Intractability and approximation of optimization theories of cognition

Iris van Rooij*

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

Todd Wareham

Department of Computer Science,
Memorial University of Newfoundland
St. John's, NL, Canada A1B 3X5

Abstract

Many computational- or rational-level theories of human cognition suffer from computational intractability: the postulated optimization functions are impossible to compute in a reasonable time by a finite mind/brain, or any other computational mechanism. It has been proposed that such intractable theories can nevertheless have explanatory force if we assume that human cognitive processes somehow *approximate* the optimal function. This raises the question of when a cognitive process can be said to approximate an optimal function. In this paper we distinguish between two notions of approximation, called *value-approximation* and *structure-approximation* respectively, and show that they are not equivalent. Although a mathematical framework for assessing degrees of tractable value-approximability has long been available, no such framework previously existed for structure-approximability. In this paper we present a framework consisting of definitions and proof techniques for assessing degrees of structure-approximability. We illustrate the use of our framework for a particular intractable cognitive theory: i.e., Thagard and Verbeurgt's (1998) Coherence model, known to be equivalent to harmony maximization in Hopfield networks. We discuss implications of our

*Corresponding author: I. van Rooij. *Telephone:* +31 (0)24 3612645. *Fax:* +31 (0)24 3616066. *E-mail address:* I.vanRooij@donders.ru.nl

findings for this class of theories, as well as explain how similar results may be derived for other intractable optimization theories of cognition.

Keywords: computational-level theory, rational explanation, computational complexity, optimization, approximation, coherence, constraint satisfaction, harmony maximization, NP-hard

---

1. Introduction

Cognitive outcomes—such as decisions, judgments, inferences, and percepts—are often explained in terms of processes that optimize one or more objective functions. For example, it has been proposed that visual percepts can be explained as the outcome of visual processes that minimize descriptive complexity (Koffka, 1935; van der Helm, 2006; van der Helm and Leeuwenberg, 1996), that similarity judgments can be explained as the outcome of processes that minimize transformational distance between object representations (Chater and Hahn, 1997; Hahn et al., 2003; Imai, 1977), that object categories can be explained as the outcome of categorization processes that maximize within-category similarity and minimize between-category similarity (Pothos and Chater, 2001, 2002; Rosch, 1973; Rosch and Mervis, 1975), that choices can be explained as the outcome of decision processes that maximize utility (Körding, 2007; Luce and Raiffa, 1956; Trommershäuser et al., 2009), and that beliefs can be explained as the outcome of belief fixation processes that maximize conditional probability (Baker et al., 2009; Chater and Oaksford, 2009; Chater et al., 2006) or, alternatively, maximize explanatory coherence (Schoch, 2000; Thagard, 2000; Thagard and Verbeurgt, 1998).

Many such optimization theories—situated at what Marr (1982) coined the 'computational level' of explanation—one way or another run into the problem of computational intractability, in the sense that postulated optimization functions turn out to be impossible to compute in a reasonable amount of time for finite minds/brains, or any other computing machine. It seems difficult in those cases to maintain that the optimization functions are really explaining human cognition, since it is impossible for humans to compute them.[1] It has been proposed that intractable cognitive theories can

[1]Some cognitive scientists may object that the aim of computational-level theories is not to explain 'how' human minds/brains compute the postulated optimization functions, but rather to explain 'why' human behavior is as it is, i.e., its purpose or rationale. Be that as it

be maintained by assuming that human cognition somehow *approximates* the optimal function (Chater et al., 2003, 2006; Love, 2000; Sanborn et al., 2010; Thagard and Verbeurgt, 1998).This proposal raises two questions:

1. What does it mean for a cognitive outcome to approximate an optimal outcome?
2. How can we assess if approximating an optimal outcome is computationally tractable?

In this paper we address these two questions.

The paper is structured as follows. We start by defining optimization problems (or functions) and explain their role as computational-level theories of human cognition (Section 2). We next explain what is meant by 'computational intractability' in the current context and why it poses a (potential) problem for computational-level theories (Section 3). In Section 4, we consider 'approximation' as a way of coping with computational intractability. We make a distinction between value-approximation (traditionally adopted in computer science, see e.g. Ausiello et al. (1999)) and structure-approximation (a newer concept, introduced by ourselves and others in Hamilton et al. (2007)).[2] We furthermore argue that for coping with the intractability of computational-level theories whose primary explanandum is the *structure* of some cognitive outcome (e.g., the form of a percept, the partition yielded by a categorization etc.) approximability should be

may, for an optimization function to be computationally plausible at all there should minimally exist at least one (albeit it unknown) tractable algorithm for computing it (see also Frixione (2001); Gigerenzer et al. (2008); van Rooij (2008)). Our intended interlocutors recognize this tractability constraint on computational-level theories, as they specifically propose that it can be met by assuming approximation rather than exact computation. Ultimately, computability and rationality are orthogonal aspects of computational-level theories, and assessing the computational plausibility of a computational-level theory can be done independently from assessing how that same theory fares as a rational or purposive explanation.

[2]During the preparation of the final version of this paper, we learned of two previously published conference papers that discuss notions of approximability akin to our own (Kumar and Sivakumar, 1999; Feige et al., 2000). Both of the papers also use the notion of "distance" as a measure of the similarity between optimal and approximate outputs, but the reported work does not formulate a general framework for assessing degrees of structural-approximability as we do in this paper. Moreover, we believe that the proof techniques we use in our framework are simpler and correspondingly more accessible.

taken to mean *structure*-approximability. In Section 5 we present a formal framework for quantifying degrees of structure-approximability of optimization problems. We furthermore describe a set of proof techniques which we illustrate using an existing optimization model of belief fixation. We discuss the implications of our results, as well as the general applicability of our framework for other models of cognition, in Section 6.

2. Optimization Problems as Models of Human Cognition

Psychological theories that are situated at Marr's computational-level 1982, or what Anderson (1990) called the rational-level, typically have the purpose to explain both the 'what' and the 'why' of some cognitive process. That is, such theories aim at answering the following two questions about the studied cognitive process.

1. *What* is the problem that the cognitive process is solving?
2. *Why* is the postulated problem the appropriate problem for the process to solve?

An answer to (1) is to be posed in the form of a computational problem, i.e., an input-output mapping (As computational problems and functions both define mathematical input-output mappings, we will use the words 'problem' and 'function' interchangeably). An answer to (2) may take the form of reasons why it may be useful, functional, desirable or rational for the cognitive process (or the cognizer) to be solving instances of the postulated problem. Possibly because explaining the 'why' is seen as part of explaining the 'what', many computational-level theories take the form of optimization problems. In this paper we will be concerned specifically with computational-level theories of this type,[3] and we will focus on the question of how to assess the computational plausibility of optimization functions *qua* 'what' explanations, regardless of whether or not those same functions also figure in 'why' explanations (cf. footnote 1).

[3]Our framework also applies, however, to computational-level theories that take the form of 'satisficing' problems (i.e., problems where the output has to have some satisfactory value; e.g. Simon (1957)), because such problems can always be redefined as optimization problems (i.e., all values equal to, or higher than, k are defined as 'optimal'.)

We present here a unified way to describe any type of optimization problem, following notational conventions from computer science (see Ausiello et al. (1999, Section 1.4.2)). Any optimization problem can be defined as consisting of five components as specified by the following template:

> **Problem name:** A name by which to refer to the problem, Π.
> **Input:** An input $i \in I$ belonging to a class of possible inputs I.
> **Candidate Solutions:** The set of candidate solutions, $cansol(i)$.
> **Value function:** A function that associates a value $v(c)$ with each candidate solution $c \in cansol(i)$.
> **Goal:** The optimization goal, which is either to minimize or to maximize $v(c)$ over all $c \in cansol(i)$.

Table 1 illustrates how the six computational-level theories mentioned in the Introduction can be formulated as optimization problems by filling in each of the five components defined above. To understand such computational-level theories as making the claim that some human cognitive process 'solves' or 'computes' a given optimization problem, we need to define what we mean by this. We say an algorithm *solves* (or, equivalently, *computes*), an optimization problem Π if it produces as output a candidate solution $c_{opt} \in cansol(I)$ that has the optimal value $v(c_{opt}) = v_{opt}$, where $v_{opt} = \min_{c \in cansol(i)} v(c)$ ($v_{opt} = \max_{c \in cansol(i)} v(c)$) if the goal is to minimize (maximize) $v(c)$ over all $c \in cansol(i)$. Here we say c_{opt} is an *optimal output* for Π and v_{opt} is the *optimal value*. It is often convenient to describe an optimization problem as an input/output mapping, where the last three components are collapsed into the definition of the required output, e.g.,

> **Problem name:** Π.
> **Input:** An input $i \in I$ belonging to a class of possible inputs I.
> **Output:** A candidate solution $c \in cansol(i)$ such that $v(c)$ is maximized (minimized).

The second column in Table 2 lists the (optimal) outputs associated with each of our example theories.

Note that our example optimization problems are not yet well-defined, i.e., they are *informal* computational-level theories. Each such informal theory can be formalized in many different ways (in which case they would be typically be referred to as computational models). See, for example, van der

Table 1: Optimization problems associated with six optimization theories from cognitive science.

Problem Name	Input	Candidate Solutions	Value Function	Goal
Perceptual Encoding	An image $i \in I = \{0,1\}^n$ and a decoding function $D : E \rightarrow I$ mapping encodings to images	All possible encodings $e \in E$ such that $D(e) = i$	The length of $e \in E(i)$	MIN
Transformational Similarity	Two representations x and y and a set of transformations T	All sequences $T(x,y) \in T^*$ that transform x into y	Length of $t \in T(x,y)$	MIN
Object Categorization	A set of objects A with for each pair of objects $(a,b) \in A \times A$ a similarity value $s(a,b) > 0$	All possible partitions of A	For partition $(A_1, A_2, \ldots, A_k)$, the value of the sum $\sum_{a,b \in A_i} s(a,b) - \sum_{a \in A_i, b \in A_j, i \neq j} s(a,b)$	MAX
Subset Choice	A set of choice alternatives A and a utility function $u : 2^A \rightarrow \mathcal{N}$, where 2^A is the powerset of A	All possible subsets $A' \subseteq A$	The utility $u(A')$ for $A' \subseteq A$	MAX
Belief Fixation (Bayesian)	A set of hypotheses H, a set of observations D, and a knowledge base K	All possible truth assignments $T : H \rightarrow \{true, false\}$	The conditional probability $P(D\|H', K)$, where $H' \subseteq H$ are the hypotheses believed to be true	MAX
Belief Fixation (Coherentist)	A set of propositions P and a set C of positive and negative constraints on $P \times P$	All possible truth assignments $T : H \rightarrow \{true, false\}$	Number of constraints in C satisfied by T	MAX

Helm (2004), van der Helm and Leeuwenberg (1996), Müller et al. (2009), van Rooij (2008), van Rooij et al. (2005), Abdelbar and Hedetniemi (1998), and Thagard and Verbeurgt (1998), respectively, for formal computational models consistent with the six informal theories in Table 1. Many such formalizations turn out to postulate optimization problems that are computationally intractable. We next explain why this may pose a problem for purposes of psychological explanation.

3. Computational Intractability

3.1. Computational Intractability as Non-polynomial Time-complexity

Informally, an optimization problem is said to be computationally intractable if it consumes an unrealistic amount of computational resources to solve it. In this paper we will be concerned specifically with the resource *time*. We measure (or estimate) computation time by the number of basic computational operations required to compute the output for any given input, and we express this number as a function of input size using the *Big-Oh* notation, $O(\cdot)$. A function $f(x)$ is $O(g(x))$ if there are constants $c \geq 0$, $x_0 \geq 1$ such that $f(x) \leq cg(x)$, for all $x \geq x_0$. In other words, the $O(\cdot)$ notation gives a smoothed, simplified version $g(x)$ of $f(x)$ that ignores constants and lower-order terms and thus focuses attention on the broad behavior of $f(x)$ as x goes to infinity. For this reason $O(g(x))$ is also called the *order of magnitude* of $f(x)$. Let $|i|$ denote the size of the input $i \in I$ for an (optimization) problem $\Pi : I \rightarrow O$. Then the *running time* of an algorithm computing the problem Π is said to be $O(f(|i|))$ if the number of steps computed by the algorithms is on the order of $f(|i|)$. The time-complexity of a given problem Π is equal to the running time of the fastest algorithm computing it. We will say that an optimization problem Π is computationally intractable (for all but small inputs) if the time required to compute it grows excessively fast as a function of input size. To make precise what we mean by "excessively fast", we adopt a definition that is widely used in both computer science and cognitive science (Frixione, 2001; Garey and Johnson, 1979; van Rooij, 2008):

Definition 1. (Computational Intractability) An optimization problem Π is *computationally tractable* if it can be computed in polynomial-time, i.e., time $O(|i|^{\alpha})$, where α is a constant. If an optimization problem requires super-polynomial time, e.g., exponential-time $O(\alpha^{|i|})$, where α is a constant, then Π is *computationally intractable*.

Table 2: Optimal and approximate outputs of optimization problems associated with six optimization theories from cognitive science.

Problem Name	Optimal Output	Value-Approx Output	Struct-Approx Output
Perceptual Encoding	An encoding $c \in E(i)$ such that length of c is minimized	An encoding $c' \in E(i)$ such that length of c' is "close to" the length of c	An encoding $c' \in E(i)$ that "resembles" c
Transformational Similarity	The shortest sequence of transformations t that transform x into y	A sequence of transformations t' with length "close to" the length of t	A sequence of transformations t' that "resembles" t
Object Categorization	A partition of A into $A_1, A_2, \ldots, A_k$ such that $val(A_1, A_2, \ldots, A_k) = \sum_{a,b \in A_i} s(a,b) - \sum_{a \in A_i, b \in A_j, i \neq j} s(a,b)$ is maximized	A partition $A'_1, A'_2, \ldots, A'_k$ such that $val(A'_1, A'_2, ..., A'_k)$ is "close to" $v(A_1, A_2, \ldots, A_k)$	A partition $A'_1, A'_2, \ldots, A'_k$ that "resembles" the partition $A_1, A_2, ..., A_k$
Subset Choice	A subset $A' \subseteq A$ such that $u(A')$ is maximized	A subset $A'' \subseteq A$ such that $u(A'')$ is "close to" $u(A')$	A subset $A'' \subseteq A$ that "resembles" $A' \subseteq A$
Belief Fixation (Bayesian)	A truth assignment T that maximizes the conditional probability $P(H'\|D, K)$	A truth assignment T' such that $P(H''\|D, K)$ is "close to" $P(H'\|D, K)$	A truth assignment T' that "resembles" T
Belief Fixation (Coherentist)	A truth assignment T that satisfies the maximum number of constraints	A truth assignment T' that satisfies "close to" the number of constraints satisfied by T	A truth assignment T' that "resembles" T

Table 3: Function growth rates.

n	$2n$	n^2	n^3	2^n
2	4	4	8	4
5	10	25	125	32
10	20	100	1000	1024
20	40	400	8000	1048576
50	100	2500	125000	$> 10^{15}$
100	200	10000	1000000	$> 10^{30}$
200	400	40000	8000000	$> 10^{60}$

To see why this definition has merit, compare the speed with which polynomial functions (say, $2n$, n^2 or n^3) and an exponential function (say, 2^n) grow as a function of input size (n). Table 3 shows how for small n, the numbers n^2 and 2^n do not differ much, and 2^n is even smaller than n^3, but as n grows, 2^n rockets up so fast that it is no longer plausible that a resource-limited mind/brain can perform that number of computations in a reasonable time-frame. As reference points, consider that the number of neurons in a human brain is estimated to be 10^{12}, and 10^{27} is about the number of seconds that have passed since the birth of the universe. It seems highly unlikely that a human mind could perform this number of operations (say, simply *count* the number) in only a couple of minutes (which is a generous upper bound on the time-scale of most cognitive processes of interest). Even if a human mind/brain could perform as many parallel computations per second as there are neurons in the brain, it would take days for it to complete 10^{18} operations and as much as centuries for it to complete 10^{27} operations. In other words, knowing that an optimization problem is of super-polynomial time-complexity is good reason to consider that optimization problem computationally intractable for all but small input sizes.

Polynomial-time complexity is one of the most commonly used notions of tractability and is the definition adopted by cognitive psychologists that claim efficient (i.e., polynomial-time) approximability of the optimization problem that they postulate as rational-level cognitive models (Chater et al., 2006; Love, 2000; Thagard and Verbeurgt, 1998). Therefore we adopt this classical definition of tractability throughout this paper. We note, however,

that nothing in the formal framework that we propose in Section 5.1 depends on this particular formalization of tractability, and the framework can be adapted for any other formalization as well (e.g., for parameterized notions of tractability, see van Rooij (2008) and van Rooij and Wareham (2008)).

3.2. Computational Intractability of Cognitive Models

The problem of computational intractability can arise whenever the size of the set of candidate solutions for an optimization problem (also called its *search space*) is a super-polynomial function of the input. Take, for example, the optimization problems sketched in Table 1. In the Transformational Similarity theory there exist k^n sequences of transformations of length at most k when given $|T| = n$ transformational rules,[4] in the Object Categorization theory the number of possible ways in which one can partition $|A| = n$ objects is lower bounded by 2^n, in the Subset Choice theory the number of possible subset sets of $|A| = n$ choice alternatives is given by 2^n (including the empty set), and in the Belief Fixation theories the number of possible truth assignments is 2^n for $|H| = n$ hypotheses. As for the Perceptual Encoding theory, we observe that the number of possible encodings of strings depends on the encoding rules employed, but if any encoding rule can be used, then the number can far exceed 2^n for an image of size $|i| = n$.

As all these search spaces are of exponential size, exhaustively searching them in order to find the one output that is optimal is computationally intractable (see Table 3). This means that these models may be considered computationally intractable unless one can show that there exists at least one way to find an optimal output without having to perform an exhaustive search of the entire search space, or a super-polynomial part thereof. Showing that there exists at least one polynomial-time algorithm that computes the optimization function $\Pi : I \rightarrow O$ suffices to show that Π is computationally tractable. Showing, on the other hand, that no such polynomial-algorithm can ever exist is very difficult and even impossible for most optimization functions of interest to cognitive psychologists. To overcome this difficulty, one can use techniques developed by complexity theorists to show that an optimization problem Π in $\mathcal{NP}$-hard. We next explain how a proof that a problem Π is $\mathcal{NP}$-hard would constitute strong evidence that Π is not

[4]If there is no upperbound on sequence length, then the Transformational Similarity theory allows for an infinity of possible sequences of transformations.

computationally tractable (for information on how to prove $\mathcal{NP}$-hardness the reader is referred to Garey and Johnson (1979)).

We first explain the classes $\mathcal{NP}$ and $\mathcal{P}$, which are classes of decision problems. A decision problem is a problem with a binary ('Yes'/'No') output. It is often stated in the form of a question, e.g.,

> **Problem name:** Π_D.
> **Input:** An input $i \in I$ belonging to the set of possible inputs I.
> **Question:** A 'Yes'/'No'-question about i.

By solving (or computing) a decision problem we mean correctly answering the posed question with either 'Yes' or 'No'. Now observe that for any given optimization problem we can formulate an associated decision problem by introducing a threshold value k, and then asking the question 'Does there exist a candidate solution $c \in \mathit{cansol}(i)$ such that $v(c) \geq k$ (if the goal is maximization), or $v(c) \leq k$ (if the goal is minimization)?', e.g.,

> **Problem name:** Π_D.
> **Input:** An input $i \in I$ and an integer k.
> **Question:** Does there exist a candidate solution $c \in \mathit{cansol}(i)$ such that $v(c) \geq k$ (or $v(c) \leq k$)?

Note that as long as the value $v(c)$ can be computed in polynomial-time (which is the case for known formalizations of our six example theories), then there always exist proofs of correctness of a 'Yes'-answer for Π_D that can be verified in polynomial-time. Namely, if we present one c with $v(c) \geq k$ or $v(c) \leq k$ as necessary then we have such a proof. If Π_D has this property then it is said to belong to the class $\mathcal{NP}$, i.e., the class of decision problems for which 'Yes'-proofs can be verified in polynomial-time. Now note that even if verifying a 'Yes'-proof is easy, finding such a proof may be quite hard and possibly cannot be done in polynomial-time. In other words, it is conceivable, and even plausible, that there exist $\mathcal{NP}$ decision problems that do not belong to the class $\mathcal{P}$, i.e., the class of decision problem that can be solved in polynomial time. This idea is expressed by the conjecture that $\mathcal{P} \neq \mathcal{NP}$.

One reason for believing this conjecture is that there exist so-called $\mathcal{NP}$-hard problems. An $\mathcal{NP}$-hard problem (which may be an optimization problem Π, its associated decision problem Π_D, or any other decision problem) is a problem with the property that if it were solvable in polynomial-time then

all decision problems in $\mathcal{NP}$ would be solvable in polynomial-time. Despite continued efforts over the last 50 years, nobody to date has been able to devise a polynomial-time algorithm for solving an $\mathcal{NP}$-hard problem, providing empirical support for the idea that $\mathcal{P} \neq \mathcal{NP}$. Another reason for believing that $\mathcal{P} \neq \mathcal{NP}$ is mathematical intuition: It seems intuitively possible that there can exist problems for which solutions are easy to check but hard to find (think, for example, of puzzles such as Sudoku or Crossword puzzles). For a summary of current evidence in support of the $\mathcal{P} \neq \mathcal{NP}$ conjecture, see Fortnow (2009). In the remainder of this paper, consistent with current computer science practice, we will work under the assumption that $\mathcal{P} \neq \mathcal{NP}$.

Are any optimization problems that figure in computational-level theories $\mathcal{NP}$-hard? Yes, in fact, many are. Examples can be found in the domains of vision (Tsotsos, 1990), reasoning (Levesque, 1988), planning (Bylander, 1994), and analogy derivation (Veale and Keane, 1997), and include formalizations of the theories listed in Table 1 (van der Helm, 2004; van der Helm and Leeuwenberg, 1996; Müller et al., 2009; van Rooij, 2008; van Rooij et al., 2005; Abdelbar and Hedetniemi, 1998; Thagard and Verbeurgt, 1998). In other words, many optimization problems are implausibly computable by resource-bounded minds and, consequently, fail to provide computationally plausible *explanations* of human cognitive outputs. Yet, many such computationally intractable optimization problems often seem to adequately *describe*, and sometimes even *predict*, human cognitive outputs. This poses a theoretical challenge: How can an intractable model be adapted so as to make it tractable without loss of descriptive and predictive power? We see at least two possible—not necessarily mutually exclusive—approaches to achieving this aim.

The first possibility is to investigate if perhaps the postulated optimization problem is an overgeneralization of the modeled cognitive ability. For example, the input domain of the function may include all logically possible inputs (e.g., all possible similarity weight assignments in the Categorization model), while in fact human cognizers may only ever be confronted with a restricted set of inputs (e.g., only similarity weights that satisfy some metric). It is well-known that an intractable optimization function $\Pi : I \rightarrow O$ may be tractable for one or more restricted input domains $I' \subset I$. If there exists a psychologically plausible and ecologically motivated restriction I' that renders Π tractable then we can resolve the intractability paradox by simply revising our model to be $\Pi' : I' \rightarrow O$ rather than $\Pi : I \rightarrow O$. This approach was pursued, for example, by van Rooij et al. (2005) for the Subset

Choice model of Fishburne and LaValle (1996), by van Rooij et al. (2008) for the Structure Mapping analogy model of Gentner (1983), by Müller et al. (2009) for the Transformational Similarity model of Hahn et al. (2003), and by Blokpoel et al. (2010) for the Bayesian Inverse Planning model of Baker et al. (2009).

The second possibility is to weaken the claim that the modeled cognitive process computes $\Pi : I \to O$ exactly so as to claim that it computes $\Pi : I \to O$ approximately. Although this approach seems in principle a sensible one, and is commonly adopted (or alluded to) by cognitive modelers, few modelers make precise exactly what they mean by 'approximately'. Also, rarely are claims of approximability in the cognitive science literature accompanied by explicit demonstrations that at least one tractable (polynomial-time) algorithm exists for approximating the intractable optimization problem in the relevant sense. In this paper, we aim to give the approximation approach more substance by making the idea of approximating an optimization function mathematically precise.

Before continuing, we would like to emphasize that the two strategies noted above for dealing with intractability—input restriction and approximation—are not mutually exclusive and may yet be fruitfully combined. In fact, our framework for assessing the approximability of a function $\Pi : I \to O$ makes no assumption about the nature of I and, without loss of generality, the set I can be assumed to include only those inputs I' that are considered to be ecologically relevant (i.e., $I = I'$). We note that the possibility of input restriction by itself need not render approximation obsolete. For instance, there exist intractable optimization functions (e.g., the Traveling Salesperson Problem (TSP)) that remain intractable even for stringent input restrictions (for example, TSP restricted to Euclidean distances)[5] yet are

[5]This is not so say that no input restrictions can make computational-level theories tractable to compute, but rather that it is important to verify this on a case by case basis. For instance, it has been shown that intuitions about which input restrictions render computational-level theories tractable are often mistaken (Blokpoel et al., 2010; van Rooij et al., 2008). Moreover, even if input restrictions render the computational-level theory tractable to compute it remains important to assess the ecological validity of said restrictions. This is important because cognitive modelers often make simplifying assumptions on the input domain of their theories to model the specifics of stimuli used in the lab to test their theories. Although such simplifying assumptions may buy tractability of the computational-level model—which seems necessary for the modeler to compute the model's predictions for the stimuli used—the question remains to what extent such a

tractably approximable under these input restrictions (see also Section 4.1 for more details). For a recent demonstration of the potential interaction between input restriction and approximation in a probabilistic computational-level model of cognition, we refer the reader to Kwisthout and van Rooij (2012), who have shown that approximation is neither a *placebo* nor *panacea* when it comes to coping with intractability.

4. Approximating Intractable Computational-level Models

What do computational-level modelers mean when they claim or hypothesize that the intractable optimization functions that their theories postulate can be tractably approximated? In this section we consider two possible meanings of 'approximation': viz., value-approximation and structure-approximation. We discuss the conditions under which the one or the other seems the more relevant notion of approximation. We observe that although the notion of value-approximation concords with tradition in computer science, cognitive science seems to also require the additional notion of structure-approximation.

4.1. Value-approximation: A Notion from Computer Science

The idea that computer science and its subdisciplines are providing algorithms that approximate optimization functions in a sense that is relevant also for computational-level theorizing is illustrated by the following quote from Chater et al. (2006):

> ... 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. (p. 290)

In computer science, an algorithm is typically said to be a tractable approximation algorithm for an optimization problem if it runs in polynomial-time and produces an output with associated value that is "close" to the optimal

restricted model can scale to ecologically relevant situations in the real-world (Gigerenzer et al., 2008, p. 236).

value (see Ausiello et al. (1999) and Hamilton et al. (2007, Section 2.1) for details). We refer to this notion of approximation as *value-approximation*. Several types of value-approximation algorithms are commonly distinguished in computer science, each defined by a different value-closeness criterion (see Ausiello et al. (1999) and Hamilton et al. (2007, Section 2.1) for details).

The notion of value-approximation is a natural one for many computer science applications, e.g., for the design of approximation algorithms for optimization problems arising in operations research and network design. As an illustration we consider the well-known Traveling Salesperson Problem (TSP). The input to this optimization problem is a set of cities with pairwise distances (or costs of travel) and the output is a tour that visits every city that minimizes the total traveled distance (or total cost). TSP is known to be $\mathcal{NP}$-hard, even if the distances between cities are constrained to the Euclidean metric (denoted E-TSP) (Garey and Johnson, 1979, Problem ND23). Yet, value-approximation algorithms for E-TSP exist that guarantee to produce a solution with a value that is within an arbitrary specified closeness of the optimal value within polynomial time (Ausiello et al., 1999, Problem ND34).[6] Given that finding a shortest (or cheapest) tour is computationally intractable, having such an approximation algorithm may be a good thing. After all, if the salesperson cannot find a tour that is guaranteed to be the shortest (cheapest) in a reasonable time, she may appreciate being able to quickly find a tour that is almost as short (or cheap) as the optimal tour.

Now let us consider how the notion of approximation illustrated in the TSP example applies to optimization theories of cognition. By analogy, a cognitive outcome can be considered an approximation of an optimal output to the extent that its associated value for the relevant objective function is close to the optimal value (see Table 2). For example, a percept may have a degree of descriptive complexity that is close to minimum, a categorization may induce a difference in within- and between-category similarities that is close to maximum, a chosen alternative may have close to maximum utility, and, lastly, a set of beliefs may have close to maximum conditional probability (for the Bayesian) or close to maximum explanatory coherence (for the Coherentist).

[6]To be fair, we should note that the degree of the polynomials in the running times of these algorithms increases dramatically as the specified degree of closeness decreases, making the guarantee of even moderate degrees of closeness impractical in general.

In all these cases, the cognitive output can be seen as approximating the (optimal) output of the postulated optimization problem in the *value*-approximation sense. Furthermore, it is conceivable that, even though computing the optimal outputs is computationally intractable, value-approximating those outputs may be tractable. It may seem, then, that value-approximation is a way to cope with the computational intractability of optimization theories of cognition. This certainly seems to hold for a special class of optimization theories, viz. those in which the target explanandum is modelled by the *value* of an optimal output.[7] An example may be the Transformational Similarity model by Hahn et al. (2003) (see also Table 2). Here, the primary explananda are the similarity judgments themselves, which are modelled by (a numeric function of) the *length* of the sequence of transformations. It is unlikely, however, that value-approximation suffices as a notion of approximation for all cognitive modeling, because value-approximations do not fare well as explanations of *all* relevant properties of cognitive outcomes.

Recall that outputs of intractable optimization problems describe (and sometimes predict) cognitive outcomes, and the challenge is to achieve model tractability by weakening the optimality requirement while maintaining this descriptive power. Description and prediction, in this context, often means that there is a relevant resemblance between the outputs of the optimization function and the observed (or inferred) cognitive outputs for different kinds of inputs. This resemblance may pertain to the values associated with each of the two outputs, but it does not need to. More importantly, the relevant resemblance is seldom limited to value alone. For example, an Object Categorization model (as sketched in Table 2) may be considered to describe and predict human categories well to the extent that the object partitions produced by the model structurally resemble the partitions observed in human categorizations. To accommodate this concept of approximate description, we introduce a new and generalized notion of approximation in Section 4.2.

[7]For simplicity, we assume that a theory has one primary explanandum. We acknowledge that an optimization theory could be used to explain more than one cognitive phenomenon. In those cases, which notion of approximation is most relevant will depend on which properties of a theory's output are taken to correspond to the different target explananda.

4.2. Structure-approximation: A Notion for Cognitive Science

For purposes of computational-level theorizing about cognition, a cognitive outcome and an optimal output can presumably be seen as approximations of each other to the extent that the two are similar in some relevant sense. For example, a perceptual encoding may be similar to the simplest perceptual encoding, a categorization may induce a partition that is similar to the optimal partition, a set of chosen alternatives may be similar to the set with maximum utility, and a set of beliefs may resemble either the set of beliefs that maximizes conditional probability or the set of beliefs that maximizes explanatory coherence. This form of approximation we call *structure-approximation* (see Table 2).

It is conceivable that the relevant dimension of similarity for a cognitive modeler is the extent to which the values associated with the two outputs are close to each other. For this reason, structure-approximation can be seen as a generalization of the notion of value-approximation, i.e., the former includes the latter as a special case. More typically, however, 'similarity' will refer to the extent to which the two output structures (the two encodings, the two partitions, the two sets of choices, or the two belief assignments) structurally resemble each other. Unless otherwise indicated, in the remainder of this paper, by structure-approximation we will mean that two outputs are similar in this stricter sense.

At first sight, one may tacitly assume that there is a close relationship between value-approximation and structure-approximation. However, this assumption has been conjectured to be wrong, e.g., by Chater and Oaksford (1999) in the context of rational analysis:

> "The tacit assumption is that good suboptimal behaviors will be similar to optimal behaviors, but this is not necessarily true—in principle, it is possible that a problem could have two or more good solutions that are very different." (p. 59)

In fact, as it turns out value-approximation and structure-approximation are fully dissociable, in the following sense: (1) an approximate output can be close to an optimal output in terms of value, yet far from the optimal output in terms of structural similarity (cf. the quote from Chater and Oaksford (1999)), but also conversely (2) an approximate output can be structurally similar to the optimal output, yet far from the optimal output in terms of

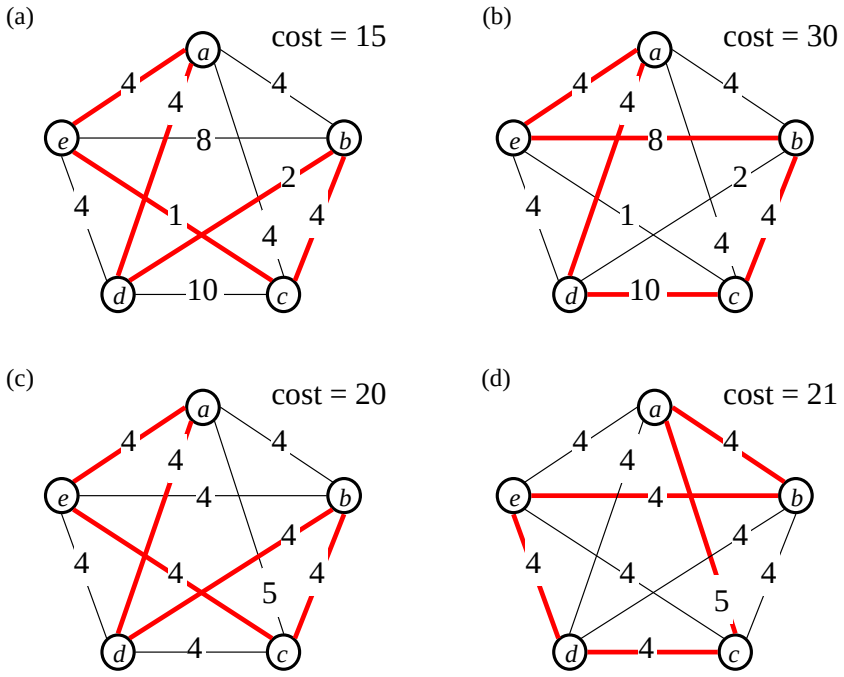


Figure 1: Disassociation of structure- and value-approximability for the TRAVELING SALESMAN Problem. (a-b) Solutions that are close in structure need not be close in value. Though tours $c_{opt} = ABECDA$ (edge-set $\{AB, BE, EC, CD, DA\}$)[part (a)] and $c = ABEDCA$ (edge-set $\{AB, BE, ED, DC, CA\}$)[part(b)] are very similar (differing by one swap of adjacent elements in the vertex-sequences and two edges in the edge-sets), $v(c_{opt}) = 15$ is optimal while $v(c) = 30$ is the worst possible. (c-d) Solutions that are close in value need not be close in structure. Though tours $c_{opt} = ABECDA$ (edge-set $\{AB, BE, EC, CD, DA\}$)[part (c)] and $c = AEDBCA$ (edge-set $\{AE, ED, DB, BC, CA\}$)[part (d)] are as dissimilar as possible (differing by multiple swaps of adjacent elements in the vertex-sequences and having disjoint edge-sets), $v(c_{opt}) = 20$ is optimal and $v(c) = 21$ is almost optimal.

its associated value. Figure 1 presents an illustration of this dissociation between value- and structure-approximation for TSP.

The example in Figure 1 shows that a value-approximation does not always yield a structure-approximation, nor vice versa. So which approximation is better? This all depends on the context and the aims of the approximator. A salesperson presumably does not care so much about the ordering of cities in the tour as she does for the cost of traveling. Therefore, when given the choice between a value- and a structure-approximation, the salesperson will likely prefer the former. A cognitive modeler, on the other hand, who cares to describe and predict structural properties of cognitive outcomes, and who considers optimization as a means to produce (and hence, explain) those outcomes, would prefer to establish *at least* the structure-approximability of the postulated optimization (although, for the postulated optimization to have complete explanatory force, value-approximability may be required as well). Moreover, a cognizer who aims at constructing veridical or 'truth-like' representations of events and objects in the world may prefer to structurally approximate the optimal representations (regardless any (dis)ability to value-approximate them). Such representations are also said to have high verisimilitude.

This last possibility was also noted by Millgram (2000) for the Coherence model proposed by Thagard and Verbeurgt (Thagard and Verbeurgt, 1998; Thagard, 2000). The Coherence model is a formalization of the optimization problem sketched in the last entry in Table 1, i.e., a model of human belief fixation. Because we will use this model as a case-study for our structure-approximation analyses in Section 5.2, we give the formal specification of the model below:

> Coherence
> *Input*: A belief network $N = (P, C, w)$ where P denotes a set of propositions, $C = C^- \cup C^+ \subseteq P \times P$ denotes a set of positive and negative constraints such that $C^+ \cap C^- = \emptyset$, and $w : C \to \mathcal{R}^+$ is a function that associates a positive weight $w(p, q)$ with each constraint (p, q).
> *Output*: A truth assignment $T : P \to \{true, false\}$, such that the *coherence value* $COH(T) = \sum_{(p,q)\in C^+, T(p)=T(q)} w(p, q) + \sum_{(p,q)\in C^-, T(p)\neq T(q)} w(p, q)$ is maximized.

It was found by Thagard and Verbeurgt (1998) that COHERENCE is $\mathcal{NP}$-hard. To cope with this theoretical challenge, Thagard and Verbeurgt (1998) proposed several value-approximation algorithms as candidate algorithmic-level explanations of how humans may approximate the postulated optimization function. Millgram criticized the approach on the grounds that if one were to approximate the optimal output, one would want ones beliefs to correspond approximately with the beliefs in the maximally coherent belief assignment. In other words, structure-approximation would be a more natural and useful notion in this setting. Furthermore, so Millgram argued, value-approximations do not automatically yield structure-approximations, because it is possible for two belief assignments to be arbitrarily close in coherence value yet arbitrarily far from each other in terms of which beliefs are held "true" and which ones "false" (p. 89 in Millgram (2000); see Figure 2 for an illustration of the dissociation between value and structure observed by Millgram, as well as the reverse dissociation).

To our knowledge, the type of structure-approximation envisioned by Millgram has not received much attention from complexity theorists (but see footnote 2) and no general framework for assessing degrees of structure-approximability exists in the literature other than the one that we and others introduced in a technical report Hamilton et al. (2007). Our framework builds on analogies with the existing framework for value-approximation and uses the notion of 'distance' as a measure of the similarity between optimal and approximate outputs. In the next section we review the details of our proposed framework and illustrate its use for assessing the approximability of the COHERENCE model.

5. A General Framework for Assessing Structure-Approximability

Since value-approximations do not generally yield structure-approximations, it is currently unclear if the many value-approximability results for optimization problems that can be found in the computer science literature are of any use for purposes of approximate cognitive modeling. It would benefit cognitive modeling if the relationship between value- and structure-approximation was better understood and, where the relationship is weak, if there were tools for studying structure-approximability of optimization problems directly. We aim to stimulate research to this end by:

1. proposing a definition of structure-approximation, based on the notion of distance between pairs of solutions; and

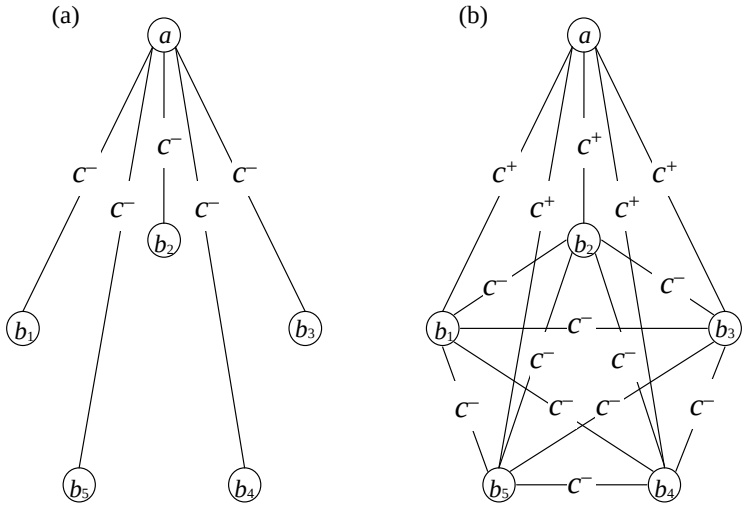


Figure 2: Disassociation of structure- and value-approximability for the COHERENCE Problem (adapted from Figure 1 in Hamilton et al. (2007)). (a) Solutions close in structure need not be close in value. (b) Solutions close in value need not be close in structure. Both figures show instances of COHERENCE in which constraints are indicated by edges, such that each positive constraint has weight c^+ and each negative constraint has c^-. In part (a), given an optimal solution c_{opt} in which a is assigned $true$ and all other propositions are assigned $false$ and a solution c in which all propositions are assigned $false$, observe that though $d_H(c, c_{opt}) = 1$, the difference in the values of c and c_{opt} is the largest possible. In part (b), given an optimal solution c_{opt} in which a and half of the b-propositions are set to $false$ and the remaining b-propositions to $true$ and a solution c in which all propositions are set to $false$, it is possible to adjust the positive and negative constraint weights c^+ and c^- so that the values of c_{opt} and c are arbitrarily close but $d_H(c, c_{opt})$ is equal to half of the total number of propositions.

2. presenting some techniques for proving various types of structure-approximability.

This is done below in Sections 5.1 and 5.2, respectively. We illustrate the use of the techniques by using the problem COHERENCE introduced in Section 4.2 as an example. Parenthetically, we note that there is an intimate connection between the computation of COHERENCE and harmony maximization in Hopfield networks (see Appendix A). This means that many properties and results reported in Section 5.2 for COHERENCE directly generalize to neural network models of the relevant type, making our structure-approximability results of independent interest to the neural network community (see also Section 6.3).

5.1. Definitions: Structure-Approximability

In this section, we will give an overview of our structure-approximability framework (for more details, see Hamilton et al. (2007)). In addition to the notations introduced in Section 2, let $optsol(i)$ be the set of optimal solutions $c_{opt} \in cansol(i)$ for an instance i of problem Π.

To discuss structure-approximability of optimal outputs, we first need a definition of what it means for candidate solutions to be similar to a particular degree. We can capture this notion of solution similarity using the concept of a distance function. Let d be a **solution distance (sd) function** associated with an optimization problem Π such that for an instance i of Π and $c, c' \in cansol(i)$, $d(c, c')$ is the distance between these solutions. As it will occasionally be convenient to define the minimum distance of $c \in cansol(i)$ to a set $X \subseteq cansol(i)$, let $d(c, X) = \min_{c' \in X} d(c, c')$. Note that each sd-function assumes a particular representation for the candidate solutions of its associated optimization problem. We will assume that each sd-function d is a metric, and hence satisfies the following four properties:

1. For all x, $d(x, x) = 0$.

2. For all distinct x and y, $d(x, y) > 0$.

3. For all x and y, $d(x, y) = d(y, x)$ (**symmetry**)

4. For all x, y, and z, $d(x, y) \leq d(x, z) + d(z, y)$ (**triangle inequality**).

Note that for a problem Π, there may be many such sd-functions.

The definitions above are readily applicable to problems of interest. For example, in the case of COHERENCE, we can represent a belief-assignment to P (and hence a candidate solution) as a $|P|$-length binary vector in which a 1 (0) in position j means that the observation or hypothesis corresponding to that position is set to *true* ($false$). Given this, let the sd-function be the **Hamming distance (d_H)** between the given solution-vectors, i.e., the number of positions at which these vectors differ. Hamming distance is a metric and hence a valid sd-function; more importantly, Hamming distance corresponds to solution-closeness as envisioned by Millgram (2000).

By analogy with value-approximability in the computer science literature (Ausiello et al., 1999, Section 3), we distinguish two types of polynomial-time structure-approximation algorithms in this paper. The first of these types computes solutions that are within an additive factor of optimal. Note that these factors range in value between 0 and $d_{max}(i)$ inclusive, where $d_{max}(i) = \max_{c,c' \in cansol(i)} d(c, c')$ is the largest possible structural difference between two candidate solutions.

Definition 2. Given an optimization problem Π, a sd-function d, and a non-decreasing function h, an algorithm A is a **polynomial-time $h(|i|)/d$ additive structure-approximation (s-a-approx) algorithm** if for every instance i of Π, $d(A(i), optsol(i)) \leq h(|i|)$ and A runs in time polynomial in $|i|$.

The second type of structure-approximation algorithm computes solutions that are within a multiplicative factor of the largest possible difference from optimal, i.e., $d_{\max}(i)$. For convenience, these factors are stated relative to versions of sd-functions that have been normalized by their maximum possible values and hence range in value between 0 and 1 inclusive. We denote the normalized version of an sd-function $d(c, c')$ by $d^N(c, c') = \frac{d(c,c')}{d_{max}(i)}$.

Definition 3. Given an optimization problem Π, a sd-function d, and a constant $r \in (0, 1)$, an algorithm A is a **polynomial-time r/d multiplicative structure-approximation (s-m-approx) algorithm** if for every instance i of Π, $d^N(A(i), optsol(i)) \leq r$ and A runs in time polynomial in $|i|$.

The s-a-approx algorithms give the strongest form of structure-approximation when $h(|i|)$ is either constant or a small sublinear function of i, e.g., $\log \log \log |i|$.

This kind of structure-approximation best corresponds to what a psychologist would typically think of when hearing the word "approximation", because the degree of approximation remains constant (or almost constant) for different input sizes. If no such s-a-approx algorithm exists, one may still be able to devise an s-m-approx algorithm; however, the systematic increase in the (possible) distance between the approximate solution and the optimal solution as instance size increases in such algorithms may be taken as evidence that the to be modelled process has not been well-captured by the model. Be that as it may, we leave it up to the cognitive modeler to decide which type of structure-approximation makes most sense for his/her domain of application and to justify his/her choice.

5.2. The Structure-Approximability of Coherence

In this section, we illustrate a set of proof techniques for proving various degrees of structure-inapproximability. All of these techniques build on Garey and Johnson's "instance-copy" strategy (see e.g. Garey and Johnson (1979, Theorem 6.7)). Previously, this strategy has only been used to show certain types of polynomial-time value-inapproximability, but we show how it can be adapted to prove several different types of structure-inapproximability. Though we apply these techniques here specifically to the Coherence model described in Section 4.2., these techniques have much wider applicability. In fact, we show this strategy works for all intractable (optimization) functions that satisfy a certain property which we call "self-paddability" (see Definition 4).

We start by giving an informal sketch of the instance-copy strategy. The intuition behind this strategy is as follows. The aim is to construct a proof by contradiction. First we assume the existence of an algorithm A that can tractably structure-approximate an intractable problem of interest Π within a factor d. Next we try to show that if A exists then the following algorithm A' for Π also exists:

Algorithm *A'*

1. Given instance I of Π, create an instance I' of Π consisting of $d+1$ copies of I.

2. Run A on I' to get solution O'.

3. As A outputs a solution that has be at most d deviations from optimal structure, at least one of the solutions to a copy of

I in I' must have an optimal solution in O'; return this solution .

If such an A' exists, then A' is a tractable algorithm for optimally solving Π. This, however, contradicts the intractability of Π, which means that such an algorithm A cannot exist. Note that as this argument assumes nothing about the nature of A but only its existence, the argument rules out the existence of *any* such algorithm A.

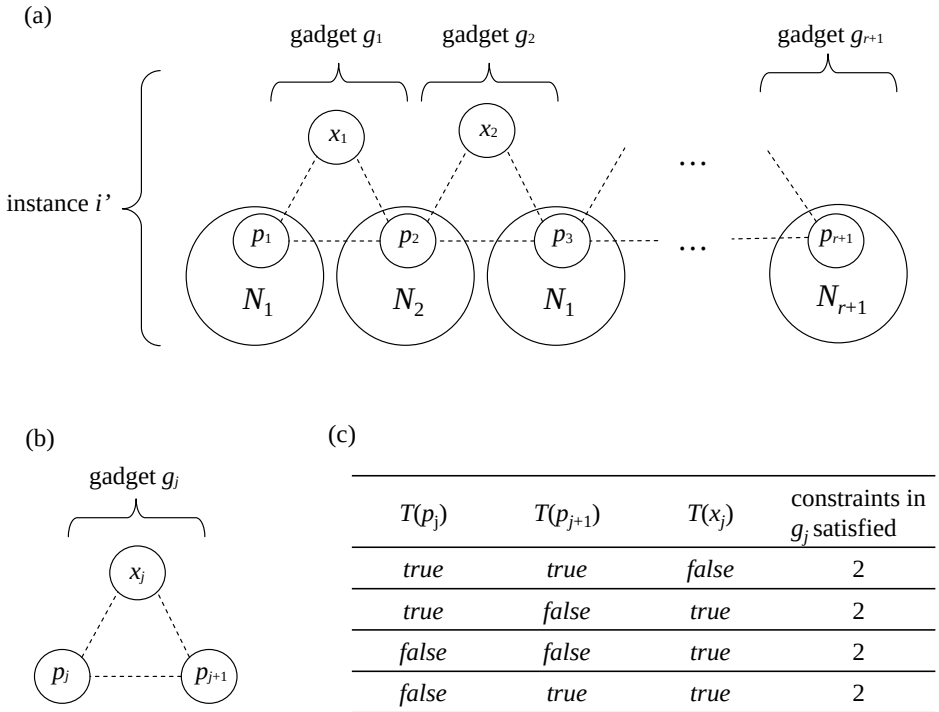
We now illustrate how the instance-copy strategy can be used for ruling out the strongest possible structure-approximability for COHERENCE, namely additive structure-approximability within a constant.

Theorem 4. COHERENCE *is not* r/d_H *s-a-approximable for any* $r \geq 1$ *(unless* $\mathcal{P} = \mathcal{NP}$*).*

Proof: Our proof is by contradiction—namely, we will show that if an r/d_H s-a-approx algorithm exists, then we can use it to solve the $\mathcal{NP}$-hard problem COHERENCE in polynomial-time, which would imply that $\mathcal{P} = \mathcal{NP}$. Figure 3 presents an illustration of the proof.

Given an instance $i = \langle N \rangle$ of COHERENCE, let p be an arbitrary proposition in N. Construct an instance $i' = \langle N' \rangle$ of COHERENCE in which N' consists of $r + 1$ copies of N which are connected together by gadgets[8] $\{(p_j, p_{j+1}), (p_j, x_j), (p_{j+1}, x_j)\} \in C^-$, $1 \leq j \leq r$, such that p_j and p_{j+1} are the propositions corresponding to p in copies N_j and N_{j+1} of N in i', the x_j are new propositions in N', each copy of N in N' has the same constraint-weights as in N, and the negative constraint-edges in the copy-connection gadgets all have weight 1 (see parts (a) and (b) of Figure 3). As shown in part (c) of Figure 3, in any optimal solution for i', the optimal satisfied-constraint value in each gadget will be 2, regardless of the belief-assignments of p_j, p_{j+1}, and x_j; hence, each gadget functions as an "insulator" which allows the propositions of each copy of N in i' to be assigned truth-values independently in an optimal solution for i'.

[8]In the computational complexity literature the term 'gadget' is commonly used to describe any small structure added to a problem instance by an algorithm to enforce some constraint or property. In our proofs we use gadgets to enforce connectedness of the newly constructed input network. The reason is that we want our proofs to be as general as possible (i.e., not only apply to disconnected networks) and, judging from applications of the Coherence model, it seems that human belief networks are often assumed to be connected (Thagard and Verbeurgt, 1998; Thagard, 2000).



$T(p_j)$	$T(p_{j+1})$	$T(x_j)$	constraints in g_j satisfied
true	*true*	*false*	2
true	*false*	*true*	2
false	*false*	*true*	2
false	*true*	*true*	2

Figure 3: Proving that COHERENCE is not r/d_H s-a-approximable for any constant $r > 1$. (a) Structure of instance i'. (b) Structure of gadget g_j. (c) Illustration that gadget g_j has optimal value 2 regardless of the truth-assignments to p_j, p_{j+1}, and x_j. By brute-force enumeration of the possible truth-assignments to p_j, p_{j+1}, and x_j one can establish that listed truth assignments are optimal. For proof details, see Theorem 4 in the main text.

Now, consider the following algorithm for solving COHERENCE: Given an instance i of COHERENCE, construct an instance i' of COHERENCE as described above. Apply the r/d_H s-a-approx algorithm to i', and let the produced solution be c'. Observe that c' consists of $r+1$ solutions to i, and that an optimal solution c'_{opt} for i' consists of $r+1$ optimal solutions for i. As $d_H(c', c'_{opt}) \leq r$ by definition and all differences between c and c' involve individual bits in the the $r+1$ solutions to i stored in the binary-vector representation of c', at least one of the solutions to i in c' cannot have been changed relative to c'_{opt} and is thus optimal (note that if any of these differences occur in propositions x_j in the connection-gadgets, even more of these solutions will be optimal). Therefore, we can compute $v(c)$ for each of the $r+1$ solutions $c \in cansol(i)$ in c' to find v_{opt}, which we can then use to solve i.

Observe that in the algorithm described above, the construction of i', the running of the r/d_H s-a-approx algorithm on i', and the scan of c' to compute v_{opt} can be done in time polynomial in $|i|$; hence, this algorithm runs in polynomial time. As COHERENCE is $\mathcal{NP}$-hard (Thagard and Verbeurgt, 1998), this implies that $\mathcal{P} = \mathcal{NP}$, completing the proof. ∎

In general, proving that a problem Π does not have an x/d s-a-approx algorithm using the approach above requires the following:

1. Propose a method for creating an instance i' of Π that encodes y copies of a given instance i of Π.
2. Propose a method which, given a solution to an instance i' created by the method in (1), can extract all y solutions for each of the encoded copies of i.
3. Ensure that the methods in (1) and (2) run in time polynomial in both the size of the given instance i of Π and the value y.
4. Ensure that the requested degree x of structure-approximability is strictly less than the number y of copies of i in the instance i' constructed in (1).

All of these requirements are important, but it is requirement (4) that is crucial, in that it forces optimal solutions for the given instance i of Π to appear in any x/d structurally-approximated solution for a constructed instance i' of Π.

There are simpler proofs of additive constant structure-inapproximability that do not involve instance copying (for example, run the r/d_H s-a-approx algorithm on instance $i' = \langle N = (P, C, w)\rangle$ of COHERENCE to create solution c' and examine all $\sum_{j=1}^{r} \left(\begin{array}{c} |P| \\ j \end{array} \right) = O(r|P|^r)$ candidate solutions of i' within Hamming distance r of c' to determine $v'_{opt} = v_{opt}$, which can then be used to solve i). Be that as it may, the instance-copy strategy can be adapted to show more powerful forms of structure-inapproximability. For example, using the instance-copy strategy, we can also rule out additive structure-approximability within any function that is logarithmic in the number of propositions in the given instance of COHERENCE.

Theorem 5. COHERENCE *is not* $O(\log_2 |P|)/d_H$ *s-a-approximable (unless* $\mathcal{P} = \mathcal{N}P$*).*

Proof: Once again, we will give a proof by contradiction which shows that if such an s-a-approx algorithm exists, we can solve a given instance $i = \langle N = (P, C, w)\rangle$ of COHERENCE in polynomial-time. By definition, $O(\log_2 |P|)$ is equivalent to $r' log_2 |P|$ for some constant $r' > 0$. If our requested degree of structure-approximability $d_H(c, c_{opt}) = O(\log_2 |P|) = r' \log_2 |P|$, let us set the number of copies of COHERENCE in constructed instance i' of COHERENCE to $|P|$, such that the number of propositions in i' is $|P'| = |P|^2 + (|P| - 1)$. Given this, we can derive an upper bound on the degree of structure-approximability as follows:

$$\begin{aligned} d_H(c', c'_{opt}) &\leq r' \log_2 |P'| \\ &= r' \log_2(|P|^2 + (|P| - 1)) \\ &\leq r' \log_2 2|P|^2 \\ &= r'(\log_2 2 + \log_2 |P|^2) \\ &= r'(1 + 2 \log_2 |P|) \end{aligned}$$

Observe that $r'(1 + 2 \log_2 |P|) < |P|$, the number of copies of i in i', when $|P|$ is suitably large; let this value of $|P|$ be denoted by r''.

Given the above, our algorithm for solving a given instance $i = \langle N = (P, C, w)\rangle$ of COHERENCE is as follows: If $|P| < r''$, compute v_{opt} by evaluating all candidate solutions in $cansol(i)$ and solve i using v_{opt}; else, run

the algorithm described in Theorem 4 relative to a constructed instance i' of COHERENCE consisting of $|P|$ copies of N and the given $O(\log_2 |P|)/d_H$ s-a-approx algorithm. As $d_H(c', c'_{opt}) \leq r' \log_2 |P'| < |P| =$ the number of copies of N in i' when $|P| \geq r''$, this `else`-phase will work correctly.

In the algorithm described above, the `else`-phase runs in time polynomial in $|i|$. Though the `if`-phase requires time exponential in r'', as r'' is a constant, this runtime reduces to a constant, and the algorithm as a whole runs in time polynomial in $|i|$. As COHERENCE is $\mathcal{NP}$-hard, this implies that $\mathcal{P} = \mathcal{NP}$, completing the proof. ∎

Given the above, one may wonder if COHERENCE is additive structure-approximable under *any* distance-bound that is sublinear in the input size. Using the instance-copy strategy, we can even rule out this possibility. To do this, we first define the following useful property of optimization problems.

Definition 6. (Adapted from Definition 18, Hamilton et al. (2007)) Given an optimization problem Π, a sd-function d, and a non-decreasing function $h : \mathcal{N} \to \mathcal{N}$ that is computable in polynomial time, Π is **(polynomial-time) $h(|i|)$-self-paddable with respect to** d if the following holds:

1. There exists a function $enc(i, h(|i|))$ such that for any instance i of Π, enc creates an instance i' of Π of size $padsize(i, h(|i|))$.
2. There exists a function $dec(enc(i, h(|i|)), c)$ such that for any instance i of Π, dec extracts from a solution $c \in cansol(enc(i, h(|i|)))$ the set $\{c_1, c_2, \ldots, c_{h(|i|)}\}$ such that for $1 \leq j \leq h(|i|)$, $c_j \in cansol(i)$;
3. enc and dec run in time polynomial in $|i|$ and $h(|i|)$; and
4. $\sum_{j=1}^{h(|i|)} d(c_j, optsol(i)) \leq d(c, optsol(enc(i, h(|i|))))$

If the definition above looks familiar, it should—self-paddability formalizes the four requirements described earlier that allow the the instance-copy approach to be applied to an optimization problem to prove various degrees of structure-inapproximability. The first three requirements in the definition above correspond directly to the first three requirements described earlier. Though the relationship between the fourth requirements may initially seem cryptic, the following theorem establishes the connection.

Theorem 7. *(Adapted from Theorem 19, Hamilton et al. (2007)) Given an $\mathcal{NP}$ optimization problem Π that is $h(|i|)$-self-paddable for a function h that is polynomially bounded, a sd-function d, and function $g : \mathcal{N} \to \mathcal{N}$ such that $g(padsize(i, h(|i|))) < h(|i|)$, if Π is $g(|i|)/d$-s-approximable and Π is $\mathcal{NP}$-hard then $\mathcal{P} = \mathcal{NP}$.*

Proof: Given an instance i of Π, we can run the following algorithm to solve i: Run the $g(|i|)/d$ s-a-approx algorithm on the instance $enc(i, h(|i|))$ of Π to create solution y. Decompose y into $\{y_1, y_2 \ldots, y_{(h(|i|))}\}$ using dec, and determine, for $1 \leq j \leq h(|i|)$, which y_j are in $optsol(i)$, i.e., compute all $v(y_j)$. As $\sum_{j=1}^{h(|i|)} d(y_j, optsol(i) \leq d(y, optsol(enc(i, h(|i|))))$ and $d(y, optsol(enc(i, h(|i|)))) \leq g(|enc(i, h(|i))|) = g(padsize(i, h(|i|))) < h(i)$, at least one y_j has $d(y_j, optsol(i))) = 0$, meaning that $y_j \in optsol(i)$. We can then use this y_j to solve i. As all steps of this algorithm run in time polynomial in $|i|$, this is a polynomial-time algorithm for Π; however, as Π is $\mathcal{NP}$-hard, this implies that $\mathcal{P} = \mathcal{NP}$, completing the proof. ∎

This theorem makes it easy to prove structure-inapproximability relative to various functions g of the input size, and hence both generalizes and replaces g-specific constructions like those in Theorems 4 and 5.

Self-paddability becomes particularly useful if a problem is polynomially self-paddable such that the produced padded instances are compact, i.e., the size of the produced padded instance is the same as (the size of the original instance plus additional structure of at most constant size) $\times$ (the number of copies of the original instance in the padded instance).

Lemma 8. *(Adapted from Lemma 20, Hamilton et al. (2007)) Given an $\mathcal{NP}$ optimization problem Π and a sd-function d, if Π is $\mathcal{NP}$-hard and Π is $|i|^\alpha$-self-paddable for all $\alpha \in \mathcal{N}$, $\alpha \geq 1$ such that this padding is compact, i.e., $padsize(i, |i|^\alpha) = O(|i|^{\alpha+r})$ for some constant $r > 0$, then Π is not $|i|^{(1-\epsilon)}/d$ s-approximable for any $\epsilon > 0$ unless $\mathcal{P} = \mathcal{NP}$.*

Proof: Suppose there is an algorithm A that is a $g(|i|) = |i|^{(1-\epsilon)}/d$ s-a-approx algorithm for Π for some $\epsilon > 0$. As Π is $h(|i|) = |i|^\alpha$ self-paddable for any integer $\alpha > 1$ such that $padsize(|i|, |i|^\alpha) = O(|i|^{\alpha+r}) \leq r'|i|^{\alpha+r}$ for some constant $r > 0$, we can rewrite the $g(padsize(i, h(|i|))) < h(|i|)$ condition from Theorem 7 as follows:

$$
\begin{aligned}
r'(|i|^{(\alpha+r)})^{(1-\epsilon)} &< |i|^{\alpha} \\
log_2 r' + (\alpha + r)(1-\epsilon)\log_2 |i| &< \alpha \log_2 |i| \\
\frac{\log_2 r'}{\log_2 |i|} + (\alpha + r)(1-\epsilon) &< \alpha \\
(\alpha + r)(1-\epsilon) &< \alpha - \frac{\log_2 r'}{\log_2 |i|} \\
\alpha + r - \epsilon(\alpha + r) &< \alpha - \frac{\log_2 r'}{\log_2 |i|} \\
-\epsilon(\alpha + r) &< -(r + \frac{\log_2 r'}{\log_2 |i|}) \\
\epsilon(\alpha + r) &\geq r + \frac{\log_2 r'}{\log_2 |i|}
\end{aligned}
$$

As $r + \frac{\log_2 r'}{\log_2 |i|} \leq r + \frac{r'}{\log_2 |i|} \leq r + r'$ and $r + r' \leq \epsilon(\alpha + r)$ holds for any $\epsilon > 0$ when $\alpha = \frac{r+r'}{\epsilon}$, the result follows by the $\mathcal{NP}$-hardness of Π, and Theorem 7. ∎

Not every problem is guaranteed to satisfy these conditions. However, as the following shows, COHERENCE is one of those problems that does.

Lemma 9. *For all $\alpha \in \mathcal{N}$, $\alpha \geq 1$,* COHERENCE *is $|i|^{\alpha}$-self-paddable with respect to d_H such that $padsize(|i|, |i|^{\alpha}) = O(|i|^{\alpha+1})$.*

Proof: Let $enc(i, |i|^{\alpha})$ create an instance i' of COHERENCE consisting of $|i|^{\alpha}$ copies of i connected by gadgets as in Theorem 4; note that as each gadget adds at most 1 proposition for each copy of i in i', $|i'| = padsize(i, |i|^{\alpha}) = (|P| \times |P|^{\alpha}) + (|P|^{\alpha} - 1) \leq 2 \times |P|^{\alpha+1} = O(|i|^{\alpha+1})$. Let $dec(i', c)$ return the $|i|^{\alpha}$ candidate solutions $\{c_1, c_2, \ldots, c_{|i|^{\alpha}}\}$ for the copies of i encoded in i'. Both *enc* and *dec* obviously run in time polynomial in $|i|$; moreover, it is also obvious that $\sum_{j=1}^{|i|^{\alpha}} d_H(c_j, optsol(i)) \leq d_H(c, optsol(enc(i, |i|^{\alpha})))$ (with equality occurring in this expression when differences between c and the closest member of $optsol(enc(i, |i|^{\alpha}))$ do not occur in the copy-connection-gadget propositions x_j). ∎

Corollary 10. COHERENCE *is not $|i|^{(1-\epsilon)}/d_H$ s-a-approximable for any $\epsilon > 0$ (unless $\mathcal{P} = \mathcal{NP}$).*

Proof: Follows from the $\mathcal{NP}$-hardness of COHERENCE and Lemmas 8 and 9. ∎

As additive structure-approximation factors must be less than or equal to $d_{\max}(i)$ and $d_{H(max)}(i) \leq |P| \leq |i|$ for COHERENCE, Corollary 10 effectively rules out any type of sublinear additive structure-approximability for COHERENCE.

We have one multiplicative structure-approximability result for COHERENCE. This result follows from the next lemma.

Lemma 11. *For* COHERENCE, $d_{H(max)}(i) = 0.5 \times |P|$.

Proof: This result exploits the property of COHERENCE that if c with associated binary-vector b is an optimal solution for some instance i, then so is c' with associated binary-vector $\overline{b}$, i.e., the bit-complement vector of b. Given an arbitrary $c \in cansol(i)$ for the given instance i, we prove that $d_H(c, c_{opt}) \leq 0.5 \times |P|$ for some c_{opt} for i by contradiction: Assume there is a $c \in cansol(i)$ such that $d_H(c, c_{opt}) > 0.5 \times |P|$ for *every* $c_{opt} \in optsol(i)$. Let c'_{opt} be the closest such optimal solution to c. As $d_H(c, c'_{opt}) > 0.5 \times |P|$, this means that at least half of the bits of c and c'_{opt} are different. This in turn means that at least half of the bits of c and $\overline{c'_{opt}}$ are the same, such that $d_H(c, \overline{c'_{opt}}) \leq 0.5 \times |P|$. However, if c'_{opt} is optimal, so is $\overline{c'_{opt}}$, which is a contradiction. ∎

As by definition no candidate solution can be further away from an optimal solution than distance $d_{\max}(i)$, by Lemma 11, *any* guess (random or non-random) is a multiplicative structure-approximation for COHERENCE relative to factor $r = 0.5$.

Corollary 12. COHERENCE *is* $0.5/d_H$ *s-m-approximable.*

Arguably, approximating COHERENCE in this sense is much too weak to be of use for psychological explanation. To defend an intractable computational-level theory by claiming it is multiplicative structure-approximable, the degree of approximation must be much smaller that 0.5 (i.e., provided multiplicative approximations make for acceptable psychological explanations at all). Though our results do not rule out the existence of such an approximation, we do not know of its existence either.

6. Discussion

In this final section, we discuss several take-home messages that can be derived from our case-study. Our discussion will progress from specific implications for the COHERENCE model (Section 6.1) to implications for its variants and related problems (Sections 6.2 and 6.3) to general lessons for assessing the structure-inapproximability of other optimization models of cognition (Section 6.4).

6.1. Implications for the Coherence Model

In this paper, we have studied the structure-approximability of COHERENCE. A natural first question is how our results relate to the value-approximation result discussed by Thagard and Verbeurgt (1998). Thagard and Verbeurgt (1998) (see also Verbeurgt (1998)) proved that there exists a value-approximation algorithm for COHERENCE that is guaranteed to output a truth-assignment T that is within 87% of optimal, in other words $COH(T) > 0.87COH(T_{opt})$. Does this mean that this algorithm also automatically produces truth assignments that have 87% of the assigned truth values correct? No it does not. As we showed in Figure 2, a truth-assignment T can have arbitrarily close to optimal coherence value, yet be maximally dissimilar from the optimal truth assignment, T_{opt}. To be precise, the Hamming distance between T and T_{opt} can be as much as $0.5n$ (where $n = |P|$ is the number of propositions, which equals the number of truth-values $|T| = |T_{opt}|$). Recall from Lemma 11 that the reason that $0.5n$ (and not n) upperbounds the maximum Hamming distance to an optimal solution for COHERENCE is that there always exists two optimal solutions, viz., T_{opt} and its complement $T'_{opt} = \overline{T_{opt}}$ where the truth values are all reversed (this follows from the symmetry in the definition of the value function $COH(\cdot)$, such that $COH(T) = COH(\overline{T})$ for any truth assignment T). As a result, any random truth assignment T_{rand} will always be within $0.5n$ Hamming distance of at least one optimal truth assignment, T_{opt} or T'_{opt}, or some other optimal solutions. Thus, a value-approximation algorithm may indeed be a structure-approximation algorithm but it need not be a good one—indeed, as shown in Figure 2, the value-approximation algorithm may perform no better than a random guess in structurally approximating the optimal output of COHERENCE.

This raises the question if COHERENCE can be otherwise structurally approximated. We proved that no structure-approximation algorithm can exist that produces an output T such that T differs in at most some constant c

truth values from an optimal truth assignment (Theorem 4). Moreover, even if the difference need not be constant, but may grow logarithmically with the size of the set of propositions $|P|$ (Theorem 5) or even sublinear in the instance size (Corollary 10), it is impossible to so structurally approximate Coherence. This definitively rules out additive structure-approximability for Coherence, which is the type of approximation that seems most relevant for the purpose of defending intractable computational-level theories as approximate explanations in that the degree of approximation remains constant (or almost constant) for different input sizes.

Because of the special property that any random guess will be within $0.5n$ Hamming distance of an optimal solution, Coherence *is* structure-approximable up to a constant *multiplicative* factor of 0.5. This seemingly positive result, however, is an artifact of the fact that there is simply no room in the search space to be further from an optimal solution than $0.5n$ Hamming distance (as explained above), and hence gives a trivial and uninteresting form of structure-approximation for Coherence. It might be interesting if one could devise a structure-approximation algorithm that can approximate an optimal solution strictly closer than $0.5n$ Hamming distance, and our results so far do not exclude this possibility. However, we have not so far been able to prove the existence of such a structure-approximation algorithm either. The previously published results of Kumar and Sivakumar (1999) establish that for any problem (i.e., also Coherence), there exists a way of representing candidate solutions such that this problem cannot be approximated within distance smaller than 0.5 (unless $\mathcal{P} = \mathcal{NP}$). However, it is not obvious that the same holds relative to the way that we represented candidate solutions for Coherence in this paper (i.e., the representation of belief-assignments as binary-vectors).

The absence of well-behaved structure-approximability results for Coherence seems to pose a problem for the claim that even though cognizer may not compute exactly the optimal solutions to Coherence whenever they fixate their beliefs, they compute solutions that approximate the optimal solutions. So far there is no reason to belief that human cognizers can tractably approximate optimal solutions to Coherence anymore than they can tractably compute the optimal solutions themselves, the latter being $\mathcal{NP}$-hard.

6.2. Implications for Generalizations of the Coherence Model

The COHERENCE model whose structure-approximability we have studied in this paper is not the most general model of Coherence described in the literature. In COHERENCE, all propositions have equal a priori believability, but as argued by Thagard (2000), propositions describing direct observations (rather than hypothetical states) seem to have a degree of believability of their own. Thagard called this the 'data priority principle'. At least two different ways have been proposed to make the data priority principle operational in the Coherence model, leading to the following two generalizations (Thagard, 2000; van Rooij, 2003):

> FOUNDATIONAL COHERENCE
> *Input*: A belief network $N = (P = D \cup H, C, w)$ where P denotes a set of propositions encoding observations (or data) D and hypotheses H, $C = C^- \cup C^+ \subseteq P \times P$ denotes a set of positive and negative constraints such that $C^+ \cap C^- = \emptyset$, and $w : C \to \mathcal{R}^+$ is a function that associates a positive weight $w(p, q)$ with each constraint $(p, q) \in C$.
> *Output*: A truth assignment $T : P \to \{true, false\}$, such that $T(d) = true$ for each $d \in D$ and the *coherence value* $COH(T) = \sum_{(p,q)\in C^+, T(p)=T(q)} w(p, q) + \sum_{(p,q)\in C^-, T(p)\neq T(q)} w(p, q)$ is maximized.

> DISCRIMINATING COHERENCE
> *Input*: A belief network $N = (P = D \cup H, C, w, w_D)$ where P denotes a set of propositions encoding observations (or data) D and hypotheses H, $C = C^- \cup C^+ \subseteq P \times P$ denotes a set of positive and negative constraints such that $C^+ \cap C^- = \emptyset$, $w : C \to \mathcal{R}^+$ is a function that associates a positive weight $w_C(p, q)$ with each constraint $(p, q) \in C$, and $w_D : D \to \mathcal{R}$ is a function that associates a non-negative weight $w_D(d)$ with each observation $d \in D$.
> *Output*: A truth assignment $T : P \to \{true, false\}$, such that the *coherence value* $COH(T) = \sum_{(p,q)\in C^+, T(p)=T(q)} w_C(p, q) + \sum_{(p,q)\in C^-, T(p)\neq T(q)} w_C(p, q) + \sum_{d\in D, T(d)=true} w_D(d)$ is maximized.

Given that both DISCRIMINATING COHERENCE and FOUNDATIONAL COHERENCE include the COHERENCE model as a special case (viz., when $D = \emptyset$) approximating these optimization problems is at least as hard as approximating COHERENCE . Hence, all structure-inapproximability results that we

reported for Coherence also hold for Discriminating Coherence and Foundational Coherence. Structure-approximability results for Coherence, however, do not automatically also hold for these generalizations, as we do not have analogues of Lemma 11 relative to Discriminating Coherence and Foundational Coherence. Hence, we do not know if these more general optimization problems are even approximable in the weak sense that Coherence is, viz., the latter has a trivial 0.5-factor multiplicative structure-approximation algorithm.

6.3. Implications for Harmony Maximization

It is known that the problem of computing Coherence is closely related to the problem of settling on a pattern a of activations in a neural network that maximizes harmony (or energy), defined as

$$H(a) = \sum_i \sum_j a_i a_j w_{ij}$$

where a_i is the activation of node i in the neural network and w_{ij} is the weight on the connection between nodes i and j in the neural network (see also Thagard and Verbeurgt (1998); Verbeurgt (1998)). In fact, when the final activation pattern has activation levels of either $+1$ or -1 and the neural network is a Hopfield net, computing maximum harmony is equivalent to computing Coherence (see Appendix A).[9] This means that all structure-inapproximability and -approximability results that we reported in Section 5.2 also hold for the problem of Harmony Maximization in Hopfield networks. As per the discussion in Section 6.2, the structure-inapproximability results also generalize to Harmony Maximization when some nodes in the network are 'clamped' to a particular activation value, say, $+1$. This possibility can be used to model the general Coherence models whose associated problems are Discriminating Coherence and Foundational Coherence as Harmony Maximization problems as follows:

> **Foundational Coherence as Harmony Maximization:** Let $x = \langle N = (P = D \cup H, C, w)\rangle$ be an instance of Foundational

[9] Moreover, it has been shown by Schoch (2000, Lemma 1.1) that for any given Hopfield network, there always exists a maximum Harmony activation pattern in which each a_i is set to either -1 or $+1$ (See also Appendix A for a proof).

COHERENCE. Then we can construct a neural net $G_x = (V, E, w')$ with a node i for each proposition $p_i \in P$ and weights on the connections set to $w'_{ij} = w(p_i, p_j)$. Furthermore, for each node i in G_x representing a proposition $p_i \in D$, we 'clamp' the value of i, i.e., we set $a_i = +1$. Computing a maximum harmony activation pattern for network G_x is equivalent to computing the maximally coherent truth assignment for x, because any maximum harmony activation pattern $a : V \rightarrow \{-1, +1\}$ corresponds to a maximally coherent truth assignment $T : P \rightarrow \{true, false\}$, where $T(p_i) = true$ if and only if $a_i = +1$.

Discriminating Coherence as Harmony Maximization: Let $x = \langle N = (P\ D\ \cup\ H, C, w, w_D\rangle$ be an instance of DISCRIMINATING COHERENCE. Then we can construct a neural net $G_x = (V, E, w')$ with a node i for each proposition $p_i \in H\ \cup\ D$, and weights on the connections set to $w'_{ij} = w(p_i, p_j)$. In addition, we include in G_x an extra node j for each proposition $p_i \in D$ and connect j to i by a connection with weight $w'_{ij} = w_D(p_i)$. Lastly, for each node i in G_x representing a proposition $p_i \in D$, we 'clamp' the value of i, i.e., we set $a_i = +1$. Computing a maximum Harmony activation pattern for network G_x is equivalent to computing the maximally coherent truth assignment for x, because any maximum harmony activation pattern $a : V \rightarrow \{-1, +1\}$ corresponds to a maximally coherent truth assignment $T : P \rightarrow \{true, false\}$, where $T(p_i) = true$ if and only if $a_i = +1$.

The constructions above imply that harmony maximization in neural networks with clamped nodes is as hard to compute, and as hard to structure-approximate, as the optimization problems COHERENCE, FOUNDATIONAL COHERENCE, and DISCRIMINATING COHERENCE. All of these intractability and structure-inapproximability results hold relative to the commonly-accepted conjecture that $\mathcal{P} \neq NP$. If one is willing to accept the slightly weaker conjecture that $\mathcal{NP} \neq$ co-$\mathcal{NP}$, slightly stronger results can be derived using results from Bruck and Goodman (1990)—namely, that even if the network convergence is allowed to take exponential rather than polynomial time, the networks cannot structure-approximate either harmony maximization or

the various Coherence problems.[10]

6.4. Implications for Assessing Approximability of Optimization Theories

Based on our case-study of COHERENCE and its variants, we are now in a position to also draw several lessons for computational-level modeling in general. For instance, we have shown that the ability to value-approximate an optimization function does not imply the ability to also structure-approximate that function. Also, we have shown that approximation is not always computationally easier than exact computation. Consequently, approximation claims in cognitive science raise the question of which notion of approximation such claims are tacitly assuming, as well as the question of whether or not such claims can be backed up by formal proofs of the relevant form of approximability. We discuss these issues in more detail below.

As we (and Millgram (2000)) have shown, value-approximability results for a computationally intractable model such as COHERENCE are not directly relevant if one wants to defend the model as a computational-level theory whose outputs may yet *structurally* approximate cognitive outcomes (unless those value-approximability results imply structure-approximability results as well, which in general is not the case). This holds for all computationally intractable computational-level theories in which it is the output's structure, rather than its value, that is the target of cognitive psychological description and explanation. In such theories, the hypothesis that cognitive processes optimize some value $v(s)$ is often used as a way of *explaining* observable cognitive judgments or behaviors, by noting that the observed cognitive behaviors seem to be implied by (or associated with) a cognitive structure s_{opt} that maximizes value over all candidate solutions s. The value function $v(\cdot)$ here is clearly a hypothetical construct, as this function itself is neither observed nor to be explained. It is rather the cognitive structure s, or direct correlates thereof, that is observed and demanding of explanation. This is perhaps most clearly illustrated in (rational) computational-level models of belief fixation, where it is the beliefs of cognizers that are to be predicted and

[10]The relevant result from Bruck and Goodman (1990) is that no neural network that is allowed exponential convergence time can solve an $\mathcal{NP}$-hard problem unless $\mathcal{NP}$ = co-$\mathcal{NP}$. If we can structure-approximate the Coherence problems using such networks, then by the proofs in Section 5.2, we can also solve the associated $\mathcal{NP}$-hard decision problems, which by Bruck and Goodman's result would imply that $\mathcal{NP}$ = co-$\mathcal{NP}$.

explained (i.e., the explanandum), and the assumption that cognizers fixate their beliefs so as to maximize conditional probability or explanatory coherence is made to explain and predict (i.e., the explanans) beliefs of human cognizers in different contexts.

Furthermore, our inapproximability results for COHERENCE illustrate that not every computationally intractable problem can be tractably approximated for every sense of 'approximation', and not even in arguably relevant senses of 'approximation'. Consequently, we believe it is vital for the validity of psychological explanations that cognitive scientists are careful not to make claims of approximability of their intractable optimization theories as long as definitions of 'approximation' and mathematical proofs of the relevant sense of approximation are missing. This point is worth emphasizing, as we know from our own research that intuitions about what makes an optimization function hard or easy to compute can be often mistaken (van Rooij et al., 2008), and the intuition that approximation is generally easier than optimization is no exception (Kwisthout et al., 2011). To be clear, we acknowledge that there may exist other intractable optimization theories than the ones we analyzed that are structure-approximable to high degrees, and these may possibly even include formalizations of one or more theories listed in Table 2. Our point is merely that whether or not this is so can only be determined if the proper mathematical analysis is done.

To assist cognitive scientists in this effort, in this paper we have presented and illustrated a general framework for assessing structure-approximability of computational-level theories. Our framework consists of both definitions and proof techniques. We illustrated the proof techniques in Section 5.2. All of our proofs build on a strategy called the 'instance-copy strategy', which can informally be sketched as follows: To prove that a particular optimization function Π is not tractably approximable within a distance d, assume the existence of a structure-approximation algorithm that returns an output which is strictly closer than distance d to the structure of the optimal structure. Now, imagine that for any given input of the optimization function Π it is possible to make multiple copies of the input *and* run A to structure-approximate the optimal output defined over the set of copies. If this were possible for more than $d + 1$ copies, then the existence of A would imply that one of the copies has been optimally solved, which in turn implies that the intractable optimization function Π is tractable. As this yields a contradiction, one can conclude that no such A can exist for this particular optimization function Π. Note that as the argument assumes nothing about

the nature of A but only its existence, the argument rules out *any* such algorithm.

Using the 'instance copy' strategy we have been able to show for one particular optimization function—as well as generalizations and equivalent functions—that it is impossible to structure-approximate it within an constant or sublinear additive distance from optimal. Weaker forms of approximation may yet be possible but, as noted by Kruschke (2010), for optimization functions to have explanatory import they should be 'good' rather than 'poor' approximations (and approximations whose 'error' grows faster than some constant or sublinear additive function of the input size seems to be rather 'poor' for purposes of psychological explanation). Is there reason to believe more optimization functions that figure in computational-level theories are similarly inapproximable? We think the answer is yes, for two reasons.

First, the dissociation between value and structure seems to be a more general property of optimization functions in computational-level theorizing. Consider, for instance, the example theories in Table 2 and observe that small changes in structure can have large impact on the value, and vice versa. For instance, deleting one central exemplar from a category and adding it to another category can have large impact on the total within-category similarity and between-category dissimilarity. The reverse is also possible. For instance, two percepts of close to equal simplicity, or two truth assignments that are close in probability, can yet be very different.

Second, many optimization problems have the property that candidate solutions that are close in structure to the optimal structure (i.e., structure approximations, as we defined them) can be tractably revised so as to become optimal solutions. This property we call neighborhood searchability (see Hamilton et al. (2007) for more details). This property seems to hold for many NP-hard problems (van Rooij et al., in press). As we have shown in Section 5.2, no problem that has this property can be structure-approximated within a constant additive distance. Moreover, we have shown that problems with the additional property of being self-paddable (see Definition 4 in Section 5.2 for details) cannot even be approximated within a sublinear additive distance.

In closing, we would like to make the following recommendations for a cognitive scientist interested in assessing the tractable approximability of a given optimization theory. A first step is to decide on which notion of 'approximation' is most relevant for ones purposes. If structure-approximation

is the relevant sense, then one can try to prove structure-inapproximability by either using our instance-copy strategy or otherwise show that the postulated optimization function has the properties of neighborhood searchability or self-paddability. Admittedly, our proof techniques (in tandem with those given in Feige et al. (2000)) so far only provide what Downey et al. (1999) coined the 'negative toolkit' of computational complexity theory. Future research may aim to also develop techniques that can make up the 'positive toolkit' of tractable structure-approximation algorithm design. It is our hope that cognitive modelers will rise to the challenge of applying our techniques to their own intractable optimization theories of cognition, and possibly develop new techniques along the way.

Acknowledgments: The authors would like to thank Johan Kwisthout, Cory Wright, Moritz Müller, and Matthew Hamilton for useful and inspiring discussions on the topic of this paper, and Renesa Nizamee and Antonina Kolokolova for bringing Kumar and Sivakumar (1999) and Feige et al. (2000) to our attention. The authors are also grateful to John Kruschke and two anonymous reviewers for comments that helped to improve this paper. Part of this research was supported by an NSERC Discovery Grant 228104 awarded to TW.

Appendix A. Coherence Maximization is Equivalent to Maximizing Harmony in Hopfield Networks

A Hopfield network can be represented by an edge-weighted complete graph $G = (V, E, w)$, where each edge $(v_i, v_j) \in E = V \times V$ has an associated weight $-1 \leq w((v_i, v_j)) = w_{ij} \leq 1$. An activation pattern in the network is an assignment of an activation weight $-1 \leq a_i \leq 1$ to every vertex $v_i \in V$. The Harmony $H(a)$ of an activation pattern $a : V \to [-1, +1]$ is defined as $H = \sum_i \sum_{j \neq i} a_i a_j w_{ij}$. Given this, we can state the computational task of maximizing harmony activation pattern for a given Hopfield network as the computation of the following input/output function:

Harmony Maximization
Input: A Hopfield network $G = (V, E, w)$.
Output: An activation pattern $a : V \to [-1, +1]$ such that $H(a) = \sum_i \sum_{j \neq i} a_i a_j w_{ij}$ is maximized.

Note that an algorithm solving HARMONY MAXIMIZATION need only consider the values -1 and $+1$ as possible activation values in constructing the harmony maximizing activation pattern a. The reason is that there always exist some $a : V \to \{-1, +1\}$, i.e., one that assigns vertices in V either the value -1 or the value $+1$, that maximizes H.

Lemma 13. *Given a Hopfield network $G = (V, E, w)$, there exists an activation pattern $a : V \to \{-1, +1\}$ that maximizes H.*

Proof: Assume activation pattern $a : V \to [-1, +1]$ with harmony $H(a)$ has at least one activation value a_i such that $-1 < a_i < +1$, i.e., an unrestricted activation value. We show that we can change the value of each unrestricted activation value to be in the set $\{-1, +1\}$ without decreasing the harmony associated with the pattern. Observe that the harmony of an activation pattern $H(a) = \sum_i \sum_{j \neq i} a_i a_j w_{ij}$ can be rewritten as $H(a) = \sum_i a_i \sum_{j \neq i} a_j w_{ij}$. Given a and an arbitrary unrestricted activation value a_i in a, create a new activation pattern a' by changing a_i as follows:

1. If $\sum_{j \neq i} a_j w_{ij} = 0$, set $a_i = +1$ or -1.

2. If $\sum_{j \neq i} a_j w_{ij} > 0$, set $a_i = +1$.

3. If $\sum_{j \neq i} a_j w_{ij} < 0$, Set $a_i = -1$.

Note that each change either results in a harmony $H(a')$ that is as high (case (1)) or higher (cases (2) and (3)) than $H(a)$; moreover, each change can be done in any order independently of the others until no unrestricted activation values remain in the pattern. ∎

This means we can restrict our attention to the following problem:

HARMONY MAXIMIZATION*
Input: A Hopfield network $G = (V, E, w)$.
Output: An activation pattern $a : V \to \{-1, +1\}$ such that $H = \sum_i \sum_{j \neq i} a_i a_j w_{ij}$ is maximized.

Note that every solution to HARMONY MAXIMIZATION* is also a solution for HARMONY MAXIMIZATION, but not necessarily vice versa.

We next show that HARMONY MAXIMIZATION* is just another way of describing COHERENCE, i.e., they do not constitute two different computational problems.

Lemma 14. *For any instance* $\langle N = (P, C, w)\rangle$ *of* Coherence*, there is a corresponding instance* $\langle G = (V, E, w)\rangle$ *of* Harmony Maximization* *such that (1)* N *and* G *are isomorphic and (2) for each truth assignment* T *for* N*, there is a corresponding activation pattern* a *for* G *such that* $H(a) = 2 \times COH(T) - O(1)$*, and vice versa.*

Proof: The proof of equivalence consists of several steps. First, we observe that we can recode every instance of Coherence as an instance of Harmony Maximization*, *and vice versa*, by using the translation key in the top half of Table A.4. Given these instances, encode a (possible) solution T to instance $\langle N = (P, C, w)\rangle$ of Coherence as a (possible) solution a to instance $\langle G = (V, E, w')\rangle$ of Harmony Maximization* as per the translation key in the bottom half of Table A.4. Observe that this direct mapping between truth assignments in T and activation values in a implies that a constraint $(p_i, p_j) \in C$ in N contributes positive value $w(p_i, p_j)$ to the overall coherence $Coh(T)$ (which occurs whenever either $(p_i, p_j) \in C^+$ and $T(p_i) = T(p_j)$ or $(p_i, p_j) \in C^-$ and $T(p_i) \neq T(p_j)$) if and only if the corresponding edge $(v_i, v_j) \in E$ in G contributes exactly the same positive value $w'_{ij} = w(p_i, p_j)$ to the

Table A.4: Translation key for instances and solutions of Coherence and Harmony Maximization*.

Translation	Coherence		Harmony Maximization*
Instance	$N = (P, C, w)$	$\leftrightarrow$	$G = (V, E, w')$
	$p_i \in P$	$\leftrightarrow$	$v_i \in V$
	$(v_i, v_j) \in E$	$\leftrightarrow$	$(v_i, v_j) \in E$
	$(p_i, p_j) \in C^+$ and $w_{ij} = k > 0$	$\leftrightarrow$	$w'_{ij} = k > 0$
	$(p_i, p_j) \in C^-$ and $w_{ij} = k > 0$	$\leftrightarrow$	$w'_{ij} = -k < 0$
Solution	$T(p_i) = true$	$\leftrightarrow$	$a(v_i) = +1$
	$T(p_i) = false$	$\leftrightarrow$	$a(v_i) = -1$

overall harmony $H(a)$ (which occurs whenever either $w'_{ij} > 0$ and $a(v_i) = a(v_j)$ or $w'_{ij} < 0$ and $a(v_i) \neq a(v_j)$). Also observe that a constraint $(p_i, p_j) \in C$ in N contributes zero value to the overall coherence $Coh(T)$ (which occurs whenever either $(p_i, p_j) \in C^+$ and $T(p_i) \neq T(p_j)$ or $(p_i, p_j) \in C^-$ and $T(p_i) = T(p_j)$) if and only if the corresponding edge $(v_i, v_j) \in E$ in G contributes

negative value $w'_{ij} = -w(p_i, p_j)$ to the overall harmony $H(a)$ (which occurs whenever either $w'_{ij} > 0$ and $a(v_i) \neq a(v_j)$ or $w'_{ij} < 0$ and $a(v_i) = a(v_j)$).

From the above, we conclude that a truth assignment T for N has value $Coh(T)$ if and only if the corresponding activation pattern a for G has harmony value $H(a) = Coh(T) - (\sum_{(p_i,p_j)\in C^-, T(p_i)=T(p_j)} w(p_i, p_j) + \sum_{(p_i,p_j)\in C^+, T(p_i)\neq T(p_j)} w(p_i, p_j))$. The two-summation term in this expression can be rewritten as

$$\begin{aligned}
&\sum_{(p_i,p_j)\in C^-, T(p_i)=T(p_j)} w(p_i, p_j) + \sum_{(p_i,p_j)\in C^+, T(p_i)\neq T(p_j)} w(p_i, p_j) \\
&= \sum_{(v_i,v_j)\in E} w'_{ij} - \Big(\sum_{(p_i,p_j)\in C^+, T(p_i)=T(p_j)} w(p_i, p_j) + \sum_{(p_i,p_j)\in C^-, T(p_i)\neq T(p_j)} w(p_i, p_j) \Big) \\
&= \sum_{(v_i,v_j)\in E} w'_{ij} - Coh(T)
\end{aligned}$$

Hence, $H(a) = Coh(T) - (\sum_{(v_i,v_j)\in E} w'_{ij} - Coh(T)) = 2 \times Coh(T) - \sum_{(v_i,v_j)\in E} w'_{ij}$. Since the term $\sum_{(v_i,v_j)\in E} w'_{ij}$ is constant for any given N and G, $H(a) = 2 \times Coh(T) - O(1)$, completing the proof. ∎

Observe that the translation-constructions in the proof above can be done in time polynomial in the size of the given instance, whether it is of Coherence or Harmony Maximization*. More importantly, in these constructed instances, a truth assignment T maximizes $Coh(T)$ if and only if its corresponding activation pattern a maximizes harmony $H(a)$.

References

Abdelbar, A., Hedetniemi, S., 1998. Approximation MAPs for belief networks is NP-hard and other theorems. Artificial Intelligence 102, 21–38.

Anderson, J., 1990. The adaptive character of thought. Lawrence Erlbaum Associates, Hillsdale, NJ.

Ausiello, G., Crescenzi, P., Gambosi, G., Kann, V., Marchetti-Spaccamela, A., Protasi, M., 1999. Complexity and Approximation: Combinatorial Optimization Problems and Their Approximability Properties. Springer, Berlin.

Baker, C., Saxe, R., Tenenbaum, J., 2009. Action understanding as inverse planning. Cognition 113, 329–349.

Blokpoel, M., Kwisthout, J., van der Weide, P. P., van Rooij, I., 2010. How Action Understanding can be Rational, Bayesian, *and* Tractable. In: Proceedings of the 32nd Annual Conference of the Cognitive Science Society. Cognitive Science Society, Austin, TX, pp. 1643–1648.

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

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

Chater, N., Hahn, U., 1997. Representational distortion, similarity, and the universal law of generalization. In: Proceedings of the Interdisciplinary Workshop on Similarity and Categorization (SimCat97). Department of Artificial Intelligence, University of Edinburgh, Edinburgh, pp. 31–36.

Chater, N., Oaksford, M., 1999. Ten years of the rational analysis of cognition. Trends in Cognitive Sciences 3, 57–65.

Chater, N., Oaksford, M., 2009. Précis of Bayesian rationality: The probabilistic approach to human reasoning. Behavioral and Brain Sciences 32, 69–120.

Chater, N., Oaksford, M., Nakisa, R., Redington, M., 2003. Fast, frugal and rational: How rational norms explain behavior. Organizational Behavior and Human Decision Processes 90, 63–86.

Chater, N., Tenenbaum, J., Yuille, A., 2006. Special issue: Probabilistic models in cognition. Trends in Cognitive Science 10 (7).

Downey, R., Fellows, M., Stege, U., 1999. Computational Tractability: The View from Mars. Bulletin of the EATCS 69, 73–97.

Feige, U., Langberg, M., Nissim, K., 2000. On the hardness of approximating $\mathcal{NP}$ witnesses. In: Proceedings of APPROX 2000: Third International Workshop on Approximationm Algorithms for Combinatorial Optimization. Vol. 1913 of Lecture Notes in Computer Science. Springer, pp. 120–131.

Fishburne, P., LaValle, I., 1996. Binary interactions and subset choice. European Journal of Operational Research 92, 182–192.

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, 379–397.

Garey, M., Johnson, D., 1979. Computers and Intractability: A Guide to the Theory of NP-Completeness. W.H. Freeman, San Francisco, CA.

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

Gigerenzer, G., Hoffrage, U., Goldstein, D., 2008. Fast and frugal heuristics are plausible models of cognition: Reply to Dougherty, Franco-Watkins, and Thomas. Psychological Review 115, 1230–1239.

Hahn, U., Chater, N., Richardson, L., 2003. Similarity as transformation. Cognition 87, 1–32.

Hamilton, M., Müller, M., van Rooij, I., Wareham, T., 2007. Approximating solution structure. In: Demaine, E., Gutin, G., Marx, D., Stege, U. (Eds.), Dagstuhl Seminar Proceedings no. 07281: Structure Theory and FPT Algorithmics for Graphs, Digraphs, and Hypergraphs. No. 07281 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fur Informatik (IBFI), Schloss Dagstuhl, Germany.

Imai, S., 1977. Pattern similarity and cognitive transformations. Acta Psychologica 41, 433–447.

Koffka, K., 1935. Principles of Gestalt Psychology. Harcourt Brace, New York, NY.

Körding, K., 2007. Decision theory: What "should" the nervous system do? Science 318, 606–610.

Kruschke, J. K., 2010. Bridging levels of analysis: comment on McClelland et al. and Griffiths et al. Trends in Cognitive Sciences 14 (8), 344–345.

Kumar, R., Sivakumar, D., 1999. Proofs, Codes, and Polynomial-time Reducibilities. In: Proceedings of the 14th IEEE Conference on Computational Complexity. IEEE Press, Los Alamitos, CA, pp. 46–53.

Kwisthout, J., van Rooij, I., 2012. Bridging the gap between theory and practice of approximate Bayesian inference. In: Proceedings of the 11th International Conference on Cognitive Modeling. TU Berlin, pp. 199–204.

Kwisthout, J., Wareham, T., van Rooij, I., 2011. Bayesian intractability is not an ailment that approximation can cure. Cognitive Science 35 (5), 779–784.

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

Love, B., 2000. A computational level theory of similarity. In: Proceedings of the 22nd Annual Conference of the Cognitive Science Society. Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 316–321.

Luce, R., Raiffa, H., 1956. Games and Decisions: Introduction and Critical Survey. Wiley, New York, NY.

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

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

Müller, M., van Rooij, I., Wareham, T., 2009. Similarity as tractable transformation. In: Proceedings of the 31st Annual Conference of the Cognitive Science Society. Cognitive Science Society, Austin, TX, pp. 49–55.

Pothos, E., Chater, N., 2001. Category learning without labels – a simplicity approach. In: Proceedings of the 23rd Annual Conference of the Cognitive Science Society. Lawrence Erlbaum Associates, Hillsdale, NJ, pp. 774–779.

Pothos, E., Chater, N., 2002. A simplicity principle in unsupervised human categorization. Cognitive Science 26, 303–343.

Rosch, E., 1973. On the internal structure of perceptual and semantic categories. In: Moore, T. (Ed.), Cognitive Development and the Acquisition of Language. Academic Press, New York, NY.

Rosch, E., Mervis, C., 1975. Family resemblances: Studies in the internal structure of categories. Cognitive Psychology 7, 573–605.

Sanborn, A. N., Griffiths, T. L., Navarro, D. J., 2010. Rational approximations to rational models: Alternative algorithms for category learning. Psychological Review 117 (4), 1144–1167.

Schoch, D., 2000. A fuzzy measure of explanatory coherence. Synthese 122, 291–311.

Simon, H., 1957. Models of Man: Social and Rational. John Wiley and Sons Inc., New York.

Thagard, P., 2000. Coherence in Thought and Action. The MIT Press, Boston, MA.

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

Trommershäuser, J., Maloney, L., Landy, M., 2009. The expected utility of movement. In: Glimcher, P., Camerer, C., Poldrack, R., Fehr, E. (Eds.), Neuroeconomics: Decision Making and the Brain. Elsevier Inc., Amsterdam, pp. 95–111.

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

van der Helm, P., 2004. Transparallel processing by hyperstrings. Proceedings of the National Academy of Sciences USA 101 (30), 10862–10867.

van der Helm, P., 2006. Dynamics of Gestalt psychology (Invited review of *perceptual dynamics: theoretical foundations and philosophical implications of gestalt psychology* by F. Sundqvist). Philosophical Psychology 19 (1), 274–279.

van der Helm, P., Leeuwenberg, E., 1996. Goodness of visual regularities: A nontransformational approach. Psychological Review 103, 429–456.

van Rooij, I., 2003. Tractable Cognition: Complexity Theory in Cognitive Psychology. Ph.D. thesis, Department of Psychology, University of Victoria.

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

van Rooij, I., Evans, P., Müller, M., Gedge, J., Wareham, T., 2008. Identifying sources of intractability in cognitive models: An illustration using analogical structure mapping. In: Proceedings of the 30th Annual Conference of the Cognitive Science Society. Cognitive Science Society, Austin, TX, pp. 915–920.

van Rooij, I., Stege, U., Kadlec, H., 2005. Sources of complexity in subset choice. Journal of Mathematical Psychology 49 (2), 160–187.

van Rooij, I., Wareham, T., 2008. Parameterized complexity in cognitive modeling: Foundations, applications and opportunities. Computer Journal 51 (3), 385–404.

van Rooij, I., Wright, C. D., Wareham, T., in press. Intractability and the use of heuristics in psychological explanation. Synthese.

Veale, T., Keane, M., 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. Morgan Kaufmann, pp. 232–237.

Verbeurgt, K., 1998. Approximation of some AI Problems. Ph.D. thesis, Department of Computer Science, University of Waterloo.