Synthese (2012) 187:471–487
DOI 10.1007/s11229-010-9847-7

Intractability and the use of heuristics in psychological explanations

Iris van Rooij · Cory D. Wright · Todd Wareham

Received: 22 January 2010 / Accepted: 25 October 2010 / Published online: 11 November 2010

Abstract Many cognitive scientists, having discovered that some computational-level characterization f of a cognitive capacity ϕ is intractable, invoke heuristics as algorithmic-level explanations of how cognizers compute f. We argue that such explanations are actually dysfunctional, and rebut five possible objections. We then propose computational-level theory revision as a principled and workable alternative.

Keywords Psychological explanation · Computational-level theory · Computational complexity · Intractability · Heuristics · NP-hard · Algorithm · Approximation

1 Introduction

Crucial to the progress of cognitive science and psychology are not only good functional characterizations of a given cognitive capacity ϕ, such as, e.g., visual perception, analogical reasoning, or judgments of grammaticality, but also good explanations of

I. van Rooij (✉)
Donders Institute for Brain, Cognition, and Behaviour, Radboud University Nijmegen, Nijmegen, The Netherlands
e-mail: i.vanrooij@donders.ru.nl

C. D. Wright
Department of Philosophy, California State University Long Beach, Long Beach, CA, USA
e-mail: cdwright@csulb.edu

T. Wareham
Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada
e-mail: harold@mun.ca

how those functions are actually computed.[1] The basic framework for telling such stories, developed in part by Marr (1982), rests on the widely-accepted distinction between computational- and algorithmic-level explanations. A computational-level explanation of some capacity ϕ is a well-defined input/output function $f : I \rightarrow O$, in which inputs $i \in I$ are mapped to outputs $f(i) \in O$.[2] An algorithmic-level explanation of ϕ is a precise characterization of one or more specific algorithms $A_1, A_2, \ldots, A_n$ that a cognitive system may be running whenever it computes (or "solves") f. Each such algorithm $A \in \{A_1, A_2, \ldots, A_n\}$ is a possible explanation of how f, as a model of ϕ, is computed.

It is well known that not every function f is computable (Turing 1936). Moreover, many functions that are computable in principle are not computable in practice (Garey and Johnson 1979). This wrinkle introduces the question of how then to explain the computation of intractable functions. This is particularly the case in cognitive science, where it is not uncommon for theorists to postulate or otherwise discover that the functional characterization of some cognitive capacity ϕ is computationally intractable—i.e., characterized by some f that is too hard to compute exactly with a practical or psychologically realistic amount of computational resources.[3]

For many cognitive scientists (philosophers included), coping with intractability at the computational level—particularly with those f that are NP-hard—requires switching from algorithms that compute exactly to inexact algorithms (or heuristics) that do not.[4] 'When you discover a problem is …hard in principle,' stated Elijah Millgram, 'the sensible approach is to start looking for ways to solve the problem approximately' (2000, p. 87). Likewise, Paul Thagard averred that intractable (NP-hard) problems 'must be handled by computational approximation, not an exhaustive algorithm' (1998, p. 55). Such recommendations are in line with the conventional wisdom in computer

[1] We take cognitive capacities $\phi_1, \phi_2, \ldots, \phi_n$ to be analyzable in terms of composite activities that can be decomposed into hierarchical systems of organized component parts and operations. Consequently, we assume that psychological explanations of these capacities are typically a subclass of mechanistic explanations (note that it does not follow that all psychological explanation just is mechanistic explanation (see Piccinini 2004; Wright and Bechtel 2007; Bechtel and Wright 2009)). We will also assume that norms of descriptive adequacy and completeness dictate that the functional characterization of a given ϕ requires an explanation of how it is computed; but see Piccinini (2010) for an argument that functionalism and computationalism are disjoint theses.

[2] In the theory of computation, functions are often also referred to as '(computational) problems' (cf. Marr's idea that a computational-level theory specifies 'the nature of the problem being solved' (Marr 1982, p. 27; see also Marr and Poggio 1977). The terms 'problem' and 'function' may invite different perspectives: whereas 'problem' may be more prescriptively connotative of an input–output mapping that is to be realized (i.e., a problem is to be solved), the term 'function' may be more descriptively connotative of an input–output that is being realized (i.e., a function is computed). Yet, from a mathematical perspective, these two terms co-refer, viz., to the mathematical object of an input–output mapping. In the context of this paper, 'problem' will be used to mean a computational problem, a function, or an input–output mapping—not only in accordance with how Marr intended the term, but, more importantly, in accordance with how our interlocutors use the term (see quotes in §1).

[3] The problem of intractable functional characterizations of cognitive capacities seems to span the whole gamut of cognitive domains; examples can be found in vision (Tsotsos 1990), reasoning (Levesque 1988), planning (Bylander 1994), analogy derivation (Veale and Keane 1997), decision-making (Gigerenzer and Goldstein 1996), and natural language processing (Barton et al. 1987) to note a few.

[4] Here, we accept that all NP-hard functions are computationally intractable. See van Rooij (2008) for qualification of this claim.

science that, when faced with NP-hard problems, 'the search for an efficient, exact algorithm should be accorded low priority [and it is] more appropriate to concentrate on other, less ambitious, approaches' (Garey and Johnson 1979, p. 3). Many cognitive scientists seem to accept this wisdom as applying also to psychological explanation, possibly reasoning as follows: If inexact algorithms are the only tractable algorithms when facing intractability, they must be acceptable explanations of how intractable functions are computed.

The presumed acceptability of pursuing inexact algorithms in the context of psychological explanation is illustrated by quotes from leading researchers in cognitive science. For example, when observing that Dedre Gentner's (1983) original characterization of analogical mapping as a graph matching problem is NP-hard, Markman and Gentner saw inexact (in this case, sub-optimal) algorithms as the only acceptable explanatory approach:

> Graph matching is known to be in the class of NP-hard problems, meaning that the running time needed for any algorithm that is guaranteed to find the best match increases as an exponential function of the size of the domains being compared. Thus, any psychologically plausible process for finding analogical correspondences must either restrict itself to trivial problems (an unacceptable course) or simplify the solution process (at the risk of finding sub-optimal matches). (2000, p. 507)

In a similar vein, Chater et al. point to the possibility of inexact algorithms as a way of coping with the intractability of the currently popular probabilistic (or Bayesian) cognitive models:

> …we may take models such as stochastic grammars for language or vision, or Bayesian networks, as candidate hypotheses about cognitive representation. Yet, when scaled-up to real-world problems, full Bayesian computations are intractable …From this perspective, the fields of machine learning, artificial intelligence, statistics, informational theory and control theory can be viewed as rich sources of hypotheses concerning tractable, approximate algorithms that might underlie probabilistic cognition. (2006, p. 290)

Lastly, Thagard, when confronted with the fact that 'Abduction characterized in terms of coherence is …intractable', is comforted by the idea that 'efficient approximation algorithms exist' and 'the possibility that heuristic search …[can] improve the computability of abductions' (Thagard and Shelley 1997, p. 426). With co-author Karsten Verbeurgt, Thagard expressed a similar optimism:

> Coherence problems are inherently intractable computationally, in the sense that, under widely-held assumptions of computational complexity theory, there are no efficient (polynomial-time) procedures for solving them. There exist, however, several effective approximation algorithms for maximizing coherence problems, including one using connectionist (neural network) techniques. (1998, p. 2)

It is not difficult to empathize with theorists who appeal to heuristics (or otherwise inexact algorithms) as algorithmic-level explanations of how some intractable f is

computed, given that no exact polynomial-time algorithm A could exist for doing so.[5] Nevertheless, such coping strategies are (fundamentally) misguided. In this paper, we argue that appeals to one or more tractable heuristics H cannot explain how cognizers actually compute an intractable function f. The suggestion otherwise, upon which appeals to heuristics are motivated, leaves these kinds of psychological explanations dysfunctional.[6] We end the paper with a positive suggestion about properly refining this strategy so that the appeal to heuristics can abide by a normative constraint on good psychological explanations.

2 The wrong way to cope with intractability

Most generally, a procedure A is said to be an *algorithm* just in case A is a procedure with a finite description, and, for any given input to process, A halts in a finite number of steps, possibly producing some output.[7] Useful for cognitive science is also a more circumscribed concept of algorithms—one that explicitly relates them to the particular function (or problem) being computed by them. Accordingly, let A be an *algorithm for computing function* $f : I \rightarrow O$ iff A is an algorithm, and, for any input $i \in I$, the algorithm A produces as output $A(i) = f(i)$. To emphasize that A computes *exactly* f for *all* inputs in its domain, we sometimes also refer to A as an *exact algorithm*.

If a function f is NP-hard, there is no algorithm A for computing f that runs in a time which is bounded by some polynomial function of the size of the input (e.g., the running time of the algorithm is upper bounded by $|i|^a$, for each $i \in I$ with $|i| > b$, where a and b are constants). Because nonpolynomial-time algorithms require an excessive amount of time for their completion, even for intermediate input sizes, they are considered to be computationally intractable. Consequently, computationally intractable algorithms are unfeasible algorithmic-level explanations of how subjects compute the functions characterizing their cognitive capacities.[8]

This consequence raises the question of whether the computation of some cognitive capacity ϕ, characterized by intractable function f, is simply inexplicable if only polynomial-time algorithms count as legitimate algorithmic-level explanations. Currently, a popular view among philosophers and cognitive scientists is the contrarian one: good psychological explanations are still possible, intractability notwithstanding. As the earlier quotes from cognitive scientists suggest, one need only posit a polynomial-time heuristic H for f as an algorithmic-level explanation of how it is computed.

[5] This is, of course, modulo the assumption that P $\neq$ NP. This assumption remains unproven, but is nevertheless widely conjectured by mathematicians and computer scientists to be true on both intuitive and empirical grounds (Garey and Johnson 1979; Fortnow 2009). As do our intended interlocutors, we will likewise here assume that the P $\neq$ NP conjecture is indeed true.

[6] A preliminary version of the argument has appeared in van Rooij (2008). The version herein is significantly expanded and strengthened and also rebuts several additional objections.

[7] Along with terms like 'function' and 'computation', 'algorithm' has been used in various senses; unfortunately, the result has often been confusion rather than gains in expressive power (Piccinini 2004, 2010).

[8] See, e.g., Frixione (2001).

Following convention in computer science (see Ausiello et al. 1999), let some procedure H be a *heuristic* just in case the following two conditions obtain. Firstly, H is known to not compute f exactly (i.e., it will output something different from $f(i)$ for at least some $i \in I$). Secondly, some relationship—weaker than equality—obtains between f and the particular function f_H computed exactly by H. (Note that it may be conjectured that $f(i)$ and $f_H(i)$ are the same for many inputs i, or it may be conjectured that the difference between $f(i)$ and $f_H(i)$ is small for many inputs $i \in I$.)

There are two fundamental problems with this view, popular as it is. Not only does the appeal to heuristics introduce an inconsistency between the computational- and algorithmic-level explanations, but it also leaves the computation of f unexplained for any and all $i \in I$ with $f(i) \neq f_H(i)$.

Firstly, to demonstrate the inconsistency, let H be a heuristic for f, and let f_H be the function computed by H. From the definition of heuristics, it follows that there exists a nonempty set of inputs $I' \subseteq I$, such that $f_H(i) \neq f(i)$ for all $i \in I'$. Suppose further that a cognitive scientist—as represented by the aforementioned—maintains that $f(i)$ is a computational-level explanation of some ϕ and that H is its associated algorithmic-level explanation. Now, what should she predict as the observable outcome of presenting a subject directly with some input $i \in I'$ as produced by the capacity of interest? According to the computational-level explanation, she should predict the output will be $f(i)$; but, according to the algorithmic-level explanation, the predicted output should be $f_H(i)$. But since $f_H(i) \neq f(i)$, the two predictions are inconsistent competitors: the computational-level explanation is confirmed by the data only if the algorithmic-level theory is disconfirmed, and the algorithmic-level theory is confirmed by the data only if the computational-level explanation is disconfirmed.

Besides the inconsistency between the computational- and algorithmic-level explanations that is introduced by maintaining the heuristic H as an algorithmic-level explanation, H also fails as an explanation of how the computation of f is achieved by cognizers. This follows simply from the fact that H does not compute f, but instead computes f_H, which is a distinct computational problem from f. After all, there exist (possibly many) $i \in I$ such that $f(i) \neq f_H(i)$, and for all those i, the heuristic H cannot explain how $f(i)$ is computed.

Cognitive scientists who persist in maintaining that H is a legitimate and accurate algorithmic-level explanation certainly can resolve the aforementioned problems of inconsistency and explanatory failure. Perhaps the most obvious resolution would be to revise the computational-level explanation, such that the cognitive capacity characterized is the function f_H instead of f. Although ad hoc, the upshot of this revision is that the function-algorithm pair (f_H, H) turns out to be a viable (that is, internally consistent) psychological explanation of ϕ: after all, H is a tractable and exact algorithm for f_H, and therefore a feasible explanation of how the computational problem f_H is computed. Of course, since the two functions f_H and f differ according to their tractability, this sort of ad hoc revision makes plain that the cognitive capacity actually explained is the ϕ' characterized by f_H, not the ϕ that cognitive scientists initially set out to explain. On the other hand, refinements to descriptions of the target explanandum are often crucial for establishing good psychological explanations, although, prima facie there is at least some difference between redescribing an explanandum

versus describing an altogether different explanandum. Hence, whether one takes this sort of revision to be tantamount to an admission that the target explanandum has been mischaracterized, or merely just a stage in refining it, we leave up to the reader.

3 Objections and replies

Having demonstrated why a currently popular view among cognitive scientists is misguided, in this section we reply to five common objections, both implicit and explicit in the work of cognitive scientists and philosophers such as John R. Anderson, Peter Carruthers, Nick Chater, Kenneth Forbus, Dedre Gentner, Gerd Gigerenzer, Richard Samuels, and Paul Thagard.

Before doing so, however, it is important to forestall a series of related and superficial complaints that are based primarily on misreading or otherwise misunderstanding the argument of this paper. First, worries about teleology and normativity are orthogonal. For to say that computational-level explanations of ϕ are well-defined input/output mappings is in no way to abnegate the normative or teleological dimensions of computational or functional characterizations, and so in no way to say that non-normative specifications of input/output mapping functions are sufficient for good computational-level explanations, or that those explanations are reducible to those functions. Consequently, the observation that the exact computation of functions characterizing cognitive capacities is a constraint on good computational explanations does not prohibit theories of psychological explanation from proposing additional constraints on good computational-level explanations (such as the constraint dictating that those functions' normative or teleological dimensions may be important to account for). Hence, the argument of this paper is independent of claims about the relationship between normative stories and computational-level explanations because such normative stories are orthogonal to the assessment of the tractability or intractability of the input/output mapping function that is being postulated. Tractability is a property of functions, not normative stories.

This misreading is also sometimes rephrased in terms of the distinction between 'why'-explanations and 'how-' or 'how-possibly' explanations; but, here too, the argument of this paper neither denies, nor depends on denying the importance and utility of purely computational-level 'why'-explanations. Relatedly, neither does it involve the assertion that 'how-' or 'how-possibly' mechanistic explanations about the computation of those functions are the only legitimate kinds of psychological explanations.

It might also be complained that the argument of this paper depends on a skewed view of the research of actual practitioners in the psychological and cognitive sciences, and, in particular, on the implausible claim that those researchers are primarily concerned with showing in the abstract how to compute tractable functions for a given cognitive capacity. As an objection, this complaint also misses the mark. For starters, it badly conflates the argument's prescriptive conclusion about there being a normative constraint on good psychological explanation with a descriptive assumption about the current practice of research in the cognitive sciences. Moreover, even if there were no conflation, the argument of this paper involves no such assumption or pretense that researchers are primarily concerned with computational-level modeling or with

abstractly characterizing all and only the tractable functions. And even if the argument did so depend, it is sufficiently clear that practitioners take both computational- and algorithmic-level modeling to be useful and important, and are, in fact, concerned with how their research is constrained by issues of complexity, tractability, and norms of explanation.

With these various misreadings out of the way, we can now turn to the five objections to our claim that heuristics are inapposite explanations of how intractable functions are computed. First, consider *the objection from optimism* [Forbus, Gentner, Samuels, Thagard]:[9] H may be a heuristic that computes f inexactly, but it does process the correct output for most, or even all, of the inputs that it was tested on—namely, I_χ. Therefore, since H performed so well on our trials, we believe it is a good algorithmic-level explanation for f.

This objection is simply an ignoratio elenchus. Regardless of whether $f(i) = f_H(i)$ for all $i \in I_\chi$, where I_χ is some—possibly large—set of tested inputs, it is known that there also exist inputs $i \in I'$, with $I' \bigcap I_\chi = \emptyset$ such that $f_H(i) \neq f(i)$. Otherwise, H would be an exact algorithm for f_H, which would contradict both the heuristic status of H and the NP-hardness of f. For all those $i \in I'$ for which $f(i) \neq f_H(i)$, there still exists an inconsistency between the computational- and algorithmic-level explanations that this objection fails to address. Again, H fails to explain how f is actually computed; at worst, it fails in general, and at best it fails for all those $i \in I'$ for which $f(i) \neq f_H(i)$.

A successor to the objection from optimism is *the objection from persistent optimism* [Forbus, Gentner, Samuels]:[10]. H may return the wrong output for some inputs to f, but there appear to be only a small number of these inputs. Hence, H can still be a good algorithmic-level explanation for f.

As it were, this condition cannot be met, even granting that a (very) small number of errors may render an explanation (more-or-less) acceptable. This result follows from an observation by Schöning (1990, p. 232) that the existence of a polynomial-time algorithm H for a function f which misbehaves on a finite set of inputs (by, e.g., giving the wrong outputs or running in super-polynomial time on those inputs) implies the existence of an algorithm H' which computes f correctly on all inputs in polynomial time. This algorithm H' first checks if a given input is in a table of misbehaving inputs. If so, the answer associated with that input in the table is returned; otherwise, the answer is computed by running H on that input. As the set of misbehaving inputs is finite, the lookup table preprocessing can be done in constant time, allowing H' to run in polynomial time. This implies the following lemma, which in turn addresses the objection as phrased.

Lemma 1 *If $f : I \rightarrow O$ is an NP-hard function, then there cannot exist a polynomial-time heuristic H for f that returns $f(i)$ for all $i \in (I - I')$ where $I' \subset I$ is finite (assuming $P \neq NP$; see footnote 5).*

[9] See, e.g., Forbus and Oblinger (1990, pp. 5–6); Genter and Markman (1995, p. 133); Samuels (2005, p. 109, footnote 2); Thagard and Verbeurgt (1998, p. 33).

[10] See, e.g., Forbus and Oblinger (1990, pp. 5–6); Genter and Markman (1995, p. 133); Samuels (2005, p. 109, footnote 2).

Third is *the objection from implicit revision* [Anderson, Gigerenzer]:[11] H may be a heuristic for f, but it does process the correct output for all (psychologically) relevant inputs, and it was these inputs that were always intended. Hence, H can still be a good algorithmic-level explanation for f.

This objection suggests two things, each of which is possible in principle but in need of verification; and yet, once verified, the objection becomes vacuous. Firstly, the objection states that not all possible inputs to $f : I \rightarrow O$ are relevant for explaining ϕ but only a proper subset $I' \subset I$ is. This raises the question of what I' is (i.e., the set is in need of characterization if H is to explain how f is computed for inputs restricted to I' and why only inputs in I' are relevant (possibly this is because only $i \in I'$ are believed to occur in the real world whenever ϕ is manifest)). Of course, not every ad hoc revision that appeals to the 'relevant', 'plausible', 'interesting', etc. subset of inputs is inappropriate. Yet, the danger is that the theorist is playing a shell game; for such appeals are only as acceptable as the warrant for targeting precisely that subset as opposed to any other.

Either way, grant for the sake of argument that $I' \subset I$ can both be characterized and (psychologically) motivated, and now let us consider the function $f' : I' \rightarrow O$, which denotes the function f restricted to inputs $I' \subset I$ (i.e., $f(i) = f'(i)$ for all $i \in I'$ and $f'(i)$ is undefined otherwise). The objection states that it is f', rather than f, that properly characterizes the cognitive capacity ϕ as is to be explained. Moreover, since H is claimed to compute f' correctly for all $i \in I'$, and H is a tractable algorithm, the objection entails that f' is tractable. This raises the question of whether 'f' is tractable' is true. Yet, if we grant that f' is indeed tractable and that H computes it, then plainly the objector has not actually maintained the intractable f as a psychological explanation of ϕ and H as a heuristic for explaining how f is computed. Rather, she is merely maintaining tractable f', as a characterization of ϕ and the exact algorithm H is the algorithmic-level explanation of how f' is computed. The restriction is such that the target subset is being now claimed to be tractable; yet, intractability was what motivated the shift to an inexact algorithm in the first place.

A fourth objection is *the objection from approximation* [Chater, Thagard]:[12] H may be a heuristic (i.e. nonexact) algorithm for f, but its output $f_H(i)$ may approximate the correct output $f(i)$ for all inputs i. Hence, H appears to be a good algorithmic-level explanation for f.

Much of the force of this objection depends on what is meant by 'approximate', but unfortunately that meaning is unclear. We suspect a variety of its meanings may be subsumed by the following.[13] Firstly, it is implicitly understood that approximate outputs for $f(i)$ are drawn from a set of candidate outputs, which is a superset including $f(i)$. We denote the candidate-output set function by 'f_c'. To illustrate, consider the

[11] Although we are not aware of any cognitive scientist who explicitly defends this position, we think it is a natural objection that in our view improves upon the two previous ones. We see the idea that tractability of cognitive processes can in part be explained by exploiting ecological constraints on the inputs as being implicit in the work of both Anderson (1990, p. 29) and Gigerenzer (2000, p. 736).

[12] See, e.g., Chater et al. (2006, p. 290); Thagard and Verbeurgt (1998, p. 2).

[13] For conceptions of approximation relevant to cognitive science, see e.g., Ausiello et al. (1999) and Kieseppä (1996).

well-known NP-hard computational-level explanation of coherence by Paul Thagard. This theory postulates a function that takes as input a network of pair-wise positive and negative constraints, and gives as output a truth assignment on the nodes of the network that satisfies a maximum number of constraints. For this function the candidate-output set is naturally defined as the set of all possible truth assignments. Secondly, it is implicitly assumed that there is some distance function $d(o, o')$ measuring the distance between $o, o' \in f_c(i)$ for any given $i \in I$. For example, a possible measure of the distance between two truth assignments is the number of values on which they differ, also known as the Hamming distance (e.g., distance between TTTFFF and TFTFTF would be 2 in this case). Finally, given a candidate-output set function f_c and a distance function d, we say that outputs of a heuristic H ϵ-approximate the outputs of f iff $f_H(i) \in f_c$ and $d(f(i), f_H(i)) < \epsilon$, for some (preferably small) constant ϵ for all $i \in I$.

We prove that it is not possible for a polynomial-time algorithm, like H, to approximate a significant set of NP-hard functions in this sense. This set of functions is defined by the property *neighborhood-searchability*, a property known to hold for many (if not all) NP-hard functions—including Thagard's constraint satisfaction theory of coherence—for natural choices of f_c and d. Without loss of generality, in the following we assume for simplicity that ϵ is a non-negative integer.

Definition 1 A function f is neighborhood-searchable relative to a distance function d if, for any input $i \in I, o \in f_c(i)$, and constant ϵ both of the following conditions are met:

1. when given o, it is possible to generate all outputs $o' \in f_c(i)$ with $d(o, o') \leq \epsilon$ in time polynomial in the size $|i|$ of input i, and
2. when given all the outputs $o' \in f_c(i)$, with $d(o, o') \leq \epsilon$, it is possible to recognize which of the outputs in this ϵ-neighborhood is $f(i)$ in time polynomial in the size $|i|$ of input i.

To see that Thagard's constraint satisfaction theory of coherence satisfies condition 1, for f_c and d as proposed above, consider that for any given truth assignment $o \in f_c(i)$ of length n, there are at most n possible truth assignments that are distance 1 from o, at most $\binom{n}{2}$ that are at distance 2 from o, …and at most $\binom{n}{\epsilon}$ that are at distance ϵ from o. In total, the set of all outputs $o' \in f_c(i)$ with $d(o, o') \leq \epsilon$ contains no more than ϵn^{ϵ} possible truth assignments. Generating all these truth assignments, can be done in no more than $\epsilon n^{\epsilon+1}$ computational steps—which is polynomial time since $n \leq |i|$ and ϵ is a constant—simply by writing out each n-length truth assignment one by one.

We note that condition 2 is met, by definition, by all NP-hard functions that are in NP (i.e., all NP-complete functions). To see that Thagard's theory (which is not in NP) satisfies condition 2, observe that checking the number of constraints that are satisfied by a given truth assignment can be done in a time that is linear in the number of constraints in the network: for every positive constraint, simply check if the two nodes that it connects have the same truth value (in which case it is satisfied), and for every negative constraint, check if the two nodes that it connects have the opposite truth

value (in which case it is satisfied). If we know that the ϵ-neighborhood of o contains $f(i)$, we can thus recognize $f(i)$ in polynomial-time, by verifying that it satisfies the maximum number of constraints of all the truth assignments in $o's$ ϵ-neighborhood. To verify this, we need at worst to compute the number of constraints satisfied by all other o' in the ϵ-neighborhood of o and verify that the number is lower than for truth assignment $f(i)$.

Lemma 2 *If f is an NP-hard function that is neighborhood-searchable for a distance function d, then no polynomial-time algorithm can ϵ-approximate f (assuming P $\neq$ NP; see footnote 5).*

The proof is by reductio ad absurdum. Assume P $\neq$ NP and assume that there is a polynomial-time algorithm H that ϵ-approximates. Then we can run H on i to create output o such that $d(o, f(i)) \leq \epsilon$. Consider a second algorithm A that takes as input the output of H, and determines for all $o' \in f_c(i)$ with $d(o, o') \leq \epsilon$ whether $o' = f(i)$. Because we know that $d(o, f(i)) \leq \epsilon$, at least one such $o' = f(i)$, and hence H and A together afford the exact computation of $f(i)$. Since H is a polynomial time algorithm and, by clause 2 of Definition 1, A is a polynomial time algorithm, $f(i)$ can be computed in polynomial time. However, as f is an NP-hard function, this implies that P = NP, and is thus a contradiction.

Corollary 1 *If f is an NP-hard function that is neighborhood- searchable relative to a distance function d, and H is a polynomial-time algorithm, then there exist inputs i of f such that $d(f(i), f_H(i)) > \epsilon$ for any (possibly large) constant ϵ (assuming P $\neq$ NP; see footnote 5).*

The above establishes that, under reasonable operationalizations of "approximation", intractable functions are not tractably approximable. This counters the objection as phrased. To avoid any misreading of our argument, we emphasize that we do not wish to suggest that algorithmic-level explanations cannot, or should not ever be, approximations of computational-level theories. We merely argue that, at best, they can be approximations only when the to-be-approximated computational-level theory is tractable.[14]

Up to this point, all objections and replies have dealt with attempts to have single-heuristic explanations of tractably computing an intractable function. We have

[14] The point is worth emphasizing, as it addresses the common misconception that computational-level intractability is acceptable because explanations at that level are either false-but-useful idealizations, or are otherwise braced by the competence/performance distinction. We do not deny that computational-level theories can serve as competence theories or that they can be idealizations. Yet, tractability is a minimal requirement on computational feasibility of any computation running on any machine, and therefore best seen as a fundamental constraint on what competences can exist rather than just a performance limitation (for a full argument see e.g. Frixione 2001). We also do not deny the usefulness of a competence / performance disctinction and recognize that certain competences can at best be approximated when implemented in machines with performance-level limitations. However, as our response to the 'objection from approximation' shows, such implementations are impossible for computational-level theories that construe competence in terms of intractable functions. For a computational-level idealization to be a proper idealization (and not an 'overidealization') there should minimally exist algorithms that can tractably approximate the competence postulated by that computational-level theory; otherwise the idealization has failed to capture the nature of the competence and instead has mischaracterized it (cf. Franks 1995).

argued that none such attempts can work. Some may still counter that a combination of multiple heuristics (of any mix of the previously discussed types) may suffice to tractably compute such a function. In that event, each heuristic in the set, although by itself insufficient, can be rightly said to be a partial explanation of how the function is tractably computed despite its intractability.

This is the intuition behind the fifth and final objection, *the objection from partial explanation* [Carruthers, Gigerenzer]:[15] Although H is a heuristic (i.e. inexact algorithm) for f, and admittedly cannot explain *by itself* how f is computed, it could be a member of a finite set of heuristics $\mathcal{H} = \{H_1, \ldots, H_n\}$ that *together* can explain the computation of f. If indeed $\mathcal{H}$ explains the computation of f and $H \in \mathcal{H}$, then one could say that H is a (possibly necessary) *part* of an explanation of the intractable function f.

In response to the objection, we will show that no finite set of tractable heuristics can be used to create a tractable algorithm for an intractable function. The proof is by reductio ad absurdum. Assume P $\neq$ NP, and let $f : I \rightarrow O$ be some intractable function. Further, assume that there is a finite set of polynomial-time heuristics $\mathcal{H} = \{H_1, \ldots, H_k\}$ for some constant $n > 0$, such that, for each input $i \in I$, there is at least one heuristic that computes $f(i)$. Observe that as k is a constant, we can run all heuristics in $\mathcal{H}$ on any $i \in I$ in polynomial time. Call this procedure A, and let $A(i)$ be the set of outputs produced by A on $i \in I$. Note that among the $|\mathcal{H}| = k$ outputs in the set produced by A, there is at least one correct output $f(i)$. This does not mean that A has computed $f(i)$ yet, because in order to do that we still need a procedure to select from $A(i)$ the output corresponding to $f(i)$. If there were to exist a polynomial-time procedure B that selects a correct output from $A(i)$, observe that A and B can be combined, simply by running the algorithms in sequence, to produce a polynomial-time algorithm for solving f. However, as f is an NP-hard function, this implies that P = NP, and is thus a contradiction.

Lemma 3 *If $f : I \rightarrow O$ is an NP-hard function which has an associated finite set of polynomial-time heuristics $\mathcal{H}$ such that for each $i \in I$ there is at least one heuristic in $\mathcal{H}$ that computes $f(i)$, then there cannot exist a polynomial-time procedure that uses $\mathcal{H}$ to solve f (assuming P $\neq$ NP; see footnote 5).*

Note that this lemma implies that the heuristic-set approach is doubly problematic—not only are we not guaranteed that our NP-hard function f has a requisite set $\mathcal{H}$, but even if it does, we cannot use such an $\mathcal{H}$ to solve f in polynomial time.

4 The right way to cope with intractability

We have explained why postulating heuristics as (inexact) algorithmic-level explanations is the wrong way of dealing with the intractability of one's computational-level

[15] Even though strictly speaking neither Gigerenzer nor Carruthers have formulated a formal computational-level theory of known intractability, they often postulate that minds ensure tractability of cognitive computations by invoking a set of tractable modules (Carruthers 2005) or a set of tractable heuristics (Gigerenzer 2008; Todd and Gigerenzer 2000).

theory. The only way to ensure consistency between algorithmic- and computational-level theories in cognitive science, given the constraint that algorithmic-level theories can posit only tractable (polynomial-time) algorithms, is that computational-level theories posit functions that are tractably computable (see also Frixione 2001; van Rooij 2008). This means that when a cognitive scientist attempting to characterize some cognitive capacity ϕ accidentally posits an intractable function f, she must conclude that either she has mischaracterized the capacity ϕ or that ϕ resists any computational explanation. As the latter option is near blasphemy for cognitive scientists in the business of computational explanation, it seems that only the former option remains open.

Having concluded that cognitive capacity ϕ has indeed been mischaracterized by some function f, again two options seem open to the cognitive scientist: she may conjecture that ϕ is still a basic cognitive capacity, but it should be characterized by a different function f', or ϕ is not a basic cognitive capacity, yet possibly some other capacity ϕ' is which can be characterized by function f'. In the following, we will not distinguish between the two cases, as either will lead to the same revised computational-level theory—namely f'. Moreover, since cognitive capacities are only intuitively and informally understood prior to their characterization as a well-defined computational-level theory, often it is difficult or impossible to unequivocally distinguish between the intuitively understood ϕ and ϕ'.

A classic example of the revision approach to dealing with intractability can be found in the work of philosopher Christopher Cherniak (see e.g., his 1986 book *Minimal Rationality*). Cherniak points out that the 'ideal rationality' condition—i.e., 'make all and only deductively sound inferences from [one's] belief set that are apparently appropriate' (p. 77)—yields a mischaracterization of the human capacity for belief fixation. Ideal rationality would imply that humans can use their capacity for belief fixation to solve problems in logic that are proven to be intractable (or even worse, uncomputable). Therefore, so Cherniak argues, humans cannot be in possession of such a capacity.

As an alternative to the ideal rationality assumption, Cherniak proposes that the human capacity for belief fixation meets only what he calls the 'minimal rationality' condition—i.e., 'make some, but not necessarily all, sound inferences from [one's] belief set that are apparently appropriate' (p. 81). Cherniak admits that the 'minimal rationality' condition can at best be a necessary condition for (human) rationality, and hence his characterization of the human capacity for belief fixation is incomplete. Nevertheless, it is clear that Cherniak pursued a revision of the computational-level theory of human belief fixation in order to evade the problem that an existing theory at the time—ideal rationality—characterized human belief fixation by an intractable function.

A similar move has been made by decision-making researcher Gerd Gigerenzer (2008). Like Cherniak, Gigerenzer rejects the ideal rationality attributed to humans by classic economic theory, though his interest is more in decision-making than in belief fixation. Gigerenzer replaces the ideal rationality model of decision-making (i.e., maximizing expected utility) by what he calls 'ecological rationality' (i.e., making decisions that are 'good enough'). Gigerenzer fails so far to present a complete characterization of the purported human capacity for ecological rationality, but in any

event, it is clear that he pursues a revision of the ideal rationality model of human decision-making in an attempt to avoid its intractability.[16]

We consider the computational-level theory revisions alluded to by researchers such as Cherniak and Gigerenzer to be the right way of *trying* to cope with intractability of computational-level theories. Whether or not such revisions are effective—i.e., that they yield tractable functions—remains to be seen. This is important to emphasize, as the need for verifying the effectiveness of a revision seems to be underappreciated in the literature and sometimes revisions are claimed to be tractable without proof. For example, both Cherniak and Gigerenzer, in more than one place, suggest that the intractability of ideal rationalization is due to the assumption of 'perfection' in deductive inference or 'optimization' in decision making, and that, instead, making inferences that are 'better than random guesses' and choices that are 'good enough' are tractable. It is known from computational complexity theory, however, that perfection or optimization are neither necessary nor sufficient conditions for intractability (cf. Gigerenzer 2008, pp. 21–22, for example).

To illustrate the above considerations[17], consider the logic problem called SAT (for 'Satisfiability') and a closely related optimization problem called MAX-SAT. The problem SAT is defined as a function that takes as input a conjunction of disjunctions of (possibly negated) variables (e.g., $(\neg a \vee b) \wedge (b \vee c \vee \neg d \vee e) \wedge \cdots \wedge (u \vee v \vee \neg w)$) and gives output 'Yes' if there exists a possible truth assignment to the variables such that the conjunction is true, and 'No' otherwise. The problem MAX-SAT is defined as a function that takes the same inputs as SAT but instead outputs a truth assignment that ensures that a maximum number of clauses of disjunctions in the given conjunction are true.[18]

Both SAT and MAX-SAT are known to be intractable (NP-hard) (Garey and Johnson 1979).[19] This means neither problem can be (part of) a feasible computational-level theory of human reasoning abilities. Let us now revise SAT into guess-SAT, where the solution of guess-SAT requires only that the output ('Yes' or 'No') is correct with probability > 0.5 (i.e., outperform a random guess, cf. Cherniak 1986, p. 81). Further, let us revise MAX-SAT into good-enough-SAT such that the solution of good-enough-SAT requires only that λ clauses of disjunctions in the input are true, where λ is some aspiration level set by the decision-maker (cf. Gigerenzer 2008, p. 20). Both revisions appear to be easier that the original problem, and for the untrained eye they may appear to be (obviously) tractable. Such intuition would be mistaken, however. To the

[16] Gigerenzer has proposed that the type of 'ecological rationality' he attributes to humans can be computed by a toolbox of heuristics that are fast and frugal, where each heuristic works well for different decision-making environments. Neither this suggestion nor the set of concrete models for several heuristics in the toolbox, however, yet constitutes a computational-level characterization of the human capacity for ecologically rational decision-making.

[17] The same underlying ideas can be found in van Rooij (2008), which are illustrated by way of the coherence theory of Thagard (2000).

[18] In the event that this number equals the total number of clauses in a given input, solving MAX-SAT effectively amounts to also solving SAT.

[19] SAT is famous for being the first problem to be proven NP-hard. The result is due to Cook (1971).

best of current (mathematical) knowledge, guess-SAT and good-enough-SAT are both intractable (Garey and Johnson 1979; Wigderson 2007).[20]

This illustration, so far, establishes that weakening conditions of perfection or optimization are not sufficient for rendering computational-level theories tractable. Does this mean that no revision can be tractable? Surely not. Many alternative forms of revision can be pursued, instead or in addition. Consider, for example, that we revise SAT to 2-SAT, where 2-SAT takes the same types of inputs as SAT with the constraint that clauses can contain at most 2 variables. Such a revision ensures tractability of the revised problem, as 2-SAT is known to be polynomial-time computable (Garey and Johnson 1979). This example now also illustrates that perfection is not always intractable. An analogous revision for MAX-SAT does not work, as 2-MAX-SAT is known to be still intractable. Yet, if we combine the 'good enough' revision with the 'restricted input domain' revision, then we get the problem 2-good-enough-SAT, which is tractable for relatively small aspiration levels (Mahajan and Raman 1999).[21]

To sum up, theory revision is the right way of dealing with intractability at the computational-level and several options for revision strategies seem open to cognitive scientists. Cognitive scientists are warned, however, that not every computational-level theory revision will yield a tractable function. Moreover, even functions that appear to be intuitively tractable in fact may not be. Verification of tractability, by means of mathematical proof, is indispensable for cognitive scientists that hope to devise tractable computational-level characterizations of cognitive capacities.

5 Conclusion

It is a necessary condition on the provision of good scientific explanation that the phenomena to be explained be, for obvious reasons, appropriately and veridically characterized. In the psychological sciences, explanation of a cognitive capacity ϕ is often construed as computational explanation. Subsequently, it stands to reason that a minimal, normative constraint on the provision of good computational explanations of ϕ is that the function f characterizing ϕ should be computable using a realistic amount of computational resources.

Currently, however, many cognitive capacities are characterized with intractable functions, and so fail to satisfy this normative constraint. An ubiquitous and attractive strategy for addressing that failure—one proposed by many cognitive scientists, philosophers and psychologists included—has been to maintain that an intractable

[20] The intractability of guess-SAT and good-enough-SAT assumes that P $\neq$ NP and that P = BPP (BPP stands for bounded-error probabilistic polynomial-time). These are mathematical conjectures that are unproven to date, but nevertheless believed by the majority of living mathematicians (see, e.g., Garey and Johnson 1979, §2.1, and Wigderson 2007, §5.2). Although one is free to doubt the conjectures, doing so in order to claim tractability of a computational-level theory would seem to pose the burden of proof on the cognitive scientist to present some argument for why we should not trust, in this particular case, the intuitions of the majority of living mathematicians. Moreover, if cognitive scientists were to doubt the P $\neq$ NP conjecture, then SAT and MAX-SAT would be tractable, as would be all other NP-hard problems in NP, and few computational-level theories would be intractable to begin with.

[21] 2-λ-SAT is known to be fixed-parameter tractable (Downey and Fellows 1999), meaning the problem is efficiently solvable if the aspiration level λ is relatively small.

function f intended to characterize some ϕ can be dealt with by positing heuristics as algorithmic-level explanations. As we have shown, however, such heuristics turn out to be inapposite candidates for algorithmic-level explanation; their employment induces a conceptual confusion between attempting to achieve a computation and explaining how a computation is achieved. This reminder has major ramifications for what has previously passed for good computationally-construed psychological explanation.

We have suggested that theorists can abide by the aforementioned normative constraint on good computational explanation by engaging in computational-level theory revision in lieu of positing heuristics. On our view, the initial mischaracterization of some cognitive capacity ϕ by an intractable function f is not an insurmountable problem in itself, much less an unforgivable one. After all, an intractable function may be revised, such that it becomes tractable while still capturing much of the assumptions and thoughts that prompted the initial characterization.

Cognitive scientists should welcome the normative constraint of tractability on their computational-level theorizing, if only because of the in-principle unbounded space of possible characterizations to consider. Without that constraint, the realization that some putative characterization is actually a mischaracterization could only be identified, if at all, by the laborious and time-consuming method of experimental testing.

Unfortunately, there is no definitive protocol for computational-level theory revision that guarantees the transformation of an intractable function into a (psychologically-plausible) tractable one, let alone in a veridical one. (Cognitive scientists should therefore expect to entertain and reject many mischaracterizations before formulating the set of all and only appropriate and veridical ones.) But there are plausible strategies, one of which we end with. Given some ϕ, theorists might devise a tractable heuristic H for the function f that characterizes ϕ, followed by substitution of f by the function computed exactly by H, designated here as f_H. Such a revision would ensure that the algorithmic-level explanation H is not a heuristic, but rather an exact algorithm for the function whose computation is to be explained. The revision would also ensure that the function f_H is tractable. Plainly, the success of this strategy depends, inter alia, on whether the cognitive capacity being characterized, ϕ, can be appropriately and veridically characterized by the substituend, f_H. Answering that question is the hard part, but not so hard as to be harder than any of the other hardest problems in cognitive science. And for that, cognitive scientists can take solace.

References

Anderson, J. R. (1990). *The adaptive character of thought*. Hillsdale, NJ: Lawrence Erlbaum Associates.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., & Protasi, M. (1999). *Complexity and approximation: Combinatorial optimization problems and their approximability properties*. Berlin: Springer.

Barton, G., Berwick, R., & Ristad, E. (1987). *Computational complexity and natural language*. Cambridge, MA: The MIT Press.

Bechtel, W., & Wright, C. (2009). What is psychological explanation. In P. Calvo & J. Symons (Eds.), *Routledge companion to the philosophy of psychology* (pp. 113–130). New York: Routledge.

Bylander, T. (1994). The computational complexity of propositional STRIPS planning. *Artificial Intelligence, 89*(1–2), 165–204.

Carruthers, P. (2005). Simple heuristics meet massive modularity. In P. Carruthers, S. Laurence, & S. Stich (Eds.), *The innate mind. Vol. 2: Culture and cognition* (pp. 181–198). Oxford: Oxford University Press.

Chater, N., Tenenbaum, J. B., & Yuille, A. (2006). Probabilistic models of cognition. *Trends in Cognitive Science, 10*(7), 287–293.

Cherniak, C. (1986). *Minimum rationality*. Cambridge, MA: The MIT Press.

Cook, S. (1971). The complexity of theorem-proving procedures. In *Proceedings of the 3rd annual ACM symposium on the theory of computing*. New York, pp. 151–158.

Downey, R., & Fellows, M. (1999). *Parameterized complexity*. Berlin: Springer.

Forbus, K., & Oblinger, D. (1990). Making *SME* Greedy and Pragmatic. In *Proceedings of the twelfth annual conference of the cognitive science society*. Hillsdale, NJ, pp. 61–68.

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

Franks, B. (1995). On explanation in the cognitive sciences: Competence, idealization, and the failure of the classical cascade. *British Journal of Philosophy of Science, 46*(4), 475–502.

Frixione, M. (2001). Tractable competence. *Minds and Machines, 11*, 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.

Gentner, D. (1983). Structure-mapping: A theoretical framework for analogy. *Cognitive Science, 7*, 155–170.

Gentner, D., & Markman, A. (1995). Similarity is like analogy. In C. Cacciari (Ed.), *Similarity in language, thought, and perception* (pp. 111–147). Brussels: BREPOLS.

Gigerenzer, G. (2008). Why heuristics work. *Perspectives on Psychological Science, 3*, 20–29.

Gigerenzer, G., & Goldstein, D. G. (1996). Reasoning the fast and frugal way: Models of bounded rationality. *Psychological Review, 103*, 650–669.

Kieseppä, I. (1996). *Truthlikeness for multidimensional, quantitative cognitive problems*. Dordrecht: Kluwer.

Levesque, H. J. (1988). Logic and the complexity of reasoning. *Journal of Philosophical Logic, 17*, 355–389.

Mahajan, M., & Raman, V. (1999). Parameterizing above guaranteed values: MaxSat and MaxCut. *Journal of Algorithms, 31*, 335–354.

Markman, A., & Gentner, D. (2000). Structure mapping in the comparison process. *The American Journal of Psychology, 113*, 501–538.

Marr, D. (1982). *Vision: A computational investigation into the human representation and processing visual information*. San Francisco, CA: W. H. Freeman.

Marr, D., & Poggio, T. (1977). From understanding computation to understanding neural circuitry. *Neuroscience Research Progress Bulletin, 15*, 470–488.

Millgram, E. (2000). Coherence: The price of the ticket. *Journal of Philosophy, 97*, 82–93.

Piccinini, G. (2004). Functionalism, computationalism, and mental states. *Studies in the History of Philosophy of Science, 35*, 811–833.

Piccinini, G. (2010). The mind as neural software? Understanding functionalism, computationalism, and computational functionalism. *Philosophy and Phenomenological Research, 81*, 269–311.

Samuels, R. (2005). The complexity of cognition: Tractability arguments for massive modularity. In P. Carruthers, S. Laurence, & S. Stich (Eds.), *The innate mind, Vol. 1: Structure and contents* (pp. 107–121). Oxford: Oxford University Press.

Schöning, U. (1990). Complexity cores and hard problem instances. In T. Asano, T. Ibaraki, H. Imai, & T. Nishizeki (Eds.), *Proceedings of the international symposium on algorithms (SIGAL'90)*. Berlin, pp. 232–240.

Thagard, P. (1998). Computation and the philosophy of science. In T. Bynum & J. Moor (Eds.), *The digital phoenix: How computers are changing philosophy* (pp. 48–61). Oxford: Blackwell.

Thagard, P. (2000). *Coherence in thought and action*. Cambridge, MA: The MIT Press.

Thagard, P., & Shelley, C. (1997). Abductive reasoning: Logic, visual thinking, and coherence. In M.-L. D. Chiara, K. Doets, D. Mundici, & J. van Bentham (Eds.), *Logic and scientific methods* (pp. 412–427). Dordrecht: Kluwer.

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

Todd, P., & Gigerenzer, G. (2000). Précis of simple heuristics that make us smart. *Behavioral and Brain Sciences, 23*, 727–780.

Tsotsos, J. K. (1990). Analyzing vision at the complexity level. *Behavioral and Brain Sciences, 13*, 423–469.

Turing, A. (1936). On computable numbers, with an application to the Entscheidungsproblem. *Proceedings of the London Mathematical Society, 42*, 230–265.

van Rooij, I. (2008). The tractable cognition thesis. *Cognitive Science, 32*, 939–984.

Veale, T., & Keane M. T. (1997). The competence of sub-optimal theories of structure mapping on hard analogies. In *Proceedings of the 15th international joint conference on artificial intelligence (IJCAI'97)*, Vol. 1. pp. 232–237.

Wigderson, A. (2007). P, NP and mathematics—a computational complexity perspective. In: *Proceedings of ICM 2006*, Vol. I. Zurich, pp. 665–712.

Wright, C., & Bechtel, W. (2007). Mechanisms and psychological explanation. In P. Thagard (Ed.), *Handbook of philosophy of psychology and cognitive science* (pp. 31–79). New York: Elsevier.