and future trends in neural computation* (pp. 45–69). IOS Press.

Thagard, P. & Verbeurgt, K. (1998). Coherence as constraint satisfaction. *Cognitive Science*, *22*(1), 1–24.

van Gelder, T. (1995). What might cognition be, if not computation? *The Journal of Philosophy*, *92*(7), 345–381.

van Gelder, T. (1998). The dynamical hypothesis in cognitive science. *Behavioral and brain sciences*, *21*(5), 615–628.

van Rooij, I. (2008). The Tractable Cognition Thesis. *Cognitive Science*, *32*(6), 939–984.

van Rooij, I., Blokpoel, M., Kwisthout, J., & Wareham, T. (2019). *Cognition and intractability: a guide to classical and parameterized complexity analysis*. Cambridge University Press.

van Rooij, I., Wright, C. D., & Wareham, T. (2012). Intractability and the use of heuristics in psychological explanations. *Synthese*, *187*(2), 471–487.

van Rooij, I. (2012). Self-Organization Takes Time Too: Topics in Cognitive Science. *Topics in Cognitive Science*, *4*(1), 63–71.

van Rooij, I., Wright, C. D., Kwisthout, J., & Wareham, T. (2018). Rational analysis, intractability, and the prospects of 'as if'-explanations. *Synthese*, *195*(2), 491–510.

Villalobos, M. & Dewhurst, J. (2017). Why post-cognitivism does not (necessarily) entail anti-computationalism. *Adaptive Behavior*, *25*(3), 117–128.

Villalobos, M. & Dewhurst, J. (2018). Enactive autonomy in computational systems. *Synthese*, *195*(5), 1891–1908.

Wareham, T., Kwisthout, J., Haselager, P., & van Rooij, I. (2011). Ignorance is bliss: a complexity perspective on adapting reactive architectures. In *Proceedings of the first joint ieee international conference on development and learning and on epigenetic robotics* (Vol. 2, pp. 1–5).

Xu, Z. B., Hu, G. Q., & Kwong, C. P. (1996). Asymmetric hopfield-type networks: theory and applications. *Neural Networks*, *9*(3), 483–501.