The Computer Journal Advance Access published October 18, 2007

doi:10.1093/comjnl/bxm034

Parameterized Complexity in Cognitive Modeling: Foundations, Applications and Opportunities

Iris van Rooij[1,*] and Todd Wareham[2]

[1]Nijmegen Institute for Cognition and Information, Radboud University Nijmegen, Nijmegen, The Netherlands
[2]Department of Computer Science, Memorial University of Newfoundland, St. John's, NL, Canada
**Corresponding author: i.vanrooij@nici.ru.nl*

In cognitive science, natural cognitive processes are generally conceptualized as computational processes: they serve to transform sensory and mental inputs into mental and action outputs. At the highest level of abstraction, computational models of cognitive processes aim at specifying the computational problem computed by the process under study. Because computational problems are realistic cognitive models only insofar as they can plausibly be computed by the human brain given its limited resources for computation, computational tractability provides a useful constraint on cognitive models. In this paper, we consider the particular benefits of the parameterized complexity framework for identifying sources of intractability in cognitive models. We review existing applications of the parameterized framework to this end in the domains of perception, action and higher cognition. We further identify important opportunities and challenges for future research. These include the development of new methods for complexity analyses specifically tailored to the reverse engineering perspective underlying cognitive science.

Keywords: cognitive modeling; parameterized complexity

Received 10 June 2006; revised 20 September 2006

1. INTRODUCTION

Computational complexity theory finds application in many different domains of scientific enquiry. Probably best known are its uses for the analysis and design of *algorithms* for the analysis and interpretation of scientific data. In this role, the theory also finds application in sciences like biology, psychology, sociology and economics. Much less known, however, is an altogether different application of complexity theory in science, *viz.*, as a tool for the analysis and design of *scientific models* of natural computing systems such as genes, cells, neurons, brains, humans, social groups and markets. In this paper, we take a closer look at one such domain of application: the modeling of human cognitive processes.

Computational complexity analyses can aid the design of cognitive models by identifying model aspects that are computationally unrealistic or implausible. Existing approaches, based predominately on the theory of *NP*-completeness [1], have come some way in achieving this modeling support, but these approaches remain limited to date in their import and utility for cognitive modelers. One reason for this is that the classical framework is not sensitive to the relative impact of different model aspects and parameters, as it aggregates a computation's complexity in one big parameter, *viz.*, *input size*. The theory of parameterized complexity, developed by Downey and Fellows [2], overcomes this problem by allowing complexity analyses to be performed relative to different model parameters.

In this paper, we highlight the inherent utility of parameterized complexity analysis for modeling natural cognitive systems. We review a set of existing applications in specific cognitive subdomains and we identify a wealth of cognitive subdomains which could similarly benefit from systematic parameterized complexity analyses. Besides these opportunities for parameterized complexity analyses, we also identify some important methodological challenges that suggest several new research directions within parameterized complexity theory.

1.1. Overview

We start by explaining the modeling goals of cognitive science and how computational complexity analyses of cognitive models can aid in the achievement of these goals (Section 2). Section 3 details the analytic process underlying computational complexity analyses in cognitive science and contrasts classical and parameterized approaches to implementing this process. We will argue that the parameterized approach has much to offer over and above the classical approach. In Section 4, we review existing applications of parameterized complexity theory in cognitive science, considering different cognitive subdomains. In Section 5, we present a list of opportunities for new avenues of research. Finally, in Section 6, we conclude with how the research program described in this paper can benefit both cognitive and computer science.

2. COGNITIVE MODELING AND TRACTABILITY

Studying and modeling human cognitive processes is the subject matter of cognitive science and psychology. To explicate and situate explanatory practices in contemporary cognitive science, we describe the widely adopted framework for cognitive theory formation proposed by David Marr (Section 2.1). We also explain how the notion of computational tractability can help in the process of designing cognitive models (Section 2.2). As will become clear from our discussion, cognitive science is engaged in a form of reverse engineering, to be contrasted with the forward engineering that typically occurs in computer science. Because tractability plays a different role in reverse and forward engineering, we devote Section 2.3 to laying out the main differences between the two types of engineering.

2.1. The Marr hierarchy

Cognitive science aims at achieving a computational understanding of human cognitive capacities and their underlying processes. Cognitive modelers often attempt to explain a cognitive capacity (e.g. the ability to understand language) by decomposing it into several sub-capacities (e.g. the capacity to parse sentences, the capacity to recognize letters and words, etc.). The coordinated manifestation of the sub-capacities is then believed to amount to the realization of the analyzed super-capacity [3]. Each hypothesized sub-capacity again calls for its own computational understanding.

A common assumption is that the explanations of cognitive (sub-)capacities can in principle be formulated at three different levels, which Marr [4] called the computational level, the algorithmic level and the implementational level (cf. [5]). At the highest level of abstraction, the *computational level*, the model specifies the input/output function assumed to be computed by the processes under study. The human capacity for vision may serve as an example: the visual system takes as *input* a 2-D representation of a visual scene as it is projected on the retina, and it gives as *output*, among other things, a 3-D interpretation of the visual scene as we experience it before our mind's eye. A computational-level model of this process should provide a description of the input and the output, and it should posit a functional mapping from input to output that explains why our perception of 2-D projections is the way it is. Although infinitely many mappings may exist, the goal is to identify the input/output function that describes the mapping realized by human vision.

In general, formulating a *computational level* theory of capacity ψ consists of hypothesizing a domain of inputs on which the capacity operates, $I_T = \{i_1, i_2, \ldots\}$, a relevant range of outputs, $O_T = \{o_1, o_2, \ldots\}$ and a function, $\psi_T: I_T \rightarrow O_T$, mapping each input $i \in I_T$ to an output $o = \psi_T(i)$ (see also [6, p. 381]). The function ψ_T instantiates a *veridical* computational level description of capacity $\psi: I \rightarrow O$ if and only if $I = I_{\mathrm{T}}$ and $\psi_T(i) = \psi(i)$ for all $i \in I$. Assuming that ψ_T is veridical at the computational level, further attempts can be made to understand the effective procedure by which $\psi_T = \psi$ is computed (i.e. the *algorithmic level* theory), as well as, to understand how that procedure is physical implemented by neural, or other bodily, processes (i.e. the *implementational level* theory).

Marr's typology serves as a useful scheme for classifying different explanatory attempts in cognitive science. Contemporary cognitive theories are typically situated at the computational and algorithmic level, with few, if any, giving concrete accounts of the implementational level. Of primary interest in this paper are computational-level theories. These theories should look very familiar to computer scientists as they effectively instantiate computational problems. This means that they lend themselves directly to computability and complexity analyses. We will next explain how such analyses can help cognitive scientists in their attempt to explain human cognition.

2.2. The tractability constraint

A first step toward determining which input/output functions accurately model cognitive capacities is distinguishing functions that are *possible* for natural cognitive systems to compute from those that are *impossible*. Recall that with every computational-level theory ψ_T that a cognitive scientist posits she is postulating that the modeled process ψ instantiates an effective computation of the problem ψ_T (viz. ψ has an algorithmic-level explanation). Moreover, on the widespread assumption that cognitive processes are physically instantiated (e.g. in the human brain), she is also postulating that the computation is being realized using a limited amount of physical resources (viz. ψ has an implementational-level explanation). But then, if ψ_T is found not to be computable using a realistic amount of computational resources (i.e. the function is computationally *intractable*) then ψ_T cannot possibly be veridical. This is what we call the *tractability constraint* on computational-level theories (see Fig. 1).

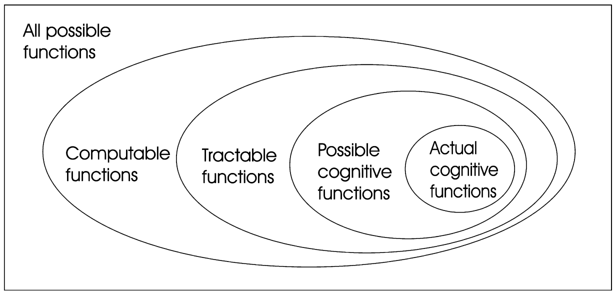

FIGURE 1. The tractability constraint provides a means of identifying non-veridical models of cognitive capacities.

TABLE 1. Traditional forward vs. cognitive reverse engineering

	Engineering-type	
Characteristic	Traditional	Cognitive
Origin of input/ output function	Derived from user specifications	Hypothesized on the basis of observations and intuition
Goals of input/ output function	Should meet user's needs	Should be a veridical model of cognitive capacity
Consequence of intractability	Devise heuristic methods	Revise hypothesized function

The study of computational (in)tractability is the subject matter of computational complexity theory. Using the concepts and methods of this mathematical theory, cognitive scientists can assess the computational (in)tractability of their computational-level theories. When formalizing the tractability constraint, care must be taken to ensure that it does not accidentally exclude possible cognitive functions.[1] If intractability is properly defined, and complexity results are interpreted with care, the tractability constraint can both delimit the space of viable computational-level theories and expose unrealistic accounts of human cognitive functioning, allowing faster convergence on veridical models of human cognitive capacities.

2.3. Cognitive modeling as reverse (not forward) engineering

Computational cognitive modeling is often seen as a form of *reverse* engineering [7], to be contrasted with traditional (forward) engineering. It is our experience that confounding the two types of engineering can lead to unproductive collaborations and miscommunication between cognitive modelers and complexity theorists. To clarify the practice of cognitive reverse engineering and its unique problems and issues, we next explicate the most important differences between the two types of engineering and their consequences for the use and interpretation of intractability results (see Table 1).

2.3.1. *User-specification vs. discovery*

In the case of forward engineering, a user specifies which input/output transformations he would like the software engineer to implement in a to-be-designed system. In reverse engineering, on the other hand, the input/output specifications of the to-be-modeled system need to be *discovered* by systematically observing the behavior of an already existing, naturally evolved system. In this case, the modeler builds on her intuitions about which input/output functions may capture the behavior of the system under study. As an example, consider again the problem of vision: based on observations of how humans interpret 2-D visual images, a cognitive modeler may come to form the hypothesis that vision interprets images according to the *simplest decomposition* of that image [8–10]. To further work out this hypothesis and build it into a well-defined computational level theory, the modeler would subsequently need to formally define what she means by 'simplest decomposition'.[2]

A pervasive problem faced by a reverse engineer is that her hypotheses about a system's input/output function may be wrong without the observational data telling her so. This is called the *empirical underdetermination* problem (see, e.g. [14–16]). It arises in cognitive science for at least two reasons. First, any finite set of observed input/output pairs is consistent with infinitely many hypothesized input/output functions. Second, the inputs and outputs of interest are typically not directly observable, because they occur 'in the head' (we cannot directly observe thoughts, concepts, percepts, plans, etc.). The empirical underdetermination problem is unique to reverse engineering and makes it in many respects harder than forward engineering. It is also the reason that the tractability constraint is such a useful theoretical tool for cognitive reverse engineers (see Fig. 1), while it is a mere nuisance for forward engineers.

2.3.2. *Dealing with intractability*

While the goal of the forward engineer is to compute as best as possible the user-specified input/output function, the reverse engineer aims at modeling as best as possible an existing system capacity. Consequently, function intractability signals something very different for forward and reverse engineers: for forward engineers it may mean that they should give up the hope of general exact procedures and instead settle for weaker heuristic methods [1], whereas for reverse engineers it means there must be something wrong

[1] For this reason, the types of tractability discussed in this paper are less restrictive than those implicit in certain areas of current research, e.g. the focus on three-layer neural network architectures (see also discussion in Section 3.2)

[2] See [11] for an approach based on Kolmogorov complexity theory and [12, 13] for an approach based on structural information theory.

with their model, so they had better get back to the drawing board and revise it [6, 17].[3]

Model revision is complicated for the reverse engineer by the fact that an intractable model can be wrong in many different ways. For example, a model can be wrong in terms of the initial, informal hypothesis (e.g. it may be false that vision produces simplest interpretations of visual displays), in terms of the particular formalization chosen to describe the hypothesis (e.g. it may be false that an visual interpretation's simplicity is equal to its Kolmogorov complexity), or in terms of the hypothesized domain of inputs on which the capacity operates (e.g. not all logically possible 2-D images may be possible projections of real-world objects).

Although it is impossible to assess the veridicality of informal hypotheses directly, it is possible to get an idea of their viability by assessing the intractability (and hence non-veridicality) of different formalizations and input restrictions. For example, if the computation of 'simplest decomposition' remains intractable under all conceivable formalizations and all reasonable input restrictions, this suggests the simplicity hypothesis of human vision is on the wrong track. If a cognitive scientist discovers, however, that the hypothesis can be rendered tractable by making certain changes to the formalization and/or by imposing certain input restrictions then this information can be used to define a new, tractable model. To ensure tractability, this new model will be making extra assumptions that can lead to new and testable predictions. In this way, the tractability-motivated revision process provides yet another way—besides model exclusion (Fig. 1)—in which cognitive scientists can reduce the empirical indeterminacy of cognitive models.

3. COMPUTATIONAL COMPLEXITY ANALYSIS OF COGNITIVE MODELS

In the previous section, we explained the goals of cognitive modeling and how tractability analyses can help cognitive scientists achieve those goals. We now discuss complexity theoretic concepts and methods underlying the core analysis process (Section 3.1) as well as the strengths and weaknesses of classical and parameterized complexity approaches for implementing this process (Section 3.2).

3.1. Core analysis process

We start by explaining why a standard measure of computational complexity suffices for our purposes of tractability analysis (Section 3.1.1). We then lay out a formal description of the process of designing tractable cognitive models (Section 3.1.2). Finally, we introduce the notion of *Source of complexity* (SoC) and discuss the important role SoCs can play in the revision of intractable cognitive models (Section 3.1.3)

[3]We emphasize this point because some cognitive modelers have responded to intractability results by devising heuristic methods as algorithmic-level explanations [18, 19]. In our view, this approach confuses the reverse and forward engineering perspectives and makes for incompatible theories at the computational and algorithmic levels (see also [20]).

3.1.1. Efficiency as time-complexity

Informally, a computation is said to be intractable if it requires an unreasonable amount of computational resources. In this paper, we will restrict ourselves to the resource *time* and adopt asymptotic worst-case time complexity as our measure of computational efficiency. The asymptotic worst-case time-complexity $O(f(n))$ of an algorithm is expressed in terms of a function $f(n)$, which asymptotically upper-bounds the number of basic operations required to run the algorithm on inputs of size n when implemented on a Turing machine [1]. Using this measure, an algorithm can be defined as *efficient* if and only if that algorithm's time complexity satisfies a certain criterion (e.g. the time complexity function is a polynomial of the input size). An input/output function is then said to be *tractable* if and only if there exists at least one efficient algorithm for computing it.

Over the years, several criticisms have been raised against this conception of cognitive tractability phrased in terms of asymptotic worst-case time complexity (e.g. [21–24]). Chief among them is the charge that the assumptions underlying asymptotic worst-case time complexity are unrealistic for human cognitive systems, e.g. because

- the neural hardware underlying cognitive computation is fundamentally different from the Turing machine model underlying asymptotic worst-case time complexity; and
- cognitive inputs that occur in practice are of bounded size and need not even be those that trigger worst-case behavior in an algorithm.

These criticisms are based on misunderstandings of the nature of computational complexity analyses of cognitive models (see also [25–30]). With respect to the first criticism, for the purposes of tractability analysis, a measure of efficiency is justifiable provided that associated classifications of function (in)tractability are not specific to the computational architecture underlying that measure (e.g. Turing machines), but instead reflect the function's inherent complexity under a wide variety of computational architectures that include those of interest (e.g. neural systems). Courtesy of various inter-architecture simulation results, this is known to be true for certain types of asymptotic worst-case time tractability, e.g. polynomial time [31, 32].

The second criticism highlights the need to properly apply complexity frameworks. In this case, the fault is not so much with the framework as with badly-formulated problems. If the worst-case inputs to a proposed input/output function do not occur in practice, then the function as specified is, at best, an overgeneralization of the cognitive capacity that the function is supposed to be modeling (see also [33]). By properly introducing input constraints, a better fit between the model and the actual capacity may be achieved.

3.1.2. *The tractable-design cycle*

Once the notions of efficiency and tractability are defined, cognitive modelers can engage in what we call the tractable-design cycle (analogous to the well-known empirical cycle in which scientists revise hypotheses if they do not fit the data), encoded in the following steps (see also [28, 33]).

Step 1: Formulate initial version of cognitive function.
Step 2: Analyze computational complexity of cognitive function.
Step 3: If cognitive function is tractable, stop.
Step 4: Else, isolate SoCs in cognitive function, revise function in light of SoCs, and go to Step 2.

If in Step 3, the hypothesized function is found to be tractable, the tractable-design cycle stops. This does not yet mean that the model is veridical, but only that we do not yet have reason to believe it is non-veridical. The model must still be subjected to empirical tests. If the model fails one or more empirical tests, then the empirical cycle generates a new hypothesized function, and the tractable-design cycle starts all over again.

If one cannot demonstrate tractability in Step 3, then one may be able to demonstrate intractability by showing that the function (or, more commonly, a decision problem associated with that function) is hard or complete for some problem-class C enclosing the class T of tractable problems relative to a tractability-preserving reduction, assuming $C \neq T$. If intractability is indeed established, then Step 4 is to identify one or more SoCs in the function. We next explain what constitutes an SoC.

3.1.3. *Sources of complexity*

An SoC is composed of one or more aspects of an intractable problem, that when restricted, or otherwise changed, yield a new problem that is of lower complexity and possibly tractable. An SoC may characterize any part of the description of the given problem or of the underlying structures invoked in solving that problem. For example, consider the following problem:

PARSE-G
Input: A grammar g of type G, a string x and positive integers d and n.
Output: A parse tree for x relative to g that has depth at most d and at most n internal nodes.

Possible aspects of this problem include the grammar-type G, the length of string x, the maximum parse-tree depth d or the size of the set T_g^x of possible parse-trees for x relative to g. An aspect can correspond to a member of a set of alternative mechanisms (e.g. one of the grammar-types in the Chomsky hierarchy) and such aspects are changed by swapping in another (possibly simpler) member of that set (e.g. use finite-state instead of context-sensitive grammars). On the other hand, an aspect may be a variable describing a mechanism (e.g. one of the variables v_i of a function $f(v_1, v_2, \ldots, v_l)$, which computes the value of $|T_g^x|$) and such aspects are restricted by fixing or placing bounds on the values of these variables (e.g. valid parse trees can be of depth at most 5).

Given a set S of one or more aspects comprising a putative SoC of a problem ψ_T, any implementation of the tractable-design cycle specified in Section 3.1.2 must be able to assess the time complexity of the version of ψ_T in which S is restricted. This time complexity can then be used to determine if S is an SoC for ψ_T. Once an SoC in an intractable model has been identified, it naturally suggests a possible model revision.

3.2. Implementing the core analysis process

To date, the generic cognitive model derivation process described in Section 3.1 has been implemented relative to classical and parameterized complexity theory. The two resulting approaches share many things—for instance, both use asymptotic worst-case complexity to measure efficiency (see Section 3.1.1) and classical hardness and completeness results to establish the intractability of cognitive functions (see Section 3.1.2). In this section, we will focus on where these approaches differ—namely, their definitions of tractability and the extents to which they support SoC identification.

3.2.1. *Classical approach*

Almost all analyses of cognitive models done to date have been done relative to classical computational complexity. In this approach, tractability is defined as polynomial-time solvability (what van Rooij [17] termed the *P-Cognition thesis*; see also [6, 33, 34]), and putative SoCs are evaluated using conventional algorithmic techniques and hardness/completeness for classes such as *NP* and *PSPACE* relative to polynomial-time many-one reducibility.

A common criticism of polynomial-time computability is that it is an overly generous definition of tractability (e.g. [35, 36]), as it allows time complexities that grow unreasonably fast as a function of input size n (e.g. $O(n^{100})$). Oddly enough, a more accurate criticism is that it is too strict! This is particularly the case when tractability is used as a constraint on cognitive models: the notion of tractability that cognitive modelers adopt should delimit the whole set of possible cognitive functions (see Fig. 1) and not classify *any* potentially veridical functions as intractable, and thereby as non-veridical. Do we have reason to believe that human cognizers compute functions of non-polynomial complexity? Yes—many time complexity functions that are exponential in general become polynomial when certain aspects are bounded in value (e.g. when $k = 5$ and $m = O(\log_2 n)$, $O(n^k m) \Rightarrow O(n^5 m)$ and $O(2^m k^3 n) \Rightarrow O(k^3 n^2)$). As many real-world cognitive inputs appear to be characterized by such bounded aspects, cognitive functions need not be restricted to general polynomial-time computable functions.

Such a relaxed criterion of tractability, in which only specified aspects of small value can participate in non-polynomial terms in time complexities, can be handled within the classical approach to a degree: one verifies such tractability by giving an algorithm whose time complexity has the desired form, and proves intractability relative to versions of the problem in which the aspects of interest are of constant value (e.g. k-Colorability is NP-complete when $k = 3$ [1, Problem GT4]). Unfortunately, such intractability results are frequently hard to prove, leaving the tractability of problems relative to certain sets of aspects in limbo. Also, even if such results are derivable, they cannot distinguish between cases in which the aspects of interest are responsible for non-polynomial terms singly (e.g. $O(2^k n)$), or in collaboration (e.g. $O(n^k m)$) and hence cannot assess tractability when bounds are not constants (e.g. $k = O(\log_2 n)$).

3.2.2. *Parameterized approach*

The difficulties with the classical approach to cognitive model analysis noted above can be traced back to the fact that in classical theories of computational complexity the input size is treated as a single monolithic parameter. As a result, it is very difficult to isolate the contribution of an individual aspect of an intractable problem to the running times of algorithms for that problem, let alone prove that there is no algorithm for that problem whose non-polynomial behavior is expressed purely in terms of that aspect.

These difficulties disappear if the theory of parameterized complexity is used. In this theory, each problem has an explicitly two-part input $\langle k, x \rangle$, where k is called the parameter and x is called the main part. This leads to the following notion of tractability:

Definition 3.1. *A parameterized problem ψ with inputs of the form $\langle k, x\rangle$ is fixed-parameter tractable (FPT) if there is an algorithm for ψ that runs in time $f(k)n^{\alpha}$ for some function f and constant α. Let FPT be the class consisting of all parameterized problems that are FPT.*

Note that this directly encodes the desired notion of cognitive tractability described in the previous section (what van Rooij [17] termed the *FPT-cognition thesis*). In the remainder of this paper, let the parameterized problem generated by parameterizing problem ψ relative to aspect-set S be denoted by $\langle S\rangle$-ψ. A putative SoC for a problem ψ based on aspect-set S of that problem can now be assessed by proving whether or not parameterized problem $\langle S\rangle$-ψ is in FPT. Fixed-parameter (fp) tractability can be proved using various fp-tractable algorithm design techniques, and fp-intractability can be proved using either (i) hardness or completeness results for members of the W-hierarchy $= \{W[1], W[2], \ldots, W[P], W[SAT], \ldots, XP\}$ (see [2] for definitions of these classes and parameterized reducibilities) or (ii) proofs of non-inclusion in a class in the W-hierarchy. The latter is of particular interest in light of the following lemma.

Lemma 3.1 [30, *Lemma* 2.1.35]. *Given a set S of aspects of a problem ψ, if ψ is $\boldsymbol{C}$-hard for some complexity class $\boldsymbol{C}$ when the value of every aspect $s \in S$ is fixed, then the parameterized problem $\langle S\rangle$-ψ is not in XP unless $P = \boldsymbol{C}$.*

Note that as we are typically only interested in ruling out fp-tractability, hardness and non-inclusion results suffice, i.e. it is not necessary to prove completeness results. The latter such results, enabled by Lemma 3.1, are particularly powerful as many cognitive functions are NP-hard when various aspects are constants (see Section 4).

As noted above, parameterized complexity is particularly suited to the assessment of potential SoCs for a problem. While such results can be derived individually, it is sometimes useful to derive such results systematically. Such a *systematic parameterized analysis* [30], in which parameterized results for a problem are derived relative to all $2^{|S|} - 1$ non-trivial subsets of a specified set S of problem aspects, can expose the range of potential SoCs for a given aspect-set, and is also to a degree helpful in assessing the minimality of a given SoC. Such minimal SoCs may be representative of particular mechanisms responsible for problem intractability (see also [37]). Performing such systematic analyses is made easier by the following lemmas (see also [30] Section 2.2.3]).

Lemma 3.2. *Given sets $S \subseteq S'$ of aspects of a problem ψ, if parameterized problem $\langle S\rangle$-$\psi \in FPT$ then parameterized problem $\langle S'\rangle$-$\psi \in FPT$.*

Lemma 3.3. *Given sets $S \subseteq S'$ of aspects of a problem ψ, if parameterized problem $\langle S'\rangle$-$\psi \notin FPT$ unless $\boldsymbol{X}$ for some conjecture $\boldsymbol{X}$, e.g. $P = NP$, then parameterized problem $\langle S\rangle$-$\psi \notin$ FPT unless $\boldsymbol{X}$.*

These lemmas mean that a full systematic analysis can often be performed by deriving a fairly small core of results. The collection of results from a systematic parameterized analysis is often most profitably viewed in a table in which each aspect-set in the analysis has its own entry; by appropriately organizing such *parameterized intractability maps* [30], the regions of tractability and intractability as well as regions of aspect-sets whose complexity is still open are easily seen, making for good summaries of current results as well as directions for future research. Examples of such intractability maps can be found in Tables 2–6.

There is currently one drawback to using parameterized complexity for cognitive model analysis: parameterized complexity is still a relatively young theory and thus has a much smaller base of results for practitioners in cognitive science to draw on than more mature classical theories of computational complexity like NP-completeness. Given the evident benefits of the parameterized approach it will be worthwhile to try and overcome this problem. This may be achieved, among other things, by stimulating interest in the computer science community for analyzing the parameterized complexity of problems arising in cognitive science, and by making the insights that such analyses generate more widely-known in the cognitive science community. To this effect the remainder of this paper considers a set of current applications of parameterized complexity in cognitive science, as well as future opportunities.

4. APPLICATIONS

We consider applications in three general domains: perception, action planning and higher cognition. While perception serves to transform sensory stimulation (e.g. of the eyes, ears, nose, etc.) into internal representations (e.g. percepts), action planning serves to transform internal representations (e.g. goals, constraints, percepts) into motor actions (e.g. locomotion, reaching, grasping). Possibly intervening between these peripheral processes are the more central, so-called higher-cognitive, processes that serve to transform internal representations (e.g. percepts, concepts, assumptions, goals, values) into other internal representations (e.g. judgments, arguments, conclusions, plans, decisions).

4.1. Perception

As the biological wetware underlying the classical senses such as sight, hearing, smell, taste and touch are readily accessible for experiments and have been extensively investigated since the dawn of psychological science, perception problems are arguably some of the most constrained computational problems in cognitive science. Despite the variety of opportunities, parameterized work on perception to date has focused on a single area—language processing.

Language processing essentially relates observable surface forms to underlying lexical forms. Surface forms can be expressed in a variety of sensory modalities; though most languages use sound, some use vision (sign languages for the deaf) and touch (adapted sign languages for the blind and deaf). Lexical forms are the abstract representations of surface forms to which meaning is added typically by higher levels of language processing. For example, a particular auditory signal, instead of denoting the ringing of an alarm clock or a door closing, is given the internal representation /k^haet/ associated with small semi-domesticated felines that are typically harbingers of mischief.

Language processing encompasses both generation (of surface forms from given lexical forms) and recognition (of underlying lexical forms from given surface forms). Ristad [38] suggests that these processes be analyzed in terms of the following templates.

***T*-Encoding**
Input: Lexical form u, lexical-surface form relation mechanism M of type $\boldsymbol{T}$.
Output: Surface form s created by applying M to u.

***T*-Decoding**
Input: Surface form s, lexicon D, lexical-surface form relation mechanism M of type $\boldsymbol{T}$.
Output: Set of lexical forms U generated by D from which M can create s.

These templates can in turn be made into computational problems by specifying s, u, M, and D relative to the representations and mechanisms underlying a particular linguistic theory $\boldsymbol{T}$.

Many classical complexity analyses of these problems have been carried out over the last 20 years relative to a variety of linguistic theories: e.g. two-level morphology [25], simplified segmental grammars (SSG) [27, 38] and optimality theory [39]. Parameterized analyses have been done by Wareham [29, 30, 40, 41] relative to these theories as well as finite-state transducer (FST) rule systems and declarative phonology. These theories fall into two groups, based on the nature of the form relation mechanism.

(1) Rule-based: Lexical form related to surface form by a sequence of rules and intermediate forms.
(2) Constraint-based: Lexical and surface forms related by a set of constraints on valid lexical-surface form pairings.

The encoding and decoding problems for all theories analyzed by Wareham except SSG are formalized in terms of finite-state mechanisms—that is, lexical and surface forms are strings, lexicons are encoded as deterministic finite-state automata (DFA), constraints are encoded by DFA and FST rules are encoded by FST and the rule- and constraint-based form relation mechanisms are encoded as the intersection and composition of given sets of DFA and/or FST.[4]

Decision versions of all of the encoding and decoding problems in the five theories examined by Wareham are *NP*-complete [25, 30, 38]. This is not suprising as the best known algorithms for intersection and composition of n finite-state automata, each of which has at most m states, iterate the classical state Cartesian-product pairwise intersection and composition algorithms and thus require $O(m^n)$ time and space (see [30, Section 2.2.3] and references). However, there yet remains a widely held belief within the

[4]The *intersection* of two DFA (FST) A_1 and A_2 is the language (relation) consisting of all strings (string-pairs) accepted by A_1 and A_2, and the *composition* of two FST A_1 and A_2 is the relation consisting of all string-pairs (x, y) such that there exists a string z such that (x, z) is accepted by A_1 and (z, y) is accepted by A_2. Intersections of $n > 2$ DFA or FST is defined in terms of acceptance by all n given automata; composition of $n > 2$ FST assumes a specified order of the given FST and a corresponding sequence $z_1, z_2, \ldots, z_{(n-1)}$ of intermediate forms. Intersection over DFA and composition over FST are closed, i.e. can be accepted by DFA and FST, respectively; however, intersection over FST is closed only for certain restricted types of FST (see [42] for details).

computational linguistics community that the simplicity of finite-state mechanisms renders any finite-state system tractable ('the lure of the finite-state' [25]).

The resulting debate over the SoCs in finite-state natural language processing problems (see [21, 25] and references) can be summarized by considering grammar derivation, i.e. is a given string x in the language generated by a given grammar g? Observe that grammar derivation is effectively a decoding problem in which M consists of the single automaton corresponding to g, s corresponds to x, and the only possible lexical-form in D is of length 1, i.e. the grammar start-symbol. The fact that this problem is solvable in polynomial time relative to finite-state and context-free grammars [43] suggests that the number of automata in M and/or the length of the lexical and surface forms may be SoCs for the encoding and decoding problems. This in turn suggests parameterized analyses in terms of the following aspects.

- The size of the alphabet ($|\Sigma|$).
- Length of lexical form ($|u|$).
- Length of surface form ($|s|$).
- Number of automata in the form-relation mechanisms ($|M|$).
- Maximum number of states in an automaton in M (M includes lexicon D where appropriate) ($|Q|$).

Results for these and other theory-specific aspects have been laid out in a number of intractability maps [30, 41]. A representative map from this set is shown in Table 2 for the following problem.

FST-Encoding
Instance: A set M of FST, all of whose input and output alphabets are Σ, a composition-order O on these FST, and a string $u \in \Sigma^+$.
Question: Is there a string-sequence $\{s_0, s_1, \ldots, s_{|M|}\}$ with $s_0 = u$ and $s_i \in \Sigma^{|u|}$ for $1 \leq i \leq |M|$ such that for the ordering $\{m_1, m_2, \ldots, m_{|M|}\}$ of M under O and $1 \leq i \leq |M|$, s_{i-1}/s_i is accepted by m_i?

TABLE 2. The parameterized complexity of FST-Encoding (adapted from [41, Table 1(b)])

	Alphabet size $\|\Sigma\|$	
Parameter	Unbounded	Parameter
	NP-complete	$\notin XP$
$\|M\|$	$W[t]$-hard	$W[t]$-hard
$\|u\|$	$W[2]$-hard	*FPT*
$\|Q\|$	$W[2]$-hard	???
$\|M\|$, $\|u\|$	$W[1]$-hard	*FPT*
$\|M\|$, $\|Q\|$	*FPT*	*FPT*
$\|u\|$, $\|Q\|$	$W[2]$-hard	*FPT*
$\|M\|$, $\|u\|$, $\|Q\|$	*FPT*	*FPT*

Note: '$\notin XP$' should be read as '$\notin XP$ unless $P = NP$'

These maps show that parameterized problems based on almost all aspect combinations that are not supersets of those invoked by the classical automaton composition- and intersection-based algorithms ($\{|M|, |Q|\}$) or the trivial algorithm that enumerates and checks all possible solutions ($\{|\Sigma|, |u|, |s|\}$) are W-hard. This means that none of the previously proposed SoCs in the literature ($|M|$, $|u|$, $|s|$) are in fact SoCs. Moreover, by virtue of systematically delimiting possible SoCs relative to all sets of aspects characterizing the 'coarse' structure of finite-state form-relation mechanisms, these maps suggest that further searches for SoCs for these problems should focus on aspects encoding the 'fine-grain' structure of these mechanisms, such as restrictions of the forms of the automata underlying the rules and constraints themselves.

4.2. Action planning

Action planning problems relate the specifications of initial and final states of a system to sequences of valid actions by which the system can progress from the initial to the final state. There are many levels of action planning in cognitive systems, from low-level planning of body motions in different types of 2- and 3-D spaces all the way up to high-level planning of action-sequences linking abstract states. Parameterized results are currently known for both of these extremes.

4.2.1. *Motion planning*

Given a known obstacle-filled environment and start and goal positions in such an environment, motion planning denotes the task of finding a path from the start to the goal position that does not collide with any obstacles. The agent attempting these moves may have a particular body structure (e.g. head, torso and limbs), which may be augmented by extra attachments (e.g. suitcases being carried or a box-loaded trolley being pushed). This agent-structure has various points at which movement is possible in some known directions with some fixed degrees of freedom (e.g. joints in limbs and handles on suitcases). Movements in certain environments may involve several re-orientations of this structure (e.g. the contortions required for a baggage-laden traveler to pass through a series of subway turnstiles), complicating motion planning still further.

Motion planning of an arbitrarily shaped agent-structure within an arbitrarily shaped 2- or 3-D environment can be modeled by the following problem.

d-Dimensional Generalized Mover
(dD-GM, $d \in \{2, 3\}$)
Instance: A set O of obstacle polyhedra, a set P of polyhedra which are freely linked together at a set of linkage vertices V such that P has k degrees of freedom of movement, and initial and final positions p_I and p_F of P in d-dimensional Euclidean space.

Question: Is there a legal movement of P from p_I to p_F, i.e. is there a continuous sequence of translation and rotations of the polyhedra in P such that at each point in time, no polyhedron in P intersects any polyhedron in O and the polyhedra in P intersect themselves only at the linkage vertices in V?

Although this problem is solvable in polynomial time if the agent is modeled as a single rigid polyhedron [44], the general version is *PSPACE*-hard in both two [45] and three [44] dimensional environments. Quite aside from the fact that no actual environments are cluttered to arbitrary degrees, agents (even when augmented by attachments) typically have very restricted structures. For example, the movements of all joints in a hand can be approximated by on the order of 20 degrees of freedom and the human body can be approximated by 43 linked parts. It would be interesting to know if and in what manners motion planning becomes tractable relative to these and other restrictions.

The parameterized analysis of 3D-GM carried out by Cesati and Wareham [46] models these restrictions as follows. Let a maximally linked group of polyhedra in P be a *component*, and each polyhedron in P be a *part*. Define the following aspects for GM:

- the number of *components* of P (c);
- the maximum number of parts in any component in P (p^c) and the total number of parts in P (p^t);
- the maximum number of algebraic inequalities, i.e. planar surfaces, lines, curves, needed to define any part (s^p) or component (s^c) of P, or the total number of such inequalities needed to define P (s^t);
- the maximum number of linkage vertices on any part (v^p) or component (v^c) of P, or the total number of linkage vertices in P (v^t); and
- the maximum number of degrees of freedom of any part (k^p) or component (k^c) of P, or the total number of degrees of freedom of P ($k^t = k$).

It is known that $\langle k^t, p^t, s^t\rangle$-3D-GM is $W[SAT]$-hard [46] and as many other versions remain *PSPACE*- and *NP*-hard when various parameters are constants [44, 45, 47], their associated parameterized versions are even harder still. All of these results are summarized in the intractability map given in Table 3.

Despite the above, hope yet remains for tractability of motion planning in restricted 2-D and 3-D environments. Promising results have recently been derived for the Rush Hour puzzle, which asks if the members of a given set of axis-parallel rectangles in the integer 2-D grid can be moved either horizontally or vertically to assume specified final positions without colliding. This problem is *PSPACE*-complete, but versions in which the number of rectangles and the total number of moves are parameters are both in *FPT* [48]; this in turn offers hope that more psychologically realistic formulations of 2-D motion planning may themselves be tractable.

TABLE 3. The parameterized complexity of *k*D-GM (adapted from [46, Table 1])

Parameter	3D-GM	2D-GM
Components		
c	$\notin XP$	$\notin XP$
Parts		
p^c	$\notin XP$	$\notin XP$
p^t	$W[SAT]$-hard	???
Surfaces		
s^p	$\notin XP$	$\notin XP$
s^c	$\notin XP$	$\notin XP$
s^t	$W[SAT]$-hard	???
Degrees of freedom		
k^p	$\notin XP$	$\notin XP$
k^c	$\notin XP$	$\notin XP$
k^t	$W[SAT]$-hard	???

Note: '$\notin XP$' should be read as '$\notin XP$ unless $P = PSPACE$'

4.2.2. *General action-sequence planning*

Many everyday tasks can be viewed as instances of a general action-sequence planning problem. In this framework, given initial and desired states s and g and a set of possible state-to-state transformation operators, the goal is find a sequence of operators that transform s into g. For example, suppose one has a particular set of ingredients (the initial state) and wants to prepare a three-layer white chocolate cake (the desired state); given a fully equipped kitchen and knowledge of basic cooking processes such as mixing, chopping, stirring and turning on the oven (the set of possible state-to-state transformation operators), the goal is to find a recipe (a sequence of operators) for baking the wanted cake using a subset of the given ingredients. Note that, as in motion planning, such an operator-sequence may not exist for certain (s, g) pairs (e.g. try making chocolate cake from ground beef and peppers or silk purses from sow ears).

One possible way of representing states is as conjunctions of true or negated assertions (e.g. $heated(oven) \wedge mixed(flour, sugar) \wedge \overline{puree(watermelon)}$), where the assertions comprising a state are a subset of a base assertion-set. State-transformation operators can then be represented as precondition/postcondition pairs, where preconditions and postconditions are conjunctions of true or negated assertions (e.g. $measured(flour) \wedge measured(sugar) \wedge empty(bowl) \Rightarrow \overline{empty(bowl)} \wedge mixed(flour, sugar)$). An operator $pre \Rightarrow post$ is applicable to a state x if all assertions in pre are in x and have the same values; the application of such an operator adds and/or modifies the assertions in $post$ to x to create state x'.

The scheme described above is propositional STRIPS [49], a simplified version of the original STRIPS formalism proposed by Fikes and Nilsson in 1971 [50]. The propositional STRIPS planning problem can equivalently be stated [51] as follows in terms of vectors whose positions correspond to individual assertions in the base assertion-set:

> *k*-step Propositional STRIPS Planning (*k*PSP)
> *Input:* A positive integer n, an initial state vector $s \in \{0, 1\}^n$, a goal vector $g \in \{0, 1, *\}^n$, a collection $O = \{o_1, \ldots, o_m\}$ of operators of the form $o_o = (P, Q)$, $P, Q \in \{0, 1, *\}^n$ and a positive integer k.
> *Question:* Is there a sequence of operators of length at most k that, when applied in order to s, produce g?

In this version, n is the number of assertions, a vector corresponds to a subset of assertions and 1 (0) at position i in that vector corresponds to assertion i being true (negated). In an operator $o = (P, Q)$, vectors P and Q are called the preconditions and postconditions of the operator. An operator o can be applied to a state vector $s \in \{0, 1\}^n$ if for all i, $1 \leq i \leq n$, $P[i] \in \{0, 1\}$ implies that $s[i] = P[i]$. The application of o to s yields a state-vector s' such that for all i, $1 \leq i \leq n$, $s'[i] = s[i]$ if $Q[i] = *$ and $s'[i] = Q[i]$ otherwise. A state vector s' is equivalent to the goal vector g if for all i, $1 \leq i \leq n$, $g[i] \in \{0, 1\}$ implies $s'[i] = g[i]$.

This problem is *PSPACE*-complete even when the number of true and negated assertions in the precondition and postcondition set of each operator is at most 2 [49]. However, state-transformation operator-sets are not of arbitrary size and operator-sequences cannot be too long if they are to be useful in practice (e.g. almost all recipes are explainable within a 30 min cooking show format). Does restricting combinations of aspects render *k*PSP tractable? Let $|pre|$ and $|post|$ be the maximum number of pre- and post-conditions in any operator in O, i.e. the maximum number of non-star entries in the P and Q sets of any operator in O. It is known that $\langle k, |pre|, |post|\rangle$-*k*PSP is $W[1]$-hard when $|pre| = 4$ and $|post| = 7$ [51] and that $\langle k, m\rangle$-*k*PSP is in *FPT* (see if the application of any of the $O(m^k)$ possible application-sequences of operators to s produces g).

All of these results are summarized in the intractability map given in Table 4. At present, one SoC ($\{m, k\}$) has been isolated; others may be derivable with further investigations of open questions relative to the aspects examined to date or by generalizing those versions of *k*PSP known to be solvable in polynomial time [49].

TABLE 4. The parameterized complexity of *k*-step Propositional STRIPS

	Operation-set size m	
Parameter	Unbounded	Parameter
–	*PSPACE*-complete	???
$\|pre\|$	$\notin XP$	???
$\|post\|$	$\notin XP$	???
k	$W[1]$-hard	*FPT*
$\|pre\|, \|post\|$	$\notin XP$	???
$\|pre\|, k$	$W[1]$-hard	*FPT*
$\|post\|, k$	$W[1]$-hard	*FPT*
$\|pre\|, \|post\|$ k	$W[1]$-hard	*FPT*

Note: '$\notin XP$' should be read as '$\notin XP$ unless $P = PSPACE$'

4.3. Higher cognition

Because, by their very nature, the inputs and outputs of central cognitive processes are not directly observable by scientists (and often not even by cognizers themselves), models of higher cognition are even more susceptible to the empirical underdetermination problem discussed in Section 2.3.1. It seems then that the tractability constraint can really benefit models of higher cognition. Although many *NP*-hardness results for models of higher cognition have accumulated over time (e.g. [19, 35, 36, 52–57]), few researchers have taken up the opportunity to systematically investigate SoCs in these models using parameterized complexity theory. Notable exceptions can be found in the domains of decision making (in particular, Subset Choice [37]) and non-deductive reasoning (Coherence [17]). We discuss these applications below.

4.3.1. Subset choice

Subset choice denotes the task of choosing a subset of items A from a set of available items V. A common assumption in the psychology of decision making is that when a decision-maker is presented with the option of choosing a subset from V she will choose a subset $A \subseteq V$ with satisfactory (e.g. maximum) value. But this means that the decision-maker can determine a satisfactory valued subset for a given choice set, i.e. she can compute instances of the following computational problem:

> Generalized Subset Choice
> *Input*: A set $V = \{x_1, x_2, \ldots, x_n\}$ of n available items, a value function u: $2^V \to \mathcal{Z}$ assigning an integer value to every subset in V, and an integer p.
> *Question*: Does there exist a subset $A \subseteq V$ such that $u(A) \geq p$?

Clearly, Generalized Subset Choice is an implausible model of human subset choice for all but very small n, because the number of subsets that need to be considered grows exponentially with n. This problem can be overcome by assuming that the value of a subset $u(A)$ can be decomposed into the values of individual elements and their combinations. A first simple model of subset value $u(A)$ is given by the additive model

$$u(A) = \sum_{x \in A} u(x), \qquad (1)$$

where $u(x)$ denotes the value of an item $x \in A \subseteq V$. If we restrict the Generalized Subset Choice to value structures that satisfy equation (1), then the problem is linear-time computable, which surely meets the tractability constraint. The problem with the additive model, however, is that it does not take into account the possibility that items can interact so as to increase or decrease subset value. Therefore, Fishburn and LaValle [54, 58] proposed the binary model of subset value

$$u(A) = \sum_{x \in A} u(x) + \sum_{(x,y) \in A^2} \Delta(x, y), \qquad (2)$$

where $\Delta(x, y) = u(x, y) - u(x) - u(y)$. Although the binary model improves on the additive model it still assumes the absence of higher-order value interactions, an assumption that seems to be violated by human decision-makers. Therefore, van Rooij *et al.* [37] proposed a generalization of the binary model, called the h-ary model of subset value,

$$u(A) = \sum_{x \in A} u(x) + \sum_{(e) \in A^2 \cup A^3 \cup \cdots \cup A^n} \Delta(e), \qquad (3)$$

where for each h-tuple $(x_1, x_2, \ldots, x_h) \in A^h$, with $2 \leq h \leq n$, $\Delta(x_1,x_2,\ldots,x_h) = u(A) - \sum_{x \in A} u(x) - \sum_{e \in A^2 \cup A^3 \cup \cdots \cup A^{h-1}} \Delta(e)$. $2_{\cup A^3 \cup \cdots \cup A^{h-1}}\Delta(e)$. Since every possible value function u obeys equation (3), the h-ary model of subset choice is in its generality as intractable, and hence as implausible, as GENERALIZED SUBSET CHOICE. It is conceivable, however, that human value structures have special properties that render subset choice under the h-ary model tractable.

- Humans may have limited sensitivity for value differences: this poses lowerbounds and upperbounds on the different value levels. This constraint is modeled by four parameters $u_{\min}$, $u_{\max}$, $\Delta_{\min}$ and $\Delta_{\max}$, where $u_{\min} \leq u(x) \leq u_{\max}$ for all $x \in V$, and $\Delta_{\min} \leq \Delta(e) \leq \Delta_{\max}$ for all $e \in A^2 \cup A^3 \cup \cdots \cup A^n$.
- Humans may have limited sensitivity to higher order value-interactions: this poses an upperbound ϵ on the degree of non-zero interactions and is modeled by setting $\Delta(x_1, x_2, \ldots, x_h) = 0$ for all $h > \epsilon$.
- Humans may have low aspiration levels p or $q = p - u(V)$.

Fishburn and LaValle [54] proved that SUBSET CHOICE is *NP*-hard for $\epsilon = 2$ (i.e. the binary model). Van Rooij *et al.* [37] furthermore showed that *NP*-hardness remains even if $\epsilon = 2$, $u_{\min} = 0$, $u_{\max} = 1$, $\Delta_{\min} = -1$ and $\Delta_{\max} = 0$. With the same restriction the problem is $W[1]$-hard when parameterized by p, but in *FPT* when parameterized by q. The *FPT* result generalizes if we relax $\Delta_{\min} \leq -1$. If we instead relax $u_{\max} \geq 1$ the problem is $W[1]$-hard for q, but it is in *FPT* when parameterized by q and $u_{\max}$. More general, SUBSET CHOICE is in *FPT* for $\epsilon \geq 2$, $u_{\min} = 0$, $u_{\max} \geq 1$, $\Delta_{\min} = -1$ and $\Delta_{\max} = 0$, when parameterized by q, $u_{\max}$ and ϵ.

These results are summarized in the intractability map given in Table 5. Note that the results imply, among other things, that subset choice under the h-ary model is intractable even if people can distinguish at most two value levels for elements and interactions, can detect at most second-order interactions, and have a low aspiration level for the value of $u(A)$.

TABLE 5. The parameterized complexity of SUBSET CHOICE

	Aspiration level parameter		
Parameter set		p	q
–	*NP*-hard	$W[1]$-hard	$W[1]$-hard
ϵ	$\notin XP$	$W[1]$-hard	$W[1]$-hard
$u_{\min}$, $\Delta_{\min}$, $\Delta_{\max}$	$\notin XP$	$W[1]$-hard	W[1]-hard
ϵ, $u_{\min}$, $\Delta_{\min}$, $\Delta_{\max}$	$\notin XP$	$W[1]$-hard	???
$u_{\min}$, $u_{\max}$, $\Delta_{\min}$, $\Delta_{\max}$	$\notin XP$	$W[1]$-hard	???
ϵ, $u_{\min}$, $u_{\max}$, $\Delta_{\min}$, $\Delta_{\max}$	$\notin XP$	$W[1]$-hard	???

Note: '$\notin XP$' should be read as '$\notin XP$ unless $P = NP$'

4.3.2. Coherence

Humans possess an impressive capacity for making plausible non-deductive inferences, both in the common sense and scientific domain—a capacity with which artificial intelligence struggles to date [59–61]. For example, people can infer in a split second a person's intentions and desires from ambiguous utterances, behaviors and context variables[5] and scientists have proven ability to generate hypothetical constructs that help explain natural phenomena. It has been proposed in [18] that humans make such non-deductive inferences by means of maximizing the *coherence* of their beliefs with the available information. A first formalization of this intuition was provided by Thagard and Verbeugt [19] in a computational-level model called COHERENCE.

COHERENCE
Input: A graph $N = (P, C)$, with $C = C^- \cup C^+$ and $C^- \cap C^+ = \emptyset$, and positive weight $w(p, q) > 0$ for each $(p, q) \in C$.
Output: A partition of P into accepted A and rejected R vertices such that the total weight of satisfied constraints $\sum_{(p,\, q) \in S(p,\, q)} w(p, q)$ is maximized. Here the set of satisfied constraints is defined by $S(p, q) = \{(p, q) \in C^+ | p, q \in A \text{ or } p, q \in R\} \cup \{(p, q) \in C^- | p \in A \text{ and } q \in R\}$.

In this model, P denotes the set of representational elements (e.g. propositions) that can be accepted (e.g. believed to be true) or rejected (e.g. believed to be false) by the coherence-based inferential process. If two elements $p, q \in P$ fit together they are said to *cohere* (e.g. the belief that God exists coheres with the belief that there is life after death), which is modeled

[5]Consider, for example, the scenario where the bill arrives after a restaurant dinner and the person sitting in front of you says 'This is on me', while she moves her hand into her pocket.

by a positive constraint $(p, q) \in C^+$; if, on the other hand, p and q resist fitting together they are said to *incohere* (e.g. the belief in Darwinian evolution incoheres with the belief in God), which is modeled by a negative constraint $(p, q) \in C^-$.[6] A non-deductive inference (A, R) has maximum coherence if it maximally satisfies the given constraints.

Note how Coherence assumes that all elements in P have equal (viz., zero) *a priori* plausibility. This does not seem to fit with how humans think about the world. For example, propositions describing direct observations typically have an acceptability of their own (e.g. if one sees that it is raining, then one's belief that *it is raining* gains support directly from one's perception). Thagard [18] coined this *the data priority principle*. To incorporate this principle in the Coherence model, we may define a special set of data elements $D \subseteq P$ that are 'favored' to be accepted. Data priority could then operate in at least two different ways: either loosely, by weighting elements in D and counting the weight of a $d \in D$ toward the total coherence if $d \in A$; or strictly, by requiring that all $d \in D$ be assigned to A. The first option is called Discriminating Coherence and the second Foundational Coherence [17].

Discriminating Coherence
Input: A graph $N = (P, C)$. Here $P = D \cup H$ with $D \cap H = \emptyset$ and $C = C^- \cup C^+$ with $C^- \cap C^+ = \emptyset$. Each $d \in D$ has a weight $w_D(d) \geq 0$ and each $(p, q) \in C$ has a weight $w_C(p, q) > 0$.
Output: A partition of P into A and R such that $\sum_{(p,q) \in S(p,q)} w_C(p, q) + \sum_{d \in D \cap A} w_D(d)$ is maximized.

Foundational Coherence
Input: A graph $N = (P, C)$. Here $P = D \cup H$ with $D \cap H = \emptyset$ and $C = C^- \cup C^+$ with $C^- \cap C^+ = \emptyset$. Each $(p, q) \in C$ has a weight $w(p, q) > 0$.
Output: A partition of P into A and R such that $D \subseteq A$ and $\sum_{(p,q) \in S(p,q)} w(p, q)$ is maximized.

It is known that Coherence is *NP*-hard [19]. Since Coherence is a special case of both Discriminating Coherence and Foundational Coherence, the hardness result propagates to these generalizations as well. This means that, in their general form, all three models are unrealistic as models of human inference.

In search of tractable special cases of these models, we consider a set of natural parameters.

- The number of positive constraints, $|C^+|$, and the number of negative constraints, $|C^-|$.
- The maximum weight on constraints, $w_{C\max}$, and on data elements $w_{D\max}$.

[6]Elements can cohere or incohere for many different reasons. For example, p and q cohere if p implies q, p is associated with q, or p and q together explain r, and incohere if p and q are inconsistent, if p is associated with $\neg q$, p and q are competing explanations of r.

TABLE 6. The parameterized complexity of Coherence and Discriminating/Foundational Coherence

Parameter	Coherence	Discriminating / Foundational Coherence
–	*NP*-hard	*NP*-hard
$\lvert H\rvert$	–	???
$\lvert D\rvert$	–	$\notin XP$
$\lvert C^+\rvert$	$\notin XP$	$\notin XP$
$\lvert C^-\rvert$	*FPT*	???
$w_{D\max}$	$\notin XP$	$\notin XP$
$w_{C\max}$	$\notin XP$	$\notin XP$
c	*FPT*	???
i	*FPT*	???

Note: '$\notin XP$' should be read as '$\notin XP$ unless $P = NP$'

- The size of the set of data elements $|D|$ and the set of hypothetical elements $|H|$.

Because a cognizer may be content with making inferences of satisfactory—not necessarily maximum—coherence, we further consider the model variant that assumes the output is a partition for which the weight of satisfied constraints is at least c, and the model variant that assumes the weight of unsatisfied constraints is at most i. This leaves us with two further parameters.

- The positive integers c and i.

Are any combinations of the listed parameters SoCs for the coherence models? Van Rooij [17] reports results pertinent to this question: Discriminating Coherence and Foundational Coherence are found to be *NP*-hard even if $w_{C\max} = 1$, $|D| = 0$ (or, equivalently, $w_{D\max} = 0$) and $|C^+| = 0$. When $|D| = 0$, the models are in *FPT* when parameterized by $|C^-|$, $|H|$, c or i [17, 62]. These results are summarized in the intractability map given in Table 6.

5. OPPORTUNITIES

The most obvious opportunities for future research are to continue the analyses described in Section 4. Other opportunities exist as well, both in analyzing new problems and in developing new techniques within the parameterized framework to assist with these analyses. We discuss concrete examples of such opportunities in Sections 5.1 and 5.2.

5.1. New application areas

We identify several interesting open problems for cognitive models in the areas of perception (Section 5.1.1), action planning (Section 5.1.2) and higher cognition (Section 5.1.3). All models were previously examined within the classical complexity framework (and typically found to be *NP*-hard or worse), but may benefit from systematic SoC analysis supported by the parameterized complexity framework.

5.1.1. *Perception*

A prime candidate for a new application area is vision. Although many lower-level visual tasks have been rendered tractable (e.g. line/surface recognition [4]), there are higher-level visual tasks for which the best known cognitive models are intractable.

We already considered the problem of form perception as a running example in Section 2. Two classes of models have been put forth to explain form perception: those based on the *likelihood* principle and those based on the *simplicity* principle [9].[7] Although both types of models have been plagued by intractability results, we here consider only a specific model of the latter type:

Generalized Minimum Encoding
Input: A string s and an encoding relation $E : S \rightarrow C$ mapping strings to sets of codes.
Output: A code $c \in E(s)$ such that $|c|$ is minimized.

Here s is a structural encoding of a retinal image and c models the interpretation of the image. If the relation E maps s to Turing machines that, when run on the empty string, produce s, then Generalized Minimum Encoding is equivalent to the computation of Kolmogorov complexity and therefore uncomputable [63]. Arguably, the visual system poses limits on E that may render Generalized Minimum Encoding tractable. Leeuwenberg and van der Helm [12, 13, 64], for example, argued that the visual system may encode visual images using a combination of ISA-rules (i.e. where ISA stands for Iteration, Symmetry, Alternation). Under this encoding scheme, the problem has been shown to be computable in subexponential time [13]. It is of interest to consider also other encoding relations and investigate the relative contributions of natural parameters such as $|s|$, $|c|$, $|E|$ and parameters specific to the adopted E, to the problem's complexity.

Another ubiquitous visual problem is that of finding a target object in a visual scene (e.g. finding your own car in the parking lot, or finding a familiar face in a crowd), called visual search [65, 66]. Successful execution of the visual search task requires vision to solve a subtask called visual matching, which is the task of recognizing that a given part of the visual field corresponds to the sought after target. A computational-level model of this subtask was proposed by Tsotsos [33, 67–69].

Bottom-up Visual Matching
Input: An image I, a target T and positive integers θ and ϕ. Each pixel $p \in I$ has associated values diff (p) and corr (p) (values have fixed precision ϵ).
Question: Does there exist a subset of pixels $I' \subseteq I$ such that $\sum_{p \in I'} \text{diff}(p) \leq \theta$ and $\sum_{p \in I'} \text{corr}(p) \geq \phi$?

Here the functions diff (p) and corr (p) are defined as

$$\text{diff}(p) = \sum_{p \in I'} \left(\sum_{j \in M_\text{i}} |t_{x,y,j} - i_{x,y,j}| \right) \tag{4}$$

and

$$\text{corr}(p) = \sum_{p \in I'} \left(\sum_{j \in M_t} |t_{x,y,j} \times i_{x,y,j}| \right), \tag{5}$$

where M_i and M_t denote the sets of measurement types in the image and target, respectively (e.g. color, brightness, motion, depth), $i_{x,y,j}$ denotes the value of pixel $p \in I$ with coordinates x and y for measurement type $j \in M_i$, and $t_{x,y,j}$ denotes the value of pixel $p \in T$ for measurement type $j \in M_\text{t}$. Note that the model does not require pixels in I' to be spatially contiguous, which may serve to ensure targets can still be detected when partially occluded by other objects.

In its general form, Bottom-up Visual Matching is known to be *NP*-hard [33, 68], but it has a pseudopolynomial-time algorithm [70, 71], which implies the problem is in *FPT* when parameterized by θ or λ (where λ is the smallest integer such that $i_{x,y,j} \leq \lambda$) [17]. It is not known, however, how other parameters—such as $|T|$, ϕ, $|M_i|$ and $|M_t|$—contribute to the problem's complexity. Also, equations (4) and (5) seem to impose strong constraints on the values that diff(p) and corr(p) can take (see also [17, p. 172]), constraints that may be utilized in the efficient computation of bottom-up visual matching.

5.1.2. *Action planning*

Although the planning problems examined in Section 4.2 are intractable, it is sobering to realize that they are actually *simplified* versions of problems which occur in practice. As these more realistic problems are known to be intractable under even more restricted circumstances than those in Section 4.2, they both offer more scope for and have greater need of parameterized analysis.

Consider first the case of motion planning. The kD-GM problem examined in Section 4.2.1 is an example of a *static movers problem*, in which the obstacles in the environment are stationary. However, motion planning in practice requires an agent to take into account obstacles that themselves move. For example, our baggage-laden traveler must deal not only with fixed walls and subway barricades, but also revolving doors, regularly scheduled buses and taxis and (perhaps worst of all) other similarly harried travelers. To date, work on such *dynamic movers problems* has focused on those cases in which all obstacle trajectories are both regular and known as part of the input, and the goal is now to find a

[7] The likelihood principle states that the visual interpretation of the retinal image is one that maximizes the probability of veridicality; the simplicity principle states that the visual interpretation of the retinal image is one that minimizes descriptive complexity [9].

path for the agent from p_I to p_F that does not collide with any of the (now moving) obstacles.

Let us denote the problem described above as k-Dimensional Dynamic Generalized Mover (kD-DGM). Two major intractability results are known for kD-DGM [72]—namely, if objects are allowed to rotate as they move, 3D-DGM is *PSPACE*-hard when the agent is modeled by a single disk that has restrictions on how quickly it can change velocity and direction and *NP*-hard when the agent is modeled as a cylinder with no such velocity/direction-change restrictions. Reif and Sharir [72] also give a number of algorithms. Of particular interest are those versions of kD-DGM in which objects are not allowed to rotate as they move (the so-called *Asteroid Avoidance Problems* [72]), because the 2-D version of this problem is solvable in polynomial time when the agent is modeled by a polygon and there are a constant number of obstacles. Given that kD-DGM is intractable for arbitrary obstacle-environments and polynomial-time for severely restricted obstacle-environments, aspects encoding the obstacle-environment (as in Section 4.2.1) are prime candidates for further research. It would also be interesting to further examine the interaction of aspects encoding agent-structure and the obstacle-environment to see if there are tractable cases of kD-DGM relative to non-trivial agent-structures.

Consider now the case of general action-sequence planning. The k-PSP problem examined in Section 4.2.2 assumed that (i) all assertions comprising a state are known and accessible and (ii) all actions are deterministic; i.e. the application of an operator to a state yields exactly one other state. However, either (and sometimes both) of these conditions may be violated in practice. For example, the availability-status of all possible cooking ingredients may not be known to a cook when cooking starts, and when an oven is heated, a cook may only know that a temperature is in a certain range. Moreover, at any point, the result of applying an operator may depend not only on the current state, but on the entire previous operator-application history (e.g. loan officers consider not only current bank balance but also credit history when a loan application is made). Versions of STRIPS planning have been introduced to model both of these contingencies (called *conformant planning* and *conditional planning*, respectively). It is known that conformant planning for plans of length 1 and conditional planning for plans of length $k > 0$ are hard for levels of the polynomial hierarchy above *NP* [73–75]. It would be interesting to know how the aspect-sets underlying SoCs derived relative to k-PSP can be extended to derive SoCs for these problems.

5.1.3. *Higher cognition*

Intractability results abound for models of higher cognition (e.g. [19, 35, 36, 52–57]), presenting ample opportunities for parameterized complexity analysis. We have selected three subdomains of higher cognition for exposition here: similarity judgment, categorization and defeasible reasoning.

The notion of similarity is foundational to many theories of higher cognition. It is often assumed to be part of the input for processes that compute conceptual coherence, metaphors and analogies [18, 57, 76], and also for object recognition and categorization [77–79]. We already saw a model of visual similarity in the form of Bottom-up Visual Matching. Hahn *et al.* [80] proposed a generalized computational-level model of human similarity judgments for any two mental representations (not only images) (see also [11]).

Generalized Transformational Similarity
Input: Two strings s and t and a set of transformation rules R.
Output: An ordered sequence of transformation rules r_1, r_2, ..., $r_k \in R$ that when applied to s yields t such that k is minimized.

The judged similarity of s and t is assumed to be a monotonically decreasing function of k. When the model is applied to visual object similarity, s and t may be assumed to be codes as computed by the Minimum Encoding function (see Section 5.1.1).

If no constraint on the set of rules R is imposed, e.g. if R could encode any Turing machine [52], then Generalized Transformational Similarity is uncomputable [63]. Hahn *et al.* considered a five-tuple of rules: $R =$ {insertion, deletion, mirroring, shifting, reversal} to model how humans judge the similarity of sequences of black and white dots. As far as we know, no computational complexity results are known for this restricted version of the problem, but possibly results may be derivable from results for related sequence-similarity problems in computational biology (see [81, 82] and references). Because the set of rules considered by Hahn *et al.* need not generalize to more abstract and conceptual similarity judgments (e.g. which concept is more similar to 'game'? The concept 'fight' or 'play'?), it is of importance to also investigate alternative restrictions on the set R.

The problem of categorization is to form groups of representations (e.g. objects, words, concepts) that are judged to belong together (e.g. humans group canines in the category 'dogs' and felines in the category 'cats'; they may further group dogs and cats together in the category 'mammal' and together with birds in the category 'animals'). An established hypothesis in psychology is that basic-level categories—such as 'dogs', 'cats', 'apples', 'cars', etc., but not necessarily superordinate categories such as 'mammals', 'animals', 'fruit' and 'vehicles'—are formed so as to maximize within-category similarity and between-category dissimilarity (see, e.g. [77, 78]). Here is one possible way to formalize this idea:

Object Categorization
Input: A set of objects A, with for each pair of objects $a, b \in A \times A$ an associated similarity weight $s(a, b)$ and dissimilarity weight $d(a, b)$.

Output: A partition of A into disjoint sets $A_1, A_2, \ldots, A_m$ such that $\sum_i \sum_{a,b \in A_i} s(a, b) + \sum_{i,j,i \neq j} \sum_{a \in A_i, b \in A_j} d(a, b)$, with $i, j \in \{1, 2, \ldots, m\}$, is maximized.

If $s(a, b) = 0$ for all $a, b \in A \times A$, the problem is equivalent to clustering, and hence *NP*-hard [1]. When the problem is further restricted such that $m = 2$ and $d(a, b) \in \{0, 1\}$ for all $a, b \in A \times A$, it is equivalent to the *NP*-hard problem Max Cut [1, Problem ND16], and hence the problem is not in *XP* for parameter m unless $P = NP$.[8] Other restrictions or parameterizations, however, may render the problem tractable. Note, for example, that in Object Categorization $s(a, b)$ and $d(a, b)$ are completely independent measures. It may be that, for purposes of object categorization, humans mentally represent similarity as the inverse of dissimilarity (though likely not for all similarity judgments, see [83]). Also, other objective functions may be more descriptive of human categorization. Consider, for example, the following variant of Object Categorization (which yields two new parameters p and q):

Object Categorization*
Input: A set of objects A, with for each pair of objects $a, b \in A \times A$ an associated similarity weight $s(a, b)$ and dissimilarity weight $d(a, b)$. Positive integers p and q.
Output: A partition of A into disjoint sets $A_1, A_2, \ldots, A_m$ such that $\sum_i \sum_{a,b \in A_i} s(a, b) \geq p$ and $\sum_{i,j,i \neq j} \sum_{a \in A_i, b \in A_j} d(a, b) \geq q$.

An altogether different formalization of object categorization, based solely on relative similarities (e.g. $s(a, b) > s(c, d)$, $s(a, b) < s(d, e)$, etc.), has been put forth by Pothos and Chater [84, 85]. Their model is based on the idea that categories serve as a compressed description of a set of relative similarity relations. In the model, an inequality $s(a, b) > s(c, d)$ is said to be correctly 'described' by the partition $A_1, A_2, \ldots, A_m$ if and only if $a, b \in A_i$, $c \in A_j$, $d \in A_k$, with $i \neq j \neq k$. The output of the categorization process is assumed to be one that minimizes both description length (i.e. the size of m) and the errors in description (we refer the reader to [85] for details). The model seems to be of high complexity (likely *NP*-hard, though no result is known to date) and lends itself to interesting analyses from a parameterized perspective.

Lastly, we consider models of reasoning, in particular non-monotonic or defeasible reasoning [86]. An inference is said to be defeasible if its conclusions may be changed upon encountering new information; in other words, all non-deductive inferences are defeasible. Most everyday inferences made by humans are defeasible (e.g. when told that John is married, you infer he has a wife; if later you find out that the spouse's name is Bill, you may conclude otherwise). Defeasibility is also a characteristic of inferences in many professional settings, e.g. when doctors infer diseases from symptoms and when judges/juries infer a defendant's guilt or innocence from the available evidence. We already encountered a model of defeasible inference in Section 4.3.2, viz. the Coherence model of Thagard and Verbeurgt [18, 19]. Its two main competitors are the Logicist model and the Bayesian model. We will sketch each in turn.

The Logicist model proposes that humans make defeasible inferences that follow from default rules (e.g. if p and q are married, then p is a man and q is a woman, or vice versa) derived from generalized beliefs (most married couples are of opposite sex). A common assumption is that a (default) belief is held provided it is consistent with the current state of a person's knowledge [87, 88]. This amounts to the computation of an instance of the following generalized problem.

Generalized Default Logic
Input: A knowledge base K and a set of default rules R.
Question: Is there a proposition p derivable from K using R, such that p and K are consistent?

In contrast, the Bayesian model assumes that people make defeasible inferences so as to maximize the conditional probability of their beliefs in light of the statistical knowledge they have of the world (e.g. all else being equal, P(John's spouse is a woman) $>$ P(John's spouse is a man), but given the fact that the spouse's name is Bill, the reverse is true). This intuition leads to the following generalized model.

Generalized Bayesian Inference
Input: A knowledge base K and a set of competing hypotheses H.
Output: A hypothesis $h \in H$ that maximizes the conditional probability $P(h|K)$.

It seems fair to say that the Bayesian model is gaining in popularity in the cognitive science community [89], in part because of its purported descriptive accuracy [86, 90, 91], and also because the Logicist model has fallen into discredit ever since the first intractability results became known [61, 87, 92, 93]. It has become clear, however, that the Bayesian model is no less subject to intractability concerns [53, 56, 94]. These negative results have led some cognitive scientists to resort to heuristics as algorithmic-level explanations [95, 96], but we believe cognitive science could benefit more from systematic SoC analyses aimed at identifying model aspects that make defeasible inference under these models computationally so demanding (see also Section 3.1.2).

5.2. New parameterized techniques

Historically, the development of computational complexity theory has been motivated in large part by forward-engineering applications like software and algorithm development. As a result, there are relatively few complexity-theoretic techniques specifically tailored to the reverse engineering approach underlying cognitive science. In this section, we describe two types of parameterized techniques that would, if developed, be of direct use to cognitive science—namely,

[8] We thank an anonymous reviewer for pointing this out.

techniques for analyzing cognitive computational architectures (Section 5.2.1) and guiding the search for SoCs (Section 5.2.2).

5.2.1. Analyzing cognitive architectures

At present, computational-level cognitive analyses typically do not incorporate processing constraints imposed by biological neural architectures. As was pointed out in Section 3.1.1, given the equivalence of Turing machines and the most general abstract formulations of these architectures relative to commonly-adopted standards of tractability, this is not necessary in computational-level analyses. However, as knowledge of the nature of biological neural architectures increases, it may be useful to be able to incorporate such constraints to both speed convergence on and decrease the size of the set of possible cognitive functions.

One way of imposing such architectural constraints in computational-level analyses is to perform complexity analysis relative to a particular computational architecture. In the context of the research described in this paper, this would require the development of new theories of parameterized complexity based on computational architectures such as circuits or neural nets that are closer to those that exist in actual cognitive systems. Parameterized tractability might then be restated in terms not only of processing time of an algorithm, but also its underlying neural structure (e.g. required circuit-algorithm depth/fanin/fanout, required number of feedback-loop traversals to achieve stability of output). Once such theories are available, it would be possible to further integrate such constraints into the problem-definition themselves, as aspects constraining operations within these architectures (e.g. maximum allowable circuit-algorithm depth or gate width). Work has been done on complexity theories based on such architectures within the classical complexity framework, both in the context of parallel and neural computation (e.g. [31, 97]), and it may be relatively easy (once an impetus is provided) to accelerate the extensions of this work within the parameterized framework that have been done to date (e.g. [98, 99]).

5.2.2. Guiding SoC search

In order to efficiently implement the tractable-design cycle in cognitive scientific practice, it would be desirable to develop general techniques that can guide the search for SoCs in cognitive models. Here, we distinguish between two types of SoCs, those that reside in the (lack of) constraints on the input domains and those that arise from the functional form of the chosen input/output mapping.

5.2.2.1. SoCs in the input domain. As described in Section 3.2.2, systematic parameterized analysis is useful in making sure that no SoC relative to the set S of aspects considered in the analysis is missed and that proposed SoCs are in fact minimal relative to S. However, such an analysis assumes that the cognitive scientist has already identified the relevant set of aspects. Although some potential SoCs may be evident from parameters stated explicitly in the problem definition, implicit parameters are typically harder to identify because they lay hidden in assumptions about the processes that produced the inputs.

The complexity of the web of interlinked cognitive processes, akin to the ecological web of relationships among various animals and plants co-existing in a particular environment, renders the total set of such assumptions (and hence the total set of implicit SoCs) inaccessible in practice—however, the fact that such interlinking exists may provide a way of deriving assumptions relative to specific processes (and hence untangle this web a piece at a time). Just as an animal population is constrained in practice by the ecological relationships in which that animal participates, the space of possible inputs of a cognitive process is ecologically constrained by the cognitive processes with which that process interacts. As an illustration, consider again the models of similarity and categorization discussed in Section 5.1.3. The input of Categorization consists of pairs of objects with assigned similarity values, which implicitly assumes that there is some preceding process that computes these values. Depending on one's model of similarity, those similarity values may have different properties and interdependencies, all of which may be SoCs in the computation of categorization.

In light of this observation, the development of techniques for identifying implicit SoCs may capitalize on insights gained from the study of the 'ecologies of hidden parameters of feasibility' [100]; e.g. by considering different problems ψ_1 and ψ_2, and analyzing the complexity of ψ_1 assuming its inputs are the output of ψ_2 (see also [51]). Furthermore, the potential input producing systems ψ_{11}, ψ_{12}, ... are all constrained by the requirement of tractability, and the same holds for processes that provide their inputs, ψ_{111}, ψ_{112}, ..., and so on. Perhaps techniques can be developed to propagate and exploit tractability constraints on whole *chains* of models, $\psi_a(\psi_b(\psi_c(\ldots)))$, in a way that allows cognitive modelers to faster converge on feasible alternatives for any individual model in the chain.

5.2.2.2. SoCs in the functional form. As explained in Section 2.3, cognitive models do not start off as well-defined input/output mappings; instead the cognitive scientist starts with some basic intuitions about the nature of the inputs and outputs and a first informal hypothesis about how inputs map onto outputs. In going from these informal ideas to a formal, well-defined input/output mapping many seemingly arbitrary choices need to be made. Any one of those choices is a potential SoC and it would be useful if cognitive scientists would have at their disposal some methods for recognizing the ones that are.

Consider, for example, the two model variants of Coherence considered in Section 4.3.2: one assumes humans compute maximum coherence and the other assumes humans compute at least *c* coherence for some aspiration level *c* (the latter being what Herbert Simon [101] called 'satisficing'). It is general knowledge in computer science (but not in cognitive science) that the change from maximizing to satisficing alone (e.g. without imposing a bound on *c*) has negligible effect on computational complexity, because maximum coherence can be determined by computing the decision version for the lowest *c* that yields a Yes answer. For other types of changes the effects on computational complexity are not as obvious, but often equivalence can still be proved by quite simple arguments (compare, for example, OBJECT CATEGORIZATION and OBJECT CATEGORIZATION* in Section 5.1.3). Possibly using a base-set of such transparent problem relations, we can develop methods for systematically mapping out relations between different cognitive models in a way that helps elucidate which functional properties introduce complexity and which reduce them.

To take this even one step further: Can we develop methods for comparing competing cognitive model *classes* in terms of the SoCs that they introduce? Recall that the three model classes, COHERENCE, (Section 4.3.2), DEFAULT LOGIC and BAYESIAN INFERENCE (Section 5.1.3) all aim at modeling the same human cognitive capacity, viz. defeasible reasoning. It would be useful if we could somehow evaluate what is gained or lost in terms of computational efficiency when moving from one model class to another. Not only can this save cognitive scientists a lot of time by not having to prove (in)tractability results for the new model class from scratch, but it will also allow the direct identification of conditions under which the frameworks do not *differ* in terms of (in)tractability, which gives a better handle on assessing the relative strengths and weaknesses of competing explanatory frameworks.

6. CONCLUSION

In this paper, we have made the case for a natural application of parameterized complexity in cognitive modeling. We have clarified the conceptual foundations of parameterized cognitive analysis, reviewed existing applications and provided a rich list of future opportunities for research. It is our hope that the work presented here will provide an impetus for others to contribute results and methods to this new fusion of cognitive and computer science. We further believe that doing so will provide benefits for both fields.

- Cognitive science gains by bringing in powerful analytic tools for identifying and isolating SoC in cognitive models. These tools will not only help to converge on model veridicality faster, but also to achieve a deeper insight into the subtle relationships among the form of the functions computed by cognitive processes, the ecology of cognitive inputs and cognitive efficiency.
- Computer science gains by the opening up of a whole field of unexplored problems raised by the reverse engineering perspective of cognitive science. Work on these problems is of interest in itself; moreover, the analytic methods and techniques derived along the way may also find application in other sciences studying natural computing processes such as biology and economics.

ACKNOWLEDGEMENTS

This article was written while I.v.R. was at Eindhoven University of Technology, Eindhoven, The Netherlands. She gratefully acknowledges the support of the J.F. Schouten school at Eindhoven University of Technology and thanks Michael Fellows and Ulrike Stege for discussions that set the seed for this paper. T.W. gratefully acknowledges the financial support provided by the National Science and Engineering Research Council (NSERC Grant 228104) and thanks Ewa Czaykowska-Higgins for discussions about the role complexity results can and should play in linguistics. The authors also thank an anonymous reviewer for useful comments on an earlier draft of this article.

REFERENCES

[1] Garey, M.R. and Johnson, D.S. (1979) *Computers and Intractability: A Guide to the Theory of NP-Completeness.* W.H. Freeman, San Francisco.

[2] Downey, R.G. and Fellows, M.R. (1999) *Parameterized Complexity*. Springer-Verlag, Berlin.

[3] Cummins, R. (2000) How does it work? vs. What are the laws? Two conceptions of psychological explanation. In Keil, F. and Wilson, R. (eds), *Explanation and Cognition*. MIT Press, Cambridge, MA.

[4] Marr, D. (1981) *Vision: A Computational Investigation into the Human Representation and Processing Visual Information*. W.H. Freeman, San Francisco.

[5] Anderson, J.R. (1990) *The Adaptive Character of Thought.* Lawrence Erlbaum Associates, Hillsdale, NJ.

[6] Frixione, M. (2001) Tractable competence. *Minds Mach.*, **11**, 379–397.

[7] Dennett, D.C. (1994) Cognitive science as reverse engineering: several meanings of top down and bottom up. *Proc. 9th Int. Congress of Logic, Methodology and Philosophy of Science*, pp. 679–689. North Holland.

[8] Koffka, K. (1935) *Principles of Gestalt Psychology*. Harcourt Brace, New York.

[9] van der Helm, P.A. (2000) Simplicity versus likelihood in visual perception: from surprisals to precisals. *Psychol. Bull.*, **126**, 770–800.

[10] van der Helm, P.A. (2007) The resurrection of simplicity in vision. In Peterson, M.A., Gillam, B. and Sedgwick, H.A. (eds), *In the Mind's Eye: Julian Hochberg on the Perception*

of Pictures, Film, and the World, pp. 518–524. Oxford University Press, New York.

[11] Chater, N. and Vitányi, P. (2003) Simplicity: a unifying principle in cognitive science? *Trends Cogn. Sci.*, **7**, 19–22.

[12] Leeuwenberg, E.L.J. and van der Helm, P.A. (1996) Goodness of visual regularities: a nontransformational approach. *Psychol. Rev.*, **103**, 429–456.

[13] van der Helm, P.A. (2004) Transparallel processing by hyperstrings. *Proc. Natl. Acad. Sci. USA*, **101**, 10862–10867.

[14] Bergströom, L. (1984) Underdetermination and realism. *Erkenntnis*, **21**, 349–365.

[15] Quine, W.V. (1975) On empirically equivalent systems of the world. *Erkenntnis*, **9**, 313–328.

[16] Anderson, J.R. (1978) Arguments concerning representations for mental imagery. *Psychol. Rev.*, **85**, 249–277.

[17] van Rooij, I. (2003) Tractable cognition: complexity theory in cognitive psychology. PhD Thesis, Department of Psychology, University of Victoria.

[18] Thagard, P. (2000) *Coherence in Thought and Action*. MIT Press, Cambridge, MA.

[19] Thagard, P. and Verbeurgt, K. (1998) Coherence as constraint satisfaction. *Cogni. Sci.*, **22**, 1–24.

[20] van Rooij, I. and Wright, C. (2006) The incoherence of heuristically explaining coherence. *Proc. 28th Annual Conf. of the Cognitive Science Society*, p. 2622. Lawrence Erlbaum Associates, Hillsdale, NJ.

[21] Koskenniemi, K. and Church, K.W. (1988) Complexity, two-level morphology, and finnish. *Proc. 12th Int. Conf. Computational Linguistics* (*COLING'88*), pp. 335–340. John von Neumann Society for Computing Sciences, Budapest.

[22] Manaster Ramer, A. (1995) Book review: Ristad, *The Language Complexity Game. Comput. Linguist.*, **21**, 124–131.

[23] Rounds, W. (1987) Book review: Barton, Berwick, and Ristad, *Computational complexity and natural language*. *Comput. Linguist.*, **13**, 354–356.

[24] Rounds, W. (1991) The relevance of computational complexity theory to natural language processing. In Sells, P., Shieber, S. and Wasow, T. (eds), *Foundational Issues in Natural Language Processing*. MIT Press, Cambridge, MA.

[25] Barton, G.E., Berwick, R.C. and Ristad, E.S. (1987) *Computational Complexity and Natural Language*. MIT Press, Cambridge, MA.

[26] Berwick, R.C. and Weinberg, A. (1983) Parsing efficiency, computational complexity, and the evaluation of grammatical theories. *Linguist. Inquiry*, **13**, 165–191.

[27] Ristad, E.S. (1990) Computational Structure of human language. PhD Thesis, Department of Electrical Engineering and Computer Science, MIT.

[28] Ristad, E.S. (1993) *The Language Complexity Game*. MIT Press, Cambridge, MA.

[29] Wareham, T. (1996) The role of parameterized computational complexity theory in cognitive modeling. In Ling, C.X. and Sun, R. (eds), *Working Notes of AAAI-96 Workshop on Computational Cognitive Modelling: Source of the Power*.

[30] Wareham, T. (1999) Systematic parameterized complexity analysis in computational phonology. PhD Thesis, Department of Computer Science, University of Victoria.

[31] Parberry, I. (1994) *Circuit Complexity and Neural Networks*. MIT Press, Cambridge, MA.

[32] van Emde Boas, P. (1990) Machine models and simulations. In van Leeuwen, J. (ed), *Handbook of Theoretical Computer Science, Volume A: Algorithms and Complexity*. MIT Press, Cambridge, MA.

[33] Tsotsos, J.K. (1990) Analyzing vision at the complexity level. *Behav. Brain Sci.*, **13**, 423–469.

[34] Simon, H.A. (1988) Rationality as process and as product of thought. In Bell, D.E., Raiffa, H. and Tversky, A. (eds), *Decision-making: Descriptive, Normative, and Prescriptive Interactions*. Cambridge University Press.

[35] Martignon, L. and Hoffrage, U. (2002) Fast, frugal, and fit: simple heuristics for paired comparison. *Theory Decis.*, **52**, 29–71.

[36] Martignon, L. and Schmitt, M. (1999) Simplicity and robustness of fast and frugal heuristics. *Minds Machi.*, **9**, 565–593.

[37] van Rooij, I., Stege, S. and Kadlec, H. (2005) Sources of complexity in subset choice. *J. Math. Psychol.*, **49**, 160–187.

[38] Ristad, E.S. (1993) Complexity of the simplified segmental phonology. Technical Report CS-TR-388-92 (revised May 1993), Department of Computer Science, Princeton University.

[39] Eisner, J. (1997) Efficient generation in primitive optimality theory. Technical Report ROA-206-0797, Rutgers Optimality Archive.

[40] Downey, R.G., Fellows, M.R., Kapron, B.M., Hallett, M.T. and Wareham, T. (1994) Parameterized complexity of some problems in logic and linguistics (extended abstract). *Logical Foundations of Computer Science*, pp. 89–101. Lecture Notes in Computer Science, Vol. 813. Springer-Verlag, Berlin.

[41] Wareham, T. (2001) The parameterized complexity of intersection and composition operations on sets of finite-state automata. *Proc. Fifth Int. Conf. Implementation and Application of Automata*, pp. 302–310. Lecture Notes in Computer Science, Vol. 2088. Springer-Verlag, Berlin.

[42] Kaplan, R.M. and Kay, M. (1994) Regular models of phonological rule systems. *Comput. Linguist.*, **20**, 331–378.

[43] Hopcroft, J.E. and Ullman, J.D. (1969) *Formal Languages and Their Relation to Automata*. Addison-Wesley, Reading, MA.

[44] Reif, J.H. (1987) Complexity of the generalized mover's problem. In Schwartz, J.T., Sharir, M. and Hopcroft, J.E. (eds), *Planning, Geometry, and Complexity of Robot Motion*. Ablex Publishing, Norwood, NJ.

[45] Joseph, D.A. and Plantinga, W.H. (1985) On the complexity of reachability and motion planning problems. *Proc. First ACM Symp. Computational Geometry*, pp. 62–66. ACM Press, New York.

[46] Cesati, M. and Wareham, T. (1995) Parameterized complexity analysis in robot motion planning. *Proc. 25th IEEE Int. Conf.*

Systems, Man, and Cybernetics: Volume 1, pp. 880–885. IEEE Press, Los Alamitos, CA.

[47] Hopcroft, J.E., Schwartz, J. and Sharir, M. (1984) On the complexity of motion planning for multiple independent objects: *PSPACE*-hardness of the Warehouseman's Problem. *Int. J. Robot. Res.*, **3**, 76–88.

[48] Fernau, H., Hagerup, T., Nishimura, N., Ragde, P. and Reinhardt, K. (2003) On the parameterized complexity of a generalized Rush Hour puzzle. *Proc. 15th Canadian Conf. Computational Geometry (CCCG'03)*, pp. 6–9.

[49] Bylander, T. (1994) The computational complexity of propositional STRIPS planning. *Artif. Intell.*, **69**, 165–204.

[50] Fikes, R.E. and Nilsson, N.J. (1971) STRIPS: a new approach to the application of theorem proving to problem solving. *Artif. Intell.*, **2**, 189–208.

[51] Downey, R.G., Fellows, M.R. and Stege, U. (1999) Parameterized complexity: a framework for systematically confronting computational intractability. *The Future of Discrete Mathematics: Proc. First DIMACS-DIMATIA Workshop*, pp. 49–99. AMS-DIMACS Series in Discrete Mathematics and Theoretical Computer Science, Vol. 49. American Mathematical Society, Providence, RI.

[52] Chater, N. and Vitányi, P. (2003) Generalized law of universal generalization. *Journal of Mathematical Psychology*, **47**, 346–369.

[53] Cooper, G.F. (1990) The computational complexity of probabilistic inference using Bayesian belief networks. *Artif. Intell.*, **42**, 393–405.

[54] Fishburn, P.C. and LaValle, I.H. (1996) Binary interactions and subset choice. *Eur. J. Oper. Res.*, **92**, 182–192.

[55] Levesque, H.J. (1988) Logic and the complexity of reasoning. *J. Philos. Log.*, **17**, 355–389.

[56] Roth, D. (1996) On the hardness of approximate reasoning. *Artif. Intell.*, **82**, 273–302.

[57] Veale, T., O'Donoghue, D. and Keane, M.T. (1999) Computability as a limiting cognitive constraint: complexity concerns in metaphor comprehension about which cognitive linguists should be aware. In Hiraga, E.M., Sinha, C. and Wilcox, S. (eds), *Cultural, Psychological and Typological Issues in Cognitive Linguistics*. John Benjamins, Amsterdam.

[58] Fishburn, P.C. and LaValle, I.H. (1993) Subset preferences in linear and nonlinear utility theory, *J. Math. Psychol.*, **37**, 611–623.

[59] Ford, K.W. and Pylyshyn, Z. (eds) (1996) *The Robot's Dilemma Revisited: The Frame Problem in Artificial Intelligence*. Ablex Publishing, Norwood, NJ.

[60] Haselager, W.F.G. (1997) *Cognitive science and folk psychology: The right frame of mind.* Sage, London.

[61] Pylyshyn, Z. (ed) (1987) *The Robot's Dilemma: The Frame Problem in Artificial Intelligence.* Ablex Publishing, Norwood, NJ.

[62] Stege, U. and van Rooij, I. (2006). Computing maximum coherence: A hard nut to crack? *Paper presented at the 39th Annual Meeting of the Society for Mathematical Psychology*, Vancouver, BC.

[63] Li, M. and Vitányi, P. (1997) *An Introduction to Kolmogorov Complexity and its Applications* (2nd edn). Springer-Verlag, Berlin.

[64] van der Helm, P.A. and Leeuwenberg, E.L.J. (1986) Avoiding explosive search in automatic selection of simplest pattern codes. *Pattern Recognit.*, **19**, 181–191.

[65] Treisman, A. (1982) Perceptual grouping and attention in visual search for features and for objects. *J. Exp. Psychol.: Hum. Percept. Perform.*, **8**, 194–214.

[66] Wolfe, J. M. (2003) Moving towards solutions to some enduring controversies in visual search. *Trends Cogn. Sci*, **7**, 70–76.

[67] Tsotsos, J.K. (1988) A complexity level analysis of immediate vision. *Int. J. Comput. Vis.*, **2**, 303–320.

[68] Tsotsos, J.K. (1989) The complexity of perceptual search tasks. In Sridharan, N.S. (ed), *Proc. Int. Joint Conf. Artificial Intelligence*, pp. 1571–1577. Morgan Kaufmann Publishers, San Francisco.

[69] Tsotsos, J.K. (1991) Is complexity theory appropriate for analyzing cognitive systems? *Behav. Brain Sci.*, **14**, 770–773.

[70] Kube, P.R. (1990) Complexity is complicated. *Behav. Brain Sci.*, **13**, 450–451.

[71] Kube, P.R. (1991) Unbounded visual search is not both biologically plausible and *NP*-complete. *Behav. Brain Sci.*, **14**, 768–773.

[72] Reif, J.H. and Sharir, M. (1994) Motion planning in the presence of moving obstacles. *J. ACM*, **41**, 764–790.

[73] Baral, C., Kreinovich, V. and Trejo, R. (2000) Computational complexity of planning and approximate planning in presence of incompleteness. *Artif. Intell.*, **122**, 241–267.

[74] Eiter, T., Faber, W., Leone, N., Pfeiffer, G. and Polleres, A. (2000) Planning under Incomplete Knowledge. *Proc. First Int. Conf. Computational Logic*, pp. 807–821. Lecture Notes in Computer Science, vol. 1861, pp. 807–821. Springer-Verlag, Berlin.

[75] Turner, H. (2002) Polynomial-length planning spans the polynomial hierarchy. *Proc. Eighth European Conf. Logics in Computer Science* (*JELIA 2002*), pp. 111–124. Lecture Notes in Artificial Intelligence, Vol. 2424. Springer-Verlag, Berlin.

[76] Holyoak, K.J. and Thagard, P. (1997) The analogical mind. *Am. Psychol.*, **52**, 35–44.

[77] Rosch, E. (1973) On the internal structure of perceptual and semantic categories. In Moore, T.E. (ed), *Cognitive Development and the Acquisition of Language*. Academic Press, New York.

[78] Rosch, E. and Mervis, C.B. (1975) Family resemblances: studies in the internal structure of categories. *Cogn. Psychol.*, **7**, 573–605.

[79] Smith, E.E. (1995) Concepts and categorization. In Osherson, D.N. and Smith, E.E. (eds), *Thinking: An Invitation to Cognitive Science*. MIT Press, Cambridge, MA.

[80] Hahn, U., Chater, N. and Richardson, L.B. (2003) Similarity as transformation. *Cognition*, **87**, 1–32.

[81] Jones, N.C. and Pevzner, P.A. (2004) *An Introduction to Bioinformatics Algorithms*. MIT Press, Cambridge, MA.

[82] Pevzner, P.A. (2000) *Computational Molecular Biology: An Algorithmic Approach*. MIT Press, Cambridge, MA.

[83] Tversky, A. (1977) Features of similarity. *Psychol. Rev.*, **84**, 327–352.

[84] Pothos, E.M. and Chater, N. (2001) Category learning without labels — A simplicity approach. *Proc. 23rd Annual Conf. Cognitive Science Society*, pp. 774–779. Lawrence Erlbaum Associates, Hillsdale, NJ.

[85] Pothos, E.M. and Chater, N. (2002) A simplicity principle in unsupervised human categorization. *Cogn. Sci.*, **26**, 303–343.

[86] Oaksford, M.R. and Chater, N. (1998) *Rationality in an Uncertain World: Essays on Cognitive Science of Human Reasoning*. Psychology Press Ltd, Publishers, East Sussex, UK.

[87] Oaksford, M.R. and Chater, N. (1993) Reasoning theories and bounded rationality. In Manktelow, K.I. and Over, D.E. (eds), *Rationality: Psychological and philosophical perspectives*. Routledge, London.

[88] Reiter, R. (1980) A logic for default reasoning. *Artif. Intell.*, **13**, 81–132.

[89] Chater, N., Tenebaum, J.B. and Yuille, A. (eds) (2006) Special issue: probabilistic models in cognition. *Trends Cogn. Sci.*, **10**, 287–344.

[90] Griffiths, T.L. and Tenenbaum, J.B. (2006) Optimal predictions in everyday cognition. *Psychol. Sci.*, **17**, 767–773.

[91] Krynski, T.R. and Tenenbaum, J.B. (2003) The role of causal models in reasoning under uncertainty. *Proc. Twenty-Fifth Annual Conf. Cognitive Science Society*, pp. 692–697. Lawrence Erlbaum Associates, Hillsdale, NJ.

[92] Dreyfus, H. L. (1979) *What Computers Can't Do*. Harper Colophon Books, New York.

[93] Dreyfus, H.L. (1992) *What Computers Still Can't Do*. MIT Press, Cambridge, MA.

[94] Abdelbar, A. and Hedetniemi, S.M. (1998) Approximation MAPs for belief networks is *NP*-hard and other theorems. *Artif. Intell.*, **102**, 21–38.

[95] Gigerenzer, G. and Goldstein, D.G. (1996) Reasoning the fast and frugal way: models of bounded rationality. *Psychol. Rev.*, **103**, 650–669.

[96] Todd, P.M. and Gigerenzer, G. (2000) Précis of simple heuristics that make us smart. *Behav. Brain Sci.*, **23**, 727–780.

[97] Valiant, L.G. (1994) *Circuits of the Mind*. Oxford University Press.

[98] Cesati, M. and Di Ianni, M. (1997) Parallel parameterized complexity. *Electron. Colloq. Comput. Complexity*, **4**.

[99] Downey, R.G., Fellows, M.R. and Regan, K.W. (1998) Parameterized circuit complexity and the *W* hierarchy. *Theor. Comput. Sci.*, **191**, 97–115.

[100] Fellows, M.R. (2000) Parameterized complexity: new developments and research frontiers. In Downey, R.G. and Hirschfeldt, D. (eds), *Aspects of Complexity*, pp. 51–72. de Gruyter Series in Logic and Its Applications, Vol. 4. de Gruyler, Berlin.

[101] Simon, H.A. (1990) Invariants of human behavior. *Ann. Rev. Psychol.*, **41**, 1–19.