Chapter 30
Logic and Complexity in Cognitive Science

Alistair M. C. Isaac, Jakub Szymanik and Rineke Verbrugge

Abstract This chapter surveys the use of logic and computational complexity theory in cognitive science. We emphasize in particular the role played by logic in bridging the gaps between Marr's three levels: representation theorems for non-monotonic logics resolve algorithmic/implementation debates, while complexity theory probes the relationship between computational task analysis and algorithms. We argue that the computational perspective allows feedback from empirical results to guide the development of increasingly subtle computational models. We defend this perspective via a survey of the role of logic in several classic problems in cognitive science (the Wason selection task, the frame problem, the connectionism/symbolic systems debate) before looking in more detail at case studies involving quantifier processing and social cognition. In these examples, models developed by Johan van Benthem have been supplemented with complexity analysis to drive successful programs of empirical research.

30.1 Introduction

How can logic help us to understand cognition? One answer is provided by the computational perspective, which treats cognition as information flow in a computational system. This perspective draws an analogy between intelligent behavior as we

A. M. C. Isaac (✉)
Department of Philosophy, University of Edinburgh, Edinburgh, Scotland
e-mail: a.m.c.isaac@ed.ac.uk

J. Szymanik (✉)
Institute for Logic, Language, and Computation, University of Amsterdam, Amsterdam, The Netherlands
e-mail: J.K.Szymanik@uva.nl

R. Verbrugge
Institute of Artificial Intelligence, University of Groningen, Groningen, The Netherlands
e-mail: L.C.Verbrugge@rug.nl

A. Baltag and S. Smets (eds.), *Johan van Benthem on Logic and Information Dynamics*, Outstanding Contributions to Logic 5,
DOI: 10.1007/978-3-319-06025-5_30,

observe it in human beings and the complex behavior of human-made computational devices, such as the digital computer. If we accept this analogy, then the behavior of cognitive systems in general can be investigated formally through logical analysis. From this perspective, logical methods can analyze:

1. the boundary between possible and impossible tasks;
2. the efficiency with which any possible task can be solved; and
3. the low-level information flow which implements a solution.

This chapter will survey some examples of the application of logical techniques in each of these areas.

In general, we will see a back-and-forth between logical analysis and empirical findings. This back-and-forth helps to bridge the gap between the normative and descriptive roles of logic. For example, one may believe that people should perform a certain way on a given task because that is the "logical" or "rational" thing to do. We observe that they do not, in fact, perform as predicted. This does not change our assessment that human behavior can be described in terms of logical operations, but it changes our analysis of which task exactly humans perform in response to a particular experimental setup. The Wason selection task provides an example of this sort (Sect. 30.2.1).

Likewise, suppose we analyze the complexity of a task and determine that it has no efficient solution. If we observe humans apparently solving this task, we use this analysis as evidence that they are in fact solving a different, simpler problem. More generally, complexity analysis can make predictions about the relationship between, for example, input size and solution speed for particular tasks. These predictions can be compared with empirical evidence to determine when subjects switch from one algorithm to another, as in the counting of objects in the visual field or quantifier processing (Sect. 30.5.3).

Given the ubiquity of logic and its flexibility as a tool for analyzing complex systems, we do not presume to cover all possible roles of logic in cognitive science.[1] However, focusing on the *computational perspective* highlights two properties of logical analysis essential for cognitive science: it can clarify conceptual debates by making them precise, and it can drive empirical research by providing specific predictions. After some additional background on the computational perspective and Marr's levels of analysis, we conclude this section with a brief discussion of the influence of Johan van Benthem and an outline of the remainder of the chapter.

[1] For more references on the interface between logic and cognition, see also the 2007 special issue of *Topoi* on "Logic and Cognition", ed. J. van Benthem, H. Hodges, and W. Hodges; the 2008 special issue of *Journal of Logic, Language and Information* on "Formal Models for Real People", ed. M. Counihan; the 2008 special issue of *Studia Logica* on "Psychologism in Logic?", ed. H. Leitgeb, including [8]; and the 2013 special issue of *Journal of Logic, Language and Information* on "Logic and Cognition" ed. J. Szymanik and R. Verbrugge.

30.1.1 The Computational Perspective

The Church-Turing Thesis states that all computation, in the intuitive sense of a mechanical procedure for solving problems, is formally equivalent to computation by a Turing machine. This is a conceptual claim which cannot be formally proved. However, all attempts so far to explicate intuitive computability, many of them independently motivated, have turned out to define exactly the same class of problems. For instance, definitions via abstract machines (random access machines, quantum computers, cellular automata, genetic algorithms), formal systems (the lambda calculus, Post rewriting systems), and particular classes of function (recursive functions) are all formally equivalent to the definition of Turing machine computability. These results provide compelling support for the claim that all computation is equivalent to Turing computation (see, for example, [27]).

If we accept that the human mind is a physical system, and we accept the Church-Turing Thesis, then we should also accept its psychological counterpart:

> The human mind can only solve computable problems.

In other words, cognitive tasks comprise computable functions. From an abstract perspective, a cognitive task is an information-processing task:

> Given some input (e.g. a visual stimulus, a state of the world, a sensation of pain), produce an appropriate output (e.g. perform an action, draw a conclusion, utter a response).

Generally, then, cognitive tasks can be understood as functions from inputs to outputs, and the psychological version of the Church-Turing Thesis states that the only realistic candidates for information-processing tasks performed by the human mind are computable functions.

Not everyone accepts the psychological version of the Church-Turing Thesis. In particular, some critics have argued that cognitive systems can do more than Turing machines. For example, learning understood as identifiability in the limit [59] is not computable (see [72] for an extensive discussion). Another strand of argumentation is motivated by Gödel's theorems. The claim is that Gödel's incompleteness results somehow demonstrate that the human mind cannot have an algorithmic nature. For example, Lucas [78] claimed: "Gödel's theorem seems to me to prove that Mechanism is false, that is, that minds cannot be explained as machines". He gives the following argument: A computer behaves according to a program, hence we can view it as a formal system. Applying Gödel's theorem to this system we get a true sentence which is unprovable in the system. Thus, the machine does not know that the sentence is true while we can see that it is true. Hence, we cannot be a machine. Lucas' argument was revived by Penrose [93] who supplemented it with the claim that quantum properties

of the brain allow it to solve uncomputable problems. Lucas' argument has been strongly criticized by logicians and philosophers (e.g. [6, 96]), as has Penrose's (e.g. [39]).

If identifiability in the limit is the correct analysis of learning, or if the arguments from Gödel's theorems are correct, then we must accept the possibility of "hyper-" or "super-Turing" computation, i.e. the physical realization of machines strictly more powerful than the Turing machine. Examples of such powerful machines have been explored theoretically, for instance Zeno machines (accelerated Turing machines), which allow a countably infinite number of algorithmic steps to be performed in finite time (see [119] for a survey), or analog neural networks, which allow computation over arbitrarily precise real values (e.g. [108, 109]). However, no plausible account of how such devices could be physically realized has been offered so far. Both Penrose's appeal to quantum properties of the brain and Siegelmann's arbitrarily precise neural networks fail to take into account the noise inherent in any real-world analog system. [2] In general, any physical system, including the brain, is susceptible to thermal noise, which defeats the possibility of the arbitrarily precise information transfer required for hyper-computation.

However, there are two more interesting reasons to endorse the psychological version of the Church-Turing Thesis than simple physical plausibility. The first is its fruitfulness as a theoretical assumption. If we assume that neural computability is equivalent to Turing computability, we can generate precise hypotheses about which tasks the human mind can and cannot perform. The second is the close concordance between the computational perspective and psychological practice. Experimental psychology is naturally task oriented, because subjects are typically studied in the context of specific experimental tasks. Furthermore, the dominant approach in cognitive psychology is to view human cognition as a form of information processing (see e.g. [118]). The natural extension of this information processing perspective is the attempt to reproduce human behavior using computational models. Although much of this work employs Bayesian or stochastic methods[3] (rather than logic-based formalisms), it is predicated on the assumption of the psychological version of the Church-Turing Thesis.

---

[2] Siegelmann repeatedly appeals to a result in Siegelmann and Sontag [107] when arguing in later papers that analog neural networks do not require arbitrary precision (and are thus physically realizable). In particular, their Lemma 4.1 shows that for every neural network which computes over real numbers, there exists a neural network which computes over truncated reals (i.e. reals precise only to a finite number of digits). However, the length of truncation required is a function of the length of the computation—longer computations require longer truncated strings. Consequently, if length of computation is allowed to grow arbitrarily, so must the length of the strings of digits over which the computation is performed in a truncated network. Thus, one still must allow computation over arbitrarily precise reals if one is considering the computational properties of analog neural networks *in general*, i.e. over arbitrarily long computation times.

[3] For an overview, see the 2006 special issue of *Trends in Cognitive Sciences* (vol. 10, no. 7) on probabilistic models of cognition, or [129].

30.1.2 Marr's Levels

If we assume the psychological version of the Church-Turing Thesis, what does it tell us about how to analyze cognition? Marr [79] proposed a general framework for explanation in cognitive science based on the computational perspective. He argued that any particular task computed by a cognitive system must ultimately be analyzed at three levels (in order of decreasing abstraction):

1. the computational level (the problem solved or function computed);
2. the algorithmic level (the algorithm used to achieve a solution); and
3. the implementation level (how the algorithm is actually implemented in neural activity).

Considerations at each of these levels may constrain answers at the others, although Marr argues that analysis at the computational level is the most critical for achieving progress in cognitive science. Combining Marr's arguments with those of Newell [90], Anderson [1] defends the *Principle of Rationality*, which asserts that the most powerful explanation of human cognition is to be found via analysis at the computational level under the assumption that task performance has been optimized on an evolutionary timescale, and not via analysis of underlying mechanisms.

However, Marr's three-level system can only be applied *relative to* a particular computational question. For instance, a particular pattern of neural wiring may implement an algorithm which performs the computational function of detecting edges at a particular orientation. But of each neuron in that pattern, we can ask what is its computational function (usually, to integrate over inputs from other neurons) and how is this function implemented (electrochemical changes in the cell). Likewise, we may take a detected edge as an informational primitive when analyzing a more high-level visual function, such as object identification. Nevertheless, the most obvious examples of computational-level analysis concern human performance on behavioral tasks, and the obvious target for implementation-level analysis is neural wiring. Algorithmic analysis via complexity theory can then play the crucial role of bridging the gap between these two domains.

30.1.3 The Contributions of Johan van Benthem

Johan van Benthem's research intersects with cognitive science at many places, several of which are discussed elsewhere in this volume. The present chapter focuses on two specific contributions of Johan van Benthem's which, when supplemented with complexity analysis, have generated fruitful programs of empirical research: his analysis of quantifiers as automata (Sect. 30.5.2) and his models of interactive social reasoning (Sects. 30.6.1, 30.6.2).

The remainder of this chapter surveys applications of logic in cognitive science which are not directly connected to the specifics of Johan van Benthem's research.

Nevertheless, our aim in crafting this survey has been to illustrate a theme which can be found throughout Johan van Benthem oeuvre, namely an ecumenical approach to formal systems, emphasizing *commonality* and *fine structure* rather than conflict and divergence. Where some would seek to defend the advantages of one system over others, Johan van Benthem's seeks to find ways in which apparent competitors are at essence the same (commonality) or, if they do differ, exactly how they differ, and whether or not intermediary systems might lie in between (fine structure). The theme of commonality emerges in our discussions of the Wason selection task (Sect. 30.2.1) and the symbolic/connectionist debate (Sect. 30.3.2). We emphasize the fine structure approach in our discussions of non-monotonic logics and hierarchies of complexity.

Our discussion is organized around Marr's three levels. After arguing for the importance of logic at the computational level through the famous examples of the Wason selection task and the frame problem (Sect. 30.2), we survey the role logic can play in bridging the gap between the algorithmic and implementation levels (Sect. 30.3). We then pause to introduce the basics of computational complexity theory and the P-Cognition Thesis (Sect. 30.4), before investigating the role that complexity analysis can play in bridging the gap between the computational and algorithmic levels through the examples of quantifier processing (Sect. 30.5) and social reasoning (Sect. 30.6).

30.2 The Computational Level: Human Behavior

A number of results from experimental psychology seem to indicate that humans do not behave in accordance with the recommendations of classical logic. Yet logic forms the foundation for norms of ideal rationality. Even fallacies of probabilistic or decision-theoretic reasoning often rest on violations of basic logical principles. Consider, for example, the *conjunction fallacy*: after reading a short passage about Linda which describes her as a social activist in college, 85 % of subjects rated the proposition that "Linda is a bank teller and is active in the feminist movement" as more probable than the proposition that "Linda is a bank teller" [133]. This is a fallacy because the axioms of probability ensure that $P(A\&B) \leq P(A)$ for all A and B, where this constraint itself follows from the basic axiom $A \wedge B \rightarrow A$ of propositional logic. Do results such as these demonstrate that humans are fundamentally irrational?

We argue that the apparent irrationality of human behavior does not undermine the use of logic in cognitive science, rather it provides evidence for the correct computational analysis of the task being performed. We examine the example of the Wason selection task, where apparently irrational behavior drove the development of increasingly sophisticated computational models. After discussing this specific example, we'll look at the frame problem, a more general challenge to the computational perspective. This problem motivated the development of non-monotonic logic as a means of providing a formal analysis of human reasoning. This tool will also prove useful when we examine the relationship between algorithm and implementation in Sect. 30.3.

30.2.1 Is Human Behavior Logical?

The Wason selection task [137, 138] is very simple. Subjects are shown four cards and told that all cards have a number on one side and a letter on the other. The faces visible to the subject read D, K, 3, and 7. The subject is then told "Every card which has a D on one side has a 3 on the other" and asked which cards they need to turn over to verify this rule. From a classical standpoint, the claim has the basic structure of the material conditional "D *is on one side* $\rightarrow$ 3 *is on the other side*", and the correct answer is to turn over cards D and 7. However, the most common answers (in order of decreasing frequency) are (1) D and 3; (2) D; (3) D, 3, and 7; and (4) D and 7. The classically correct answer ranks fourth, while the first-ranking answer includes an instance of the logical fallacy of *affirming the consequent*: judging that 3 is relevant for determining whether the conditional is true. Wason's robust and frequently reproduced result seems to show that most people are poor at modus tollens and engage in fallacious reasoning on even very simple tasks. Are we really this bad at logic? Even worse, Cheng and colleagues [20] suggest people may continue to do poorly at the task even when they have taken an introductory logic class! Does this imply that human behavior does not decompose into logical steps? Or that our neural wiring is somehow qualitatively different from the logical structure that can be found in typical computational devices?

As it turns out, there are complexities in the data. The original selection task involved an abstract domain of numbers and letters. When the problem is rephrased in terms of certain types of domain with which subjects are familiar, reasoning suddenly improves. For example, Griggs and Cox [60] demonstrate that if cards have ages on one side and types of drink on the other, subjects perform nearly perfectly when the task is to determine which cards to turn over to ensure that the rule "if a person is drinking beer, then that person is over 19 years old" is satisfied. This study builds upon earlier work by Johnson-Laird et al. [70], demonstrating a similar phenomenon when the task involves postal regulations.

What exactly is different between Wason's original setup and those involving underage drinking and postal regulations, and how should this difference affect our computational model? Johnson-Laird et al. [70] and Griggs and Cox [60] concluded that humans are better at logical reasoning in domains with which they are familiar: since the original Wason task involves an abstract domain of letters and numbers, subjects are confused and fail to reason correctly. Cosmides [28] and Cosmides and Tooby [29] argue that the results tell us something about cognitive architecture. In particular, they conjecture that questions about postal regulations and drinking laws trigger a "cheater detection module." The proposed module is said to be hard-wired to reason effectively in contexts where free-riders might undermine social structure, but provides no logical support for domain-general reasoning.

Stenning and van Lambalgen [117] propose an illuminating new logical analysis of the Wason selection task. They point out that Wason's assertion that there is only one correct answer is too quick, as it assumes a single interpretation of an ambiguous task. Subjects who interpret the described rule as stating some other kind

of dependency between *D*'s and 3's than that captured by the material conditional are not necessarily making an error. The key here is in figuring out the relevant difference between versions of the task on which subjects perform in accordance with classical rules and versions (such as the original) on which they do not. Is it because the latter are abstract and the former concrete? Because the latter are unfamiliar and the former familiar? Because the latter are domain-general while the former involve cheater detection? Stenning and van Lambalgen's novel suggestion here is that the crucial difference is in whether the subject interprets the task as merely *checking satisfaction of instances* or as actually *determining the truth of a rule*. In the case of familiar deontic rules, their truth is not at issue, only whether or not they are being satisfied. The deontic nature of these rules means that turning cards over cannot falsify the rules: underage drinking is still wrong, even if one discovers that it occurs. This strictly limits interpretation of the task to checking whether the rule has been satisfied. In contrast, the original version of the task may be interpreted as involving either a descriptive or a prescriptive rule, greatly increasing the cognitive burden on the subject.

None of these analyses of the Wason selection task abandons logic. Johnson Laird et al. [70] and Griggs and Cox [60] shift logical reasoning from innate ability to learned behavior in familiar domains. Cosmides and Tooby [29] locate logical reasoning within the hard wiring of domain-specific modules. Stenning and van Lambalgen [117] identify logical structure with neural structure and argue that apparent violations of logical principles are a side effect of task ambiguity.

30.2.2 The Frame Problem and Non-monotonic Logics

The Wason selection task was designed to probe classical deductive reasoning, but deduction does not exhaust logical inference. In a complex and changing world, cognitive agents must draw conclusions about their circumstances on the basis of incomplete evidence. Crucially, this evidence is *defeasible*, which means that conclusions drawn from it may be defeated by later evidence. For example, suppose I wake up in a strange place and hear voices around me speaking in Chinese; I might conclude that I am in a Chinese restaurant. When I feel the surface on which I lie gently undulating, however, I might revise my conclusion, deciding instead that I have been shanghaied and am currently a passenger on a Chinese junk. Although my evidence has increased, my conclusions have changed. Modeling this type of reasoning requires a *non-monotonic* framework.

Typically, a non-monotonic logic supplements an underlying classical logic with a new, non-monotonic connective and a set of inference rules which govern it. The rules describe a logic of *defeasible* inference, inferences which are usually safe, but which may be defeated by additional information. For example, from the fact that *this is a bird*, I can usually conclude that *this can fly*. This inference can be defeated, however, if I learn that *this is a penguin*. Symbolically, we want our system to ensure that $Bird(x) \Rightarrow Fly(x)$, but $Bird(x) \wedge Penguin(x) \not\Rightarrow Fly(x)$. Concepts which formalize

this defeasibility include circumscription [80], negation as failure [24], and default reasoning [98].

The original motivation for non-monotonic logic came from consideration of a particular type of defeasible reasoning, namely reasoning about a changing world. When an event occurs, humans are able to reason swiftly and effectively about both those features of the world which change *and those which do not*. The problem of how to keep track of those features of the world which do not change is called the "frame problem" [81]. The frame problem comes in both a narrow and a broad version (see the discussion in [33]). The broad version concerns the potential relevance of any piece of information in memory for effective inference, and has troubled philosophers of cognitive science since at least [42]. The original, narrow problem, however, has been effectively solved by non-monotonic logic (see [105] for a complete history and detailed treatment).

The basic idea is easy to see. If we allow ourselves default assumptions about the state of the world, we can easily reason about how it changes. For example, we might assume as a default that facts about the world do not change unless they are explicitly addressed by incoming evidence. Learning that you ate eggs for breakfast does not change my belief that my tie is blue. Without the basic assumption that features of the world not mentioned by my incoming evidence do not change, I would waste all my computational resources checking irrelevant facts about the world whenever I received new information (such as checking the color of my tie after learning what you had for breakfast). This consideration inspired McCarthy's assertion that, not only do "humans use …'non-monotonic' reasoning," but also "it is required for intelligent behavior" [80].

A more sophisticated form of default reasoning is found in systems which employ "negation as failure". Such a system may derive $\neg A$ provided it cannot derive A. Negation as failure is frequently implemented in systems using Horn clauses, such as logic programming. Horn clauses state conditional relations such that the antecedent is a (possibly empty) conjunction of positive literals and the consequent is a single positive literal or *falsum*. In general, the semantics for systems involving negation as failure involve fixed points, e.g. finding the minimal model which satisfies all clauses (for a survey, see [40]).

Kraus et al. [71] provide a unified approach to a hierarchy of non-monotonic logics of varying strengths. Their insight was to generalize the semantics of [106], which used a preference ("plausibility") ordering over worlds as a model for non-monotonic inference. Kraus et al. [71] realized that increasingly strict constraints on this semantic ordering correspond to increasingly powerful sets of syntactic rules, and used this insight to define the systems $\mathbf{C} \subseteq \mathbf{CL} \subseteq \mathbf{P} \subseteq \mathbf{M}$, where $\mathbf{C}$ ("cumulative reasoning") is the weakest non-monotonic system they consider and $\mathbf{M}$ ("monotonic") is equivalent to standard propositional logic. Intermediary systems are characterized semantically by added constraints on the plausibility ordering over worlds and syntactically by the addition of stronger inference rules. For example, models for $\mathbf{C}$ are sets of worlds ordered by a relation $\prec$ which is asymmetric and well-founded. $\mathbf{C}$ is strengthened to the system $\mathbf{CL}$ by adding the inference rule *Loop*:

$$\frac{\alpha_0 \Rightarrow \alpha_1, \alpha_1 \Rightarrow \alpha_2, \ldots, \alpha_{k-1} \Rightarrow \alpha_k, \alpha_k \Rightarrow \alpha_0}{\alpha_0 \Rightarrow \alpha_k} \quad (Loop)$$

Semantically, models for **CL** add the constraint that $\prec$ be transitive, i.e. form a strict partial order. Kraus et al. [71] show that systems which use Horn clauses collapse the distinction between **CL** and **P**.

In solving the frame problem, non-monotonic logics have proved their power for modeling defeasible inference at the computational level. As we shall see in the next section, they are also powerful tools for analyzing the structure of the implementation level.

30.3 Between Algorithm and Implementation

How are computations performed in the brain? The answer which has dominated neuroscience since the late nineteenth century is the neuron hypothesis of Ramón y Cajal. He was the first to observe and report the division of brain tissue into distinct cells: neurons. More importantly, he posited a flow of information from axon to dendrite through this web of neural connections, which he denoted by drawing arrows on his illustrations of neural tissue. From the computational perspective, it is natural to identify this flow of information from neuron to neuron as the locus of the computation for solving cognitive tasks.

It is worth noting that this is not the only game in town. A plausible alternative to the neuron hypothesis comes from the dynamical systems perspective, which asserts that the behavior of a family of neurons cannot be reduced to signals communicated between them. Instead, this perspective asserts that computations should be modeled in terms of a dynamical system seeking basins of attraction. Neuroscientists such as Freeman [44, 45] find support for this view in observed neural dynamics, while philosophers such as van Gelder [51, 52] have argued that the dynamical systems perspective constitutes a substantive alternative to the computational perspective of Sect. 30.1.1. With the recent interest in embodied and extended theories of mind, the dynamical systems perspective has become ubiquitous (e.g. Gärdenfors, this volume; see [23] for a survey). In the context of the present discussion, however, we treat it not as an alternative to computationalism, but as a substantive hypothesis within the computational paradigm.

Logic provides an abstract symbolic perspective on neural computation. As such, it can never be the whole story of the implementation level (which by definition involves the physical instantiation of an algorithm). Nevertheless, logic can help bridge the gap between the implementation and algorithmic levels by analyzing structural similarities across different proposed instantiations. For example, if we subscribe to the neuron hypothesis, it is natural to look for logic gates in the wiring between neurons. But we may also look for logic gates in the wiring between families of neurons, or equivalent structure in the relations between basins of attraction in a dynamical system. Logical analysis can distinguish the commonalities across implementation-level hypotheses from their true disagreements.

30.3.1 Logical Neurons

The classic work of McCullogh and Pitts [82] proved the first representation theorem for a logic in an artificial neural network. In general, a representation theorem demonstrates that for every model of a theory, there exists an equivalent model within a distinguished subset. In this case, the "theory" is just a time-stamped set of propositional formulas representing a logical derivation, and the distinguished subset in question is the set of neural networks satisfying a particular set of assumptions, for example, neural firing is "all or none", the only delay is synaptic delay, and the network does not change over time. McCulloch and Pitts show the opposite direction as well: the behavior of any network of the specified type can be represented by a sequence of time-stamped propositional formulas. The propositions need to be time-stamped to represent the evolution of the network through time: the activations of neurons at time t are interpreted as a logical consequence of the activations of neurons at time $t - 1$.

McCulloch and Pitts had shown how neurons could be interpreted as performing logical calculations, and thus, how their behavior could be described and analyzed by logical tools. Furthermore, their approach was modular, as they demonstrated how different patterns of neural wiring could be interpreted as logic gates: signal junctions which compute the truth value of the conjunction, disjunction, or negation of incoming signals. The applications of this result are limited by its idealizing assumptions, however. As neurophysiology has enriched our understanding of neural behavior, the hypothesis of synchronized computations cascading through a structurally unchanging network has become too distant from neural plausibility to resolve debates about implementation in the brain.

Nevertheless, logical methods continue to provide insight into the structure of neural computation. In the face of an increasingly complex theory of neurophysiology, two obvious research projects present themselves. The first focuses on realistic models of individual neurons. Sandler and Tsitolovsky [103], for example, begin with a detailed examination of the biological structure of the neuron, then develop a model of its behavior using fuzzy logic. A second project focuses on artificial neural networks designed to mimic brain dynamics as closely as possible. For example, Vogels and Abbott [136] ran a number of simulations on large networks of integrate-and-fire neurons. These artificial neurons include many realistic features, such as a resting potential and a reset time after each action potential is generated. After randomly generating such networks, Vogels and Abbott [136] investigated their behavior to see if patterns of neurons exhibited the characteristics of logic gates. They successfully identified patterns of activation corresponding to NOT, XOR, and other types of logic gate within their networks.

The idealizing assumptions of these models continue to temper the conclusions which can be drawn from them. Nevertheless, there is a trend of increasing fit between mathematical models of neural behavior and the richness of neurophysiology, and logic continues to guide our understanding of neurons *as computational units*. But from the standpoint of cognitive science, an explanatory question remains: are these

computational units the right primitives for analyzing cognition? More generally, is there some *in principle* difference between an analysis offered in terms of neural networks and one offered in terms of logical rules?

30.3.2 The Symbolic Versus Connectionist Debate

In an influential paper, Fodor and Pylyshyn [43] argued that (i) mental representations exhibit systematicity; (ii) representations in neural networks do not exhibit systematicity; therefore (iii) the appropriate formalism for modeling cognition is symbolic (not connectionist). Systematicity here is just the claim that changes in the meaning of a representation correspond systematically to changes in its internal structure (e.g. from my ability to represent "John loves Mary," it follows that I can also represent "Mary loves John"). Fodor and Pylyshyn claim that the only case in which representations in a neural network do exhibit systematicity is when the network is a "mere" implementation of a symbolic system.[4]

It is important to notice what is at stake here: if cognitive tasks manipulate representations, then the appropriate analysis of a cognitive task must respect the properties of those representations. The claim that explanations in cognitive science must be in terms of symbolic systems does not, however, restrict attention to the computational level. Paradigmatic examples of the symbolic approach in cognitive science such as [21] investigate the role of particular algorithms for solving information processing tasks (such as extracting syntactic structure from a string of words). Nevertheless, the claim is that somewhere between abstract task specification and physical implementation, explanatory power breaks down, and neural networks fall on the implementation side of this barrier.

The response from connectionist modelers was violent and univocal: Fodor and Pylyshyn [43] had simply misunderstood the representational properties of neural networks. Responses elucidated how representations in neural networks are "distributed" or "subsymbolic." Smolensky [113, 114], van Gelder [49, 50], Clark [22], and many others all emphasized the importance of acknowledging the distinctive properties of distributed representations in understanding the difference between neural networks and symbolic systems. Yet it is difficult to put one's finger on just what the essential feature of a distributed representation is which makes it qualitatively different from a symbolic representation. Since the late 1990s, the supposed distinction has largely been ignored as hybrid models have risen to prominence, such as the ACT-R architecture of Anderson and Lebiere [2], or the analysis of concepts in Gärdenfors [47]. These hybrid models combine neural networks (for learning) and symbolic manipulation (for high-level problem solving). Although pragmati-

[4] However, they do not indicate how such implementational networks avoid their general critique, see [18].

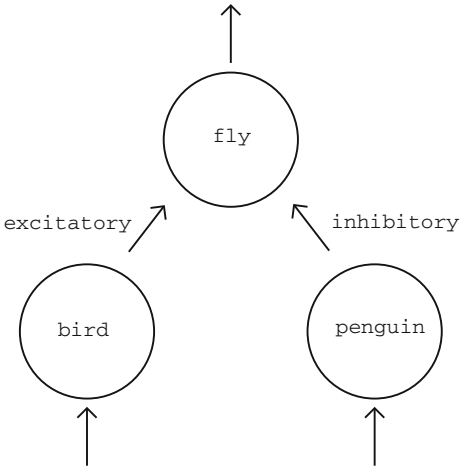

Fig. 30.1 Neural network for non-monotonic reasoning about birds

cally satisfying, the hybrid approach avoids rather than resolves questions about the essential difference between symbolic and distributed representations.

Is there some *in principle* difference between subsymbolic computation by neural networks over distributed representations and symbolic computation by Turing (or equivalent) machines? The representation theorem of McCullogh and Pitts [82] discussed above suggests differently, namely that logical theories and neural networks are essentially the same, i.e. their computational and representational properties are equivalent. Can this result be extended to a larger class of neural networks? The trick, it turns out, is to treat neural computation as non-monotonic.

It should be easy to see that some particular non-monotonic theories may be represented by neural networks. Consider the system discussed above for reasoning about birds: two input nodes (one for $Bird(x)$ and one for $Penguin(x)$) and an output node (for $Fly(x)$) are all we need to model this system with a simple neural network (Fig. 30.1). So long as there is an excitatory connection from $Bird(x)$ to $Fly(x)$ and at least as strong an inhibitory connection from $Penguin(x)$ to $Fly(x)$, this network will produce the same conclusions from the same premises as our non-monotonic theory. But this is just a specific case; a representation theorem for non-monotonic logics in neural networks would show us that for *every* non-monotonic theory, there is some neural network which computes the same conclusions. Such a theorem would demonstrate a deep computational equivalence between non-monotonic logics and neural networks.

As it turns out, representation theorems of this form have been given by several logicians coming from a variety of backgrounds and motivations. Balkenius and Gärdenfors [5] consider the inferential relationship between a fixed input to a neural network and its "resonant state," i.e. the stable activation state it reaches given that input. Partitioning the state space of these networks into *schemata*, informational components closed under conjunction, disjunction, and complementation, they show that the relation between input schemata and the corresponding resonant state satisfies

the axioms of a non-monotonic logic. Hölldobler and collaborators [64, 65] prove a representation theorem for logic programs, demonstrating that for any logic program P, a three-layer, feed-forward network can be found which computes P. Pinkas [94] provides similar results for a wider class of neural networks and penalty logic. Penalty logic is a non-monotonic logic which weights conditionals with integers representing the "penalty" if that conditional is violated. Reasoning in penalty logic involves identifying the set of propositions which minimizes the overall penalty for a set of these weighted conditionals. Blutner [14] generalizes the work of Balkenius and Gärdenfors [5] using a strategy similar to that of Pinkas. He proves a representation theorem for weight-annotated Poole systems, which differ from penalty logic in that they weight consistent sets of sentences with a positive value, the number of possible "hypotheses" (sentences) which agree with them minus the number which disagree.

These results shed some light on the formal relationship between neural networks and symbolic systems, but they also serve a practical purpose. In practical applications, an algorithm for constructing a neural network from a set of non-monotonic inference rules has computational value because it can efficiently find the fixed point which maximizes satisfaction of those rules. Unfortunately, this computational efficiency can only be achieved on a case by case basis (we discuss this point in more detail in Sect. 30.4.2). Furthermore, the practical motivations behind this research directed attention towards networks with very simple nodes and favored proofs by construction, which identify individual nodes with atomic propositions. As a resolution to the connectionism / symbolic systems debate, then, these results need to be supplemented in two ways: first, by extension to more realistic neural networks; second, by extension to the case of truly distributed representations. We conclude this section by discussing results which address these two worries.

Stenning and van Lambalgen [117] see themselves as following the tradition of Hölldobler, but focusing on neural networks which plausibly represent actual structure in the brain. They identify the work of d'Avila-Garcez and colleagues [30] as a crucial step in this direction, extending results of the kind discussed in the previous paragraphs to networks made of nodes with sigmoid activation functions, which more realistically model the behavior of actual neurons. Building on this work, and with the goal of providing a neurally plausible algorithm for their analysis of the Wason selection task, Stenning and van Lambalgen prove a representation theorem for three-valued logic programs in coupled neural networks. Three-valued logic programs can assign three distinct truth values to proposition letters: true, false, or "undecided." Here, undecided plays a role similar to negation as failure, though using three truth values allows for greater flexibility than two. Coupled neural networks are sheets of linked isomorphic networks, such that each node in the first network has a link to a corresponding node in the second network.

Theorem 30.1 ([117]). *If* P *is a 3-valued logic program and* CN(P) *is the associated coupled neural network, then the least fixed-point model of* P *corresponds to the activation of the output layer of* CN(P).

Stenning and van Lambalgen's [117] goal is neural plausibility, and they motivate their coupled neural networks by appealing to the extensive evidence for isomorphic

mappings between layers of neurons in many parts of the human brain. Nevertheless, it is not clear that any of these mappings exhibit the logical structure postulated by Stenning and van Lambalgen, nor whether this is the appropriate level of neural detail at which to model behavior on the Wason selection task.

Leitgeb identifies himself with the tradition originating with Balkenius and Gärdenfors [5], yet aims to establish a broader class of results. The theorems discussed so far typically address the relationship between a particular non-monotonic logic and a particular type of neural network. In contrast, Leitgeb [73, 74] proves a sequence of representation theorems for each system introduced by Kraus et al. [71] in distinguished classes of neural networks. These results involve inhibition nets with different constraints on internal structure, where an inhibition net is a spreading activation neural network with binary (i.e. firing or non-firing) nodes and both excitatory and inhibitory connections. Leitgeb [75] extends these results, proving representation theorems for the same logics into interpreted dynamical systems.

At the most abstract level, a dynamical system is a set of states with a transition function defined over them. An interpretation $\mathfrak{I}$ of a dynamical system is a mapping from formulas in a propositional language to regions of its state space. Leitgeb gets closure under logical connectives via the same strategy as [5], by assuming an ordering $\leq$ over informational states. If $S_{\mathfrak{I}}$ is an interpreted dynamical system, then $S_{\mathfrak{I}} \models \phi \Rightarrow \psi$ *iff* s is the resonant state of $S_{\mathfrak{I}}$ on fixed input $\mathfrak{I}(\phi)$ and $\mathfrak{I}(\psi) \leq s$. Call the set of all such conditionals $T\,S_{\mathfrak{I}}$, then

Theorem 30.2 ([75]). *If $S_{\mathfrak{I}}$ is an interpreted dynamical system, then the theory $T\,S_{\mathfrak{I}}$ is closed under the rules of the* [71] *system* **C**.

Theorem 30.3 ([75]). *If $T_{\Rightarrow}$ is a consistent theory closed under the rules of* **C**, *then there exists an interpreted dynamical system $S_{\mathfrak{I}}$ such that $T\,S_{\mathfrak{I}} = T_{\Rightarrow}$.*

Unlike the other results discussed here, Leitgeb takes pains to ensure that his representation theorems subsume the distributed case. In particular, the interpretation function may map a propositional formula to a set of nodes, i.e. distributing its representation throughout the network. From a philosophical standpoint, this result should raise questions for the debate between symbolic and connectionist approaches. Leitgeb has shown that any dynamical system performing calculations over distributed representations may be interpreted as a symbolic system performing non-monotonic inference. This result appears to show that there is no substantive difference in the representational power of symbolic systems and that of neural networks. If there is such a difference, the key to articulating it may be embedded somewhere in Leitgeb's assumptions. The key step here is the ordering over informational states of the network; it is an open question whether the states of actual networks to which connectionists attribute representational properties satisfy such an ordering. Consequently, there is work yet to be done in providing a full resolution to the symbolic systems / connectionism debate.

Even if there is no substantive difference between the representational capacities of symbolic systems and those of neural networks, there may be other principled differences between their computational powers, for instance their *computational*

efficiency. We turn next to this question, and to other applications of complexity analysis in cognitive science.

30.4 Computational Complexity and the Tractable Cognition Thesis

The Church-Turing Thesis provides a distinction between those problems which are computable and those which are not. Complexity theory supplements the computational perspective with more fine-grained distinctions, analyzing the *efficiency* of algorithms and distinguishing those problems which have efficient solutions from those which do not. Efficiency considerations can help bridge the gap between computational and algorithmic levels of analysis by turning computational hypotheses into quantitative empirical predictions. Before looking at some specific examples in Sects. 30.5 and 30.6, we first introduce the basic complexity classes and defend the Tractable Cognition Thesis, which grounds the use of complexity analysis in cognitive science.

30.4.1 Tractable Problems

Some problems, although computable, nevertheless require too much time or memory to be feasibly solved by a realistic computational device. Computational complexity theory investigates the resources (time, memory, etc.) required for the execution of algorithms and the inherent difficulty of computational problems [3, 92]. Its particular strength is that it can identify features which hold for all possible solutions to a query, thereby precisely distinguishing those problems which have efficient solutions from those which do not.

This method for analyzing queries allows us to sort them into complexity classes. The class of problems that can be computed relatively quickly, namely in polynomial time with respect to the size of the input, is called PTIME (P for short). A problem is shown to belong to this class if one can show that it can be computed by a deterministic Turing machine in polynomial time. NPTIME (NP) is the class of problems that can be computed by nondeterministic Turing machines in polynomial time. NP-hard problems are problems that are at least as difficult as any problem belonging to NP. Finally, NP-complete problems are NP-hard problems that belong to NP, hence they are the most difficult NPTIME problems.

Of course, this categorization is helpful only under the assumption that the complexity classes defined in the theory are essentially different. These inequalities are usually extremely difficult to prove. In fact, the question whether $P \neq NP$ is considered one of the seven most important open mathematical problems by the Clay Institute of Mathematics, who have offered a $1,000,000 prize for its solution. If we could show for any NP-complete problem that it is PTIME-computable, we would

have demonstrated that P = NP. Computer science generally, and computational complexity theory in particular, operate under the assumption that these two classes are different, an assumption that has proved enormously fruitful in practice.

Intuitively, a problem is NP-hard if there is no *efficient* algorithm for solving it.[5] The only way to deal with it is by using brute-force methods: searching through all possible combinations of elements over a universe. Importantly, contrary to common suggestions in the cognitive science literature (e.g. [19]), computational complexity theory has shown that many NP-hard functions cannot be efficiently approximated [4]. In other words, NP-hard problems generally lead to combinatorial explosion.

If we identify efficiency with *tractability*, computational complexity theory provides a principled method for distinguishing those problems which can reasonably be solved from those which cannot. The following thesis was formulated independently by Cobham [25] and Edmonds [35]:

> The class of practically computable problems is identical to the PTIME class, that is, the class of problems which can be computed by a deterministic Turing machine in a number of steps bounded by a polynomial function of the length of a query.

This thesis is accepted by most computer scientists. For example, Garey and Johnson [48] identify discovery of a PTIME algorithm with producing a real solution to a problem:

> Most exponential time algorithms are merely variations on exhaustive search, whereas polynomial time algorithms generally are made possible only through the gain of some deeper insight into the nature of the problem. There is wide agreement that a problem has not been "well-solved" until a polynomial time algorithm is known for it. Hence, we shall refer to a problem as intractable, if it is so hard that no polynomial time algorithm can possibly solve it.

The common belief in the Cobham-Edmonds Thesis stems from the practice of programmers. NP-hard problems often lead to algorithms which are not practically implementable even for inputs of not very large size. Assuming the Church-Turing Thesis and P $\neq$ NP, we come to the conclusion that this has to be due to intrinsic features of these problems and not to the details of current computing technology.

The substantive content of the Cobham-Edmonds thesis will become more clear if we consider some examples. Many natural problems are computable in polynomial time in terms of the length of the input, for instance calculating the greatest common divisor of two numbers or looking something up in a dictionary. However, the task of deciding whether a given classical propositional formula is satisfiable (SAT) is NP-complete [26].

Thus, even very simple logics can give rise to extremely difficult computational problems. Descriptive complexity theory deals with the relationship between logical

[5] The intuitive connection between efficiency and PTIME-computability depends crucially on considering efficiency over arbitrarily large input size n. For example, an algorithm bounded by $n^{\frac{1}{5}\log\log n}$ could be used practically even though it is not polynomial (since $n^{\frac{1}{5}\log\log n} > n^2$ only when $n > e^{e^{10}}$, [61]). Conversely, an algorithm bounded by $n^{98466506514687}$ is PTIME-computable, but even for small n it is not practical to implement. We return to these considerations in the following sections.

definability and computational complexity. The main idea is to treat classes of finite models over a fixed vocabulary as computational problems: what strength of logical language is needed to define a given class of models? The seminal result in descriptive complexity theory is Fagin's theorem, establishing a correspondence between existential second-order logic and NP:

Theorem 30.4 ([37]). Σ_1^1 *captures* NP.

This means that for every property φ, it is definable in the existential fragment of second-order logic, Σ_1^1, if and only if it can be recognized in polynomial time by a non-deterministic Turing machine [68].

What do we know about the computational complexity of reasoning with non-monotonic logics? It turns out that typically the computational complexity of non-monotonic inferences is higher than the complexity of the underlying monotonic logic. As an example, restricting the expressiveness of the language to Horn clauses allows for polynomial inference as far as classical propositional logic is concerned. However, this inference task becomes NP-hard when propositional default logic or circumscription is employed. This increase in complexity comes from the fixed-point constructions needed to provide the semantics for negation as failure and other non-monotonic reasoning rules. In general, determining minimality is NP-hard (see [16], for a survey).

But now we face a puzzle. We were attracted to non-monotonic logics because they appeared to bridge the gap between neurally plausible computational devices and formal languages. Yet the Cobham-Edmonds thesis appears to show that computing properties defined in a non-monotonic language is an intractable task. In order to resolve this puzzle, we will need to examine the relationship between complexity considerations and computational devices.

30.4.2 The Invariance Thesis

The most common model of computation used in complexity analysis is the Turing machine, yet Turing machines have a radically different structure from that of modern digital computers, and even more so from that of neural networks. In order to justify the application of results from complexity theory (e.g. that a particular problem is intractable) in cognitive science, we need to demonstrate that they hold independent of any particular implementation.

The Invariance Thesis (see e.g. [36]) states that:

> Given a "reasonable encoding" of the input and two "reasonable machines," the complexity of the computation performed by these machines on that input will differ by at most a polynomial amount.

Here, "reasonable machine" means any computing device that may be realistically implemented in the physical world. The situation here is very similar to that of the Church-Turing Thesis: although we cannot prove the Invariance Thesis, the fact that

it holds for all known physically implementable computational devices provides powerful support for it. Of course, there are well-known machines which are ruled out by the physical realizability criterion; for example, non-deterministic Turing machines and arbitrarily precise analog neural networks are not realistic in this sense. Assuming the Invariance Thesis, a task is difficult if and only if it corresponds to a function of high computational complexity, independent of the computational device under consideration, at least as long as it is reasonable.

It is worth discussing in a little more detail why neural networks fall within the scope of the Invariance Thesis. Neural networks can provide a speed-up over traditional computers because they can perform computations in parallel. However, from the standpoint of complexity theory, the difference between serial and parallel computation is irrelevant for tractability considerations. The essential point is this: any realistic parallel computing device only has a *finite* number of parallel channels for simultaneous computation. This will only provide a polynomial speed-up over a similar serial device. In particular, if the parallel device has n channels, then it should speed up computation by a factor of n (providing it can use its parallel channels with maximum efficiency). As the size of the input grows significantly larger than the number of parallel channels, the advantage in computational power for the parallel machine becomes less and less significant. In particular, the polynomial speed-up of parallel computation provides a vanishingly small advantage on NP-hard problems where the solution time grows exponentially. Therefore, the difference between symbolic and connectionist computations is negligible from the tractability perspective (see [102], particularly Sect. 6.6, for extended discussion).

These considerations should clarify and mitigate the significance of the representation theorems discussed in Sect. 30.3.2. How can we reconcile these results with the observation in Sect. 30.4.1 that fixed-point constructions are NP-hard? We provide a possible answer to this in the next section when discussing, for instance, the Fixed Parameter Tractability Thesis. Simply put, it may be the case that even though the general problem of reasoning with non-monotonic logics is intractable, in our everyday experience we only deal with a specific instance of that problem which, due to properties such as bounds on input size or statistical properties of the environment, yields tractable reasoning. Approaches such as those discussed in the next section allow us to rigorously describe properties of intractable problems that can reduce their general complexity. Then we may study whether the instances of the general problem that people routinely solve indeed constitute an easy subset of the more general problem.

30.4.3 The P-Cognition Thesis and Beyond

How can complexity considerations inform our theory of cognition? The general worry that realistic agents in a complex world must compute effective action despite limited computational resources is the problem of bounded rationality [110].

Simon [105] argued that bounded agents solve difficult problems with rough heuristics rather than exhaustive analyses of the problem space. In essence, rather than solve the hard problem presented to her by the environment, the agent solves an easier, more tractable problem which nevertheless generates an effective action. In order to apply this insight in cognitive science, it would be helpful to have a precise characterization of which class of problems can plausibly be computed by a realistic agent. The answer suggested by complexity theory is to adapt the Cobham-Edmonds Thesis:

> The class of problems which can be computed by a cognitive agent is approximated by the PTIME class, i.e. bounded agents can only solve problems with polynomial time solutions.

As far as we are aware, a version of the Cobham-Edmonds Thesis for cognitive science was first formulated explicitly in print by Frixione [46][6] and later dubbed the P-Cognition Thesis by van Rooij [102]. The P-Cognition Thesis states that a cognitive task is easy if it corresponds to a tractable problem, and hard if it corresponds to an intractable one.

The P-Cognition Thesis can be used to analyze which task an agent is plausibly solving when the world presents her with an (apparently) intractable problem. For example, Levesque [76] argues that the computational complexity of general logic problems motivates the use of Horn clauses and other tractable formalisms to obtain psychologically realistic models of human reasoning. Similarly, Tsatsos [132] emphasizes that visual search in its general bottom-up form is NP-complete. As a consequence, only visual models in which top-down information constrains visual search space are computationally plausible. In the study of categorization and subset choice, computational complexity serves as a good evaluation of psychological models [100].

Nevertheless, one might worry that the "worst-case scenario" attitude of computational complexity theory is inappropriate for analyzing the pragmatic problem-solving skills of real-world cognitive agents. Computational complexity is defined in terms of limit behavior. It answers the question: as the size of the input increases *indefinitely*, how do the running time and memory requirements of the algorithm change? The results of this analysis do not necessarily apply to computations with fixed or bounded input size.[7]

Our response to this worry is twofold. First, complexity analysis proves its value through the role it plays in fruitful ongoing programs of empirical research, such as those we discuss in Sects. 30.5 and 30.6. Second, there are natural extensions to computational complexity theory which avoid this criticism, yet rest on the same fundamental principles elucidated here. In the remainder of this section, we briefly survey some of these extensions.

Some problems are NP-hard on only a small proportion of possible inputs. Average-case complexity analysis studies the complexity of problems over randomly generated inputs, thereby allowing algorithms to be optimized for average inputs on

[6] See also [89].

[7] However, see [102].

problems for which only extraordinary inputs require exhaustive search (see e.g. [77] Chap. 4). But average-case complexity theory extends and supplements worst-case analysis; it does not, in general, replace it. For example, when deciding between competing tractable algorithmic hypotheses, average-case analysis can be used to compare their respective time-complexities with accuracy and reaction time data obtained via experimentation [57, 102]. This research does not undermine the use of complexity analysis in cognitive science, it supports and refines it.

Another extension of complexity theory which adds fine structure to the analysis of realistic agents divides a problem into parameters which can be independently analyzed for their contributions to its overall complexity. Such an analysis is useful, for example, if the intractability of a problem comes from a parameter which is usually very small, no matter how large the input (see [101] for examples). This way of thinking leads to parametrized complexity theory as a measure for the complexity of computational cognitive models. van Rooij [102] investigates this subject, proposing the Fixed-Parameter Tractability Thesis as a refinement of the P-Cognition Thesis. The FPT Thesis posits that cognitive agents can only solve problems which are tractable modulo the fixing of a parameter which is usually small in practice, and thereby subsumes many apparently "intractable" task analyses under the general P-Cognition perspective.

The proof of any formal analysis in empirical research is its success in driving predictions and increasing theoretical power. In the remainder of this chapter, we demonstrate the power of complexity analysis for driving research on quantifier processing and social cognition.

30.5 Between Computation and Algorithm: Quantifier Efficiency

Computational complexity theory has been successfully employed to probe the interaction between Marr's computational and algorithmic levels. Given a formal task analysis, complexity theory can make predictions about reaction time; conversely, abrupt changes in reaction time can provide evidence for changes in the algorithm employed on a task. For example, it is known that reaction time increases linearly when subjects are asked to count between 4 and 15 objects. Up to 3 or 4 objects the answer is immediate, so-called subitizing. For judgments involving more than 15 objects, subjects start to approximate: reaction time is constant and the number of incorrect answers increases dramatically [32]. Results such as this allow a fine-grained analysis of the algorithmic dynamics underlying a computational task.

This section illustrates how complexity theory can guide empirical research on algorithmic dynamics through the example of natural language quantifiers. Johan van Benthem's analysis of quantifiers in terms of finite state automata, when combined with complexity analysis, has produced a lively research program on quantifier processing, confirmed by empirical data on reaction times and neuroimaging.

30.5.1 Complexity and Natural Language

In Sect. 30.4.1 we saw how descriptive complexity can be used to analyze formal languages, but what about natural language? Some of the earliest research combining computational complexity with semantics can be found in Ristad's [99] *The Language Complexity Game*. Ristad carefully analyzes the comprehension of anaphoric dependencies in discourse. He considers a few approaches to describing the meaning of anaphora and proves their complexity. Finally, he concludes that the problem is inherently NP-complete and that all good formalisms accounting for it should be NP-complete as well.

More recently, Pratt-Harmann [95] shows that different fragments of natural language capture various complexity classes. More precisely, he studies the computational complexity of satisfiability problems for various fragments of natural language. Pratt-Hartmann [95] proves that the satisfiability problem for the syllogistic fragment is in PTIME, as opposed to the fragment containing relative clauses, which is NP-complete. He also describes fragments of language of even higher computational complexity, with non-copula verbs or restricted anaphora. Finally, he identifies an undecidable fragment containing unrestricted anaphora. Thorne [130] observes that the computational complexity of various fragments of English is inversely proportional to their frequency.

This work appears to challenge the P-Cognition Thesis. For, suppose that satisfiability is NP-complete, or even uncomputable, for fragments of natural language. Then it appears that we are forced not only to abandon the P-Cognition Thesis, but even to abandon the Church-Turing Thesis. This would imply that cognition involves super-Turing computation. As discussed above, this conclusion contradicts all available evidence on physical systems. While it does not defeat the possibility of formal analysis, since there is an extensive mathematics of super-Turing computation, it does complicate, and maybe defeat, the experimental investigation of the mind, which depends upon bounded and finite methods developed in continuity with the rest of empirical science. Complexity analysis provides a constructive way to move forward here. The negative results of Ristad [99] and Pratt-Hartmann [95] suggest that the brain is not solving the complete satisfiability problem when interpreting sentences of natural language. This motivates the search for polynomial time heuristics that might plausibly compute interpretations of sentences of natural language. For example, Pagin [91] tries to explain compositionality in terms of computational complexity, cognitive burden during real-time communication, and language learnability. He argues that compositionality simplifies the complexity of language communication.

30.5.2 Johan van Benthem's Semantic Automata

Johan van Benthem [7] was one of the first to emphasize and explore the tight connection between computation and meaning in natural language (see also [8]). He proposed treating "linguistic expressions as denoting certain 'procedures' performed within models for the language" [7].

Johan van Benthem formalized this proposal by identifying generalized quantifiers with automata. These *semantic automata* operate over sequences of elements from a universe, which they test for the quantified property. If they accept the sequence, the quantificational claim is true, if they reject it, the claim is false.

Before looking at some examples, let us recall the definition of a generalized quantifier. Intuitively, a generalized quantifier characterizes relations between properties over a universe; for example, "every A is B" makes a claim about the relationship between objects with property A and objects with property B. We can organize quantifiers by the number and arity of the properties required to define them. More formally:

Definition 30.1 Let $t = (n_1, \ldots, n_k)$ be a k-tuple of positive integers. A *generalized quantifier of type t* is a class Q of models of a vocabulary $\tau_t = \{R_1, \ldots, R_k\}$, such that R_i is n_i-ary for $1 \leq i \leq k$, and Q is closed under isomorphisms, i.e. if $\mathbb{M}$ and $\mathbb{M}'$ are isomorphic, then $\mathbb{M} \in \mathsf{Q}$ *iff* $\mathbb{M}' \in \mathsf{Q}$.

Therefore, formally speaking:

$$\begin{aligned}
\forall &= \{(M, A) \mid A = M\}.\\
\exists &= \{(M, A) \mid A \subseteq M \text{ and } A \neq \emptyset\}.\\
\mathsf{every} &= \{(M, A, B) \mid A, B \subseteq M \text{ and } A \subseteq B\}.\\
\mathsf{most} &= \{(M, A, B) \mid A, B \subseteq M \text{ and } card(A \cap B) > card(A - B)\}.\\
\mathsf{D_n} &= \{(M, A) \mid A \subseteq M \text{ and } card(A) \text{ is divisible by } n\}.
\end{aligned}$$

The first two examples are the standard first-order universal and existential quantifiers, both of type (1). They are classes of models with one unary predicate such that the extension of the predicate is equal to the whole universe in the case of the universal quantifier and is nonempty in the case of the existential quantifier. Quantifiers every and most of type (1, 1) are familiar from natural language semantics. Their aim is to capture the truth-conditions of sentences of the form: "Every A is B" and "Most A's are B." In other words, they are classes of models in which these statements are true. The divisibility quantifier of type (1) is a familiar mathematical quantifier.

Johan van Benthem's insight was to identify quantifier complexity with constraints on the type of automata required to verify quantifier claims over an arbitrary universe. Figures 30.2, 30.3, and 30.4 illustrate examples of these automata for some familiar quantifiers. Johan van Benthem was able to prove the following:

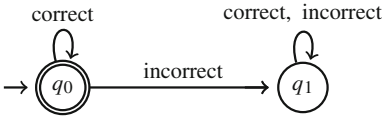

Fig. 30.2 This finite automaton checks whether every sentence in the text is grammatically correct. It inspects the text sentence by sentence, starting in the accepting state (*double circled*), q_o. As long as it does not find an incorrect sentence, it stays in the accepting state. If it finds such a sentence, then it already "knows" that the sentence is false and moves to the rejecting state, q_1, where it stays no matter what sentences come next

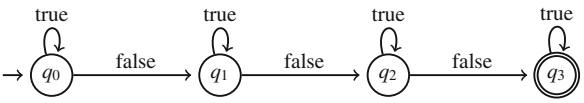

Fig. 30.3 This finite automaton recognizes whether at least 3 sentences in the text are false. This automaton needs 4 states. It starts in the rejecting state, q_0, and eventually, if the condition is satisfied, moves to the accepting state, q_3. Furthermore, notice that to recognize "at least 8" we would need 9 states, and so on

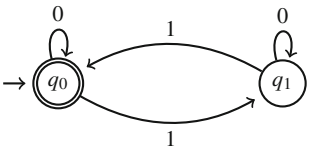

Fig. 30.4 Finite automaton checking whether the number of "1"s is even

Theorem 30.5 ([7]). *A monadic quantifier* Q *is first-order definable if and only if it can be recognized by a finite automaton without cycles.*

But this does not exhaust the class of finite automata. Finite automata with cycles can identify properties which depend on divisibility of the cardinality of the universe, such as whether the universe has an even or odd number of elements (see Fig. 30.4). Consequently, they are definable in first-order logic supplemented with special divisibility quantifiers, D_n, for each natural number n.

Theorem 30.6 ([88]). *A monadic quantifier* Q *is definable in the divisibility logic (i.e.* $FO(D_n)_{n<\omega}$*) iff it can be recognized by a finite automaton.*

However, for recognizing some higher-order quantifiers such as "less than half" or "most," we need computability models that make use of internal memory. Intuitively, to check whether sentence (1) is true we must identify the number of correct sentences and hold it in working memory to compare with the number of incorrect sentences.

(1) Most of the sentences in this chapter are grammatically correct.

Mathematically speaking, such an algorithm can be realized by an automaton supplemented with a push-down stack, a last-in / first-out form of memory.

Theorem 30.7 ([7]). *A quantifier* Q *of type (1) is definable in the arithmetic of addition iff it can be recognized by a push-down automaton.*

30.5.3 From Automata to Psycholinguistics

Johan van Benthem's formal identification of generalized quantifiers with automata can be used to generate empirical predictions about natural language processing. For example, Szymanik [120, 121] investigate whether the cognitive difficulty of quantifier processing can be assessed on the basis of the complexity of the corresponding minimal automata. He demonstrates that the computational distinction between quantifiers recognized by finite automata and those recognized by push-down automata is psychologically relevant: the more complex the automaton, the longer the reaction time and the greater the recruitment of working memory in subjects asked to solve the verification task. Szymanik and Zajenkowski [123] show that sentences with the Aristotelian quantifiers "some" and "every," corresponding to two-state finite automata, were verified in the least amount of time, while the proportional quantifiers "more than half" and "less than half" triggered the longest reaction times. When it comes to the numerical quantifiers "more than k" and "fewer than k," corresponding to finite automata with $k + 2$ states, the corresponding latencies were positively correlated with the number k. Szymanik and Zajenkowski [124, 128] explore this complexity hierarchy in concurrent processing experiments, demonstrating that during verification, the subject's working memory is qualitatively more engaged while processing proportional quantifiers than while processing numerical or Aristotelian quantifiers.

This work builds on recent neuroimaging research aimed at distinguishing the quantifier classes identified by van Benthem in terms of the neural resources they exploit. For example, McMillan and colleagues [83], in an fMRI study, show that during verification all quantified sentences recruit the right inferior parietal cortex associated with numerosity, but only proportional quantifiers recruit the prefrontal cortex, which is associated with executive resources such as working memory. These findings were later strengthened by evidence on quantifier comprehension in patients with focal neurodegenerative disease ([84]; see [67] for a survey of related results). Moreover, Zajenkowski and colleagues [140] compares the processing of natural language quantifiers in a group of patients with schizophrenia and a healthy control group. In both groups, the difficulty of the quantifiers was consistent with computational predictions. In general, patients with schizophrenia had longer reaction times. Their performance differed in accuracy only on proportional quantifiers, however, confirming the predicted qualitative increase in difficulty for quantifiers which require memory, and explainable in terms of the diminished executive control in schizophrenics.

In the next step, Szymanik [122] studied the computational complexity of multi-quantifier sentences. It turns out that there is a computational dichotomy between different readings of reciprocal sentences, for example between the following:

(2) Most of the parliament members refer indirectly to each other.
(3) Boston pitchers were sitting alongside each other.

While the first sentence is usually given an intractable NP-hard reading in which all pairs of parliament members need to be checked, the second one is interpreted by a PTIME-computable formula. This motivates the conjecture that listeners are more likely to assign readings that are simpler to compute. The psychological plausibility of this conjecture is still awaiting further investigation; however, there are already some interesting early results [104].

Szymanik [122] also asked whether multi-quantifier constructions could be computationally analyzed by extending van Benthem's framework. Threlkeld and Icard [116] give a positive answer by showing that if two quantifiers are recognizable by finite automata (push-down automata) then their iteration must also be recognizable by a finite automaton (push-down automaton). For instance, there are finite automata which recognize whether the following sentences are true:

(4) Some dots and every circle are connected.
(5) Every dot and some circles are connected.

This opens the road for further investigations into the delicate interplay between computability, expressibility, and cognitive load (see e.g. [126]).

The above results demonstrate how the P-Cognition Thesis can drive experimental practice. Differences in performance, such as in reaction times, can support a computational analysis of the task being performed [123]. Furthermore, one can track the changes in heuristics a single agent employs as the problem space changes [131].

30.6 The Complexity of Intelligent Interaction

Johan van Benthem begins his "Cognition as Interaction" [10] with a plea to cognitive scientists to move away from their myopic focus on single agents:

> It is intelligent social life which often shows truly human cognitive abilities at their best and most admirable. But textbook chapters in cognitive science mostly emphasise the apparatus that is used by single agents: reasoning, perception, memory or learning. And this emphasis becomes even stronger under the influence of neuroscience, as the only *obvious* thing that can be studied in a hard scientific manner are the brain processes inside individual bodies. Protagoras famously said that "Man is the measure of all things", and many neuroscientists would even say that it's just her brain. By contrast, this very brief chapter makes a plea for the irreducibly social side of cognition, as evidenced in the ways in which people communicate and interact. Even in physics, many bodies in interaction can form one new object, such as a Solar system. This is true all the more when we have a meeting of many minds!

We could not agree more. In this section we discuss how computational constraints can be taken seriously in the study of multi-agent social interactions. After examining a case study in detail, which illustrates how shifting from a global to a local

perspective in a multi-agent reasoning scenario can reduce the complexity of representations, we'll briefly survey the role that games can play as an empirical testing ground for logical models of social reasoning.

30.6.1 Plausible Epistemic Representations

Other chapters in this volume extensively discuss epistemic logics, and in particular Dynamic Epistemic Logic, DEL (see [12] for a recent survey). We argue that although these logics can describe the epistemic reasoning in social interactions in an elegant way, they postulate cognitively implausible representations. This leads to discussion of a recent proposal for more computationally plausible models for epistemic reasoning which repurposes models developed by van Benthem for representing quantifiers.

Let us start with a classic example: the Muddy Children puzzle. Three kids are playing outside. When they come back home, their father says: (1) "At least one of you has mud on your forehead." Then, he asks the children: (**I**) "Can you tell for sure whether you have mud on your forehead? If yes, announce your status." The children commonly know that their father never lies and that they are all sincere and perfect logical reasoners. Each child can see the mud on others but cannot see his or her own forehead. After the father asks (**I**) once, the children are silent. When he asks the question a second time, however, suddenly all, in this case two, muddy children respond that they know they have mud on their foreheads. How is that possible?

DEL models the underlying reasoning with Kripke structures characterizing the agents' uncertainty. Let us give the three children names: a, b, and c, and use propositional letters m_a, m_b, and m_c to express that the corresponding child is muddy. Possible worlds correspond to distributions of mud on children's foreheads, for example, $w_5 : m_a$ stands for a being muddy and b and c being clean in world w_5. Two worlds are joined with an edge labelled with i, for agent $i \in \{a, b, c\}$, if i cannot distinguish between the two worlds on the basis of his information; for clarity we drop the reflexive arrows for each state. The standard epistemic modeling (see e.g. [34, 38]) is depicted in Fig. 30.5; the boxed state stands for the actual world, in this case, all three children are muddy.

Now, let us recall how the reasoning process is modeled in this setting. The first public announcement has the form: (1) $m_a \vee m_b \vee m_c$, and after its announcement, (1) becomes common knowledge among the children. As a result, the children perform an update, that is, they eliminate world w_8 in which (1) is false. Then the father asks for the first time: (**I**) who among them knows his status. The children reason as follows. In world w_6, c knows that he is dirty (there is no uncertainty for c between this world and any other world in which he is clean). Therefore, if the actual world were w_6, agent c would know his state and announce it. The situation is similar for a and b in w_5 and w_7, respectively. The silence of the children after (**I**) is equivalent to the announcement that none of them know whether they are muddy. Hence, all agents eliminate those worlds that do not make such an announcement true: w_5, w_6, and w_7.

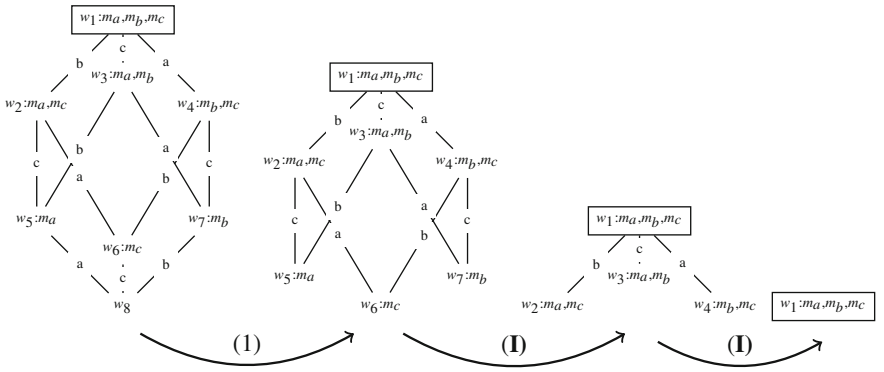

Fig. 30.5 The classical modeling of the Muddy Children puzzle. *Arrows* indicate the dynamic update to the model after the corresponding announcement

When (**I**) is announced a second time, it is again clear that if one of w_2, w_3, or w_4 were the actual world, the respective agents would have announced their knowledge. The children still do not respond, so at the start of the next round everyone knows – and in fact it is common knowledge – that the actual situation cannot be any of w_2, w_3, or w_4. Hence, they all eliminate these worlds leaving just the possibility w_1. All uncertainty disappears and they all know at the same time that they are dirty.

In spite of its logical elegance, the proposed solution is problematic from the standpoint of the Tractable Cognition Thesis. DEL flexibly models epistemic scenarios from a global perspective, but this power comes at a price: the relevant logics turn out to be very complex. First of all, DEL postulates exponential representations. Not only is it intuitively implausible that actual agents generate such exponential models of all possible scenarios, it is computationally intractable. The core of the problem is DEL's global perspective, shared with other modal logics, which assesses complexity from the modeler's point of view [115]. In fact, what we need is a new *local perspective*, a study of epistemic reasoning from the perspective of the agents involved. Such a shift in perspective can lead to logics and representations that are much more cognitively plausible [31, 55, 56]. Let's see how this applies to the Muddy Children puzzle.

In *Essays in Logical Semantics*, van Benthem [7] proposes not only computational semantics but also a geometrical representation for generalized quantifiers in the form of *number triangles*. Gierasimczuk and Szymanik [56] use number triangles to develop a new, concise logical modeling strategy for multi-agent scenarios, which focuses on the role of *quantitative* information in allowing agents to successfully converge on *qualitative* knowledge about their situation. As an example, let us consider a generalization of the Muddy Children puzzle by allowing public announcements based on an arbitrary generalized quantifier Q, for example, "Most children are muddy."

As we restrict ourselves to finite universes, we can represent all that is relevant for type (1) generalized quantifiers in a number triangle, which simply enumerates

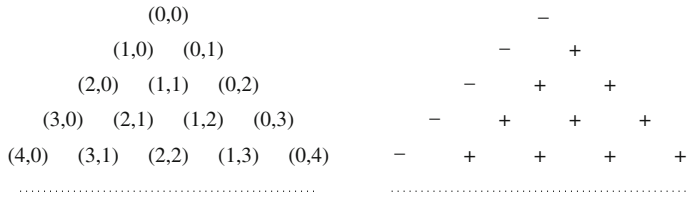

Fig. 30.6 Number triangle representing the quantifier "at least 1"

all finite models of type (1). Let U be the universe of discourse, here all the father's children, and let A be the subset of muddy children. The node labeled (k, n) stands for a model in which $|U - A| = k$ and $|A| = n$. Now, every generalized quantifier of type (1) can be represented by putting "+" at those (k, n) that belong to Q and "–" at the rest. For example, the number triangle representation of the quantifier "at least 1" is shown in Fig. 30.6. Number triangles play a crucial role in generalized quantifier theory. Gierasimczuk and Szymanik [56] interpret the pairs (k, n) as possible worlds.

Gierasimczuk and Szymanik illustrate how the number triangle may be used to derive a more concise solution for the Muddy Children puzzle. As before, we consider three agents a, b, and c. All possibilities with respect to the *size* of the set of muddy children are enumerated in the fourth level of the number triangle. Let us also assume at this point that the actual situation is that agents a and b are muddy and c is clean. Therefore, the real world is (1, 2), one child is clean and two are muddy:

(3,0) (2,1) **(1,2)** (0,3)

Now, let us focus on what the agents observe. Agent a sees one muddy child and one clean child. The same holds symmetrically for agent b. Their observational state can be encoded as (1, 1). Accordingly, the observational state of c is (0, 2). In general, if the number of agents is n, each agent can observe $n - 1$ agents. As a result, what agents observe is encoded in the third level of the number triangle:

(2,0) (1,1) (0,2)

(3,0) (2,1) **(1,2)** (0,3)

Each of the agents faces the question whether he is muddy. For example, agent a has to decide whether he should *extend* his observation state (1, 1) to the left state (2, 1) (a decides that he is clean) or to the right state (1, 2) (a decides that he is muddy). The same holds for agent b. The situation of agent c is similar: his observational state is (0, 2), which has two potential extensions, namely (1, 2) and (0, 3). In general, note that every observational state has two possible successors.

Given this representation, we can now analyze what happens in the Muddy Children scenario. Figure 30.7 represents the process, with the initial model at the top. First, the announcement is made: "At least one of you is muddy." This allows elimination of those possible states, i.e. those on the bottom row, that are not in

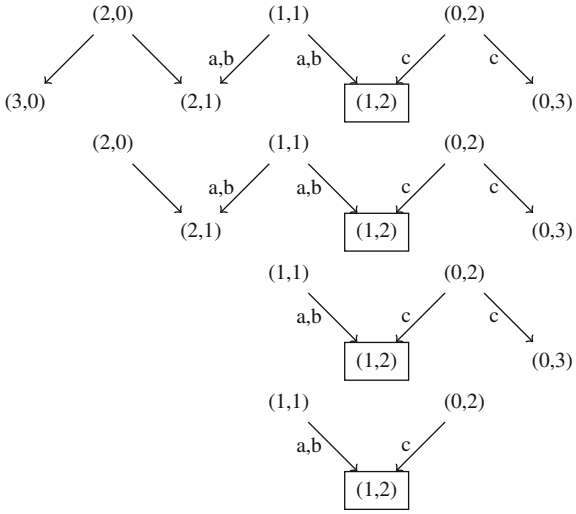

Fig. 30.7 Modeling of the Muddy Children puzzle based on number triangles

the quantifier (i.e. intersecting with the "−"s in Fig. 30.6). In this case, $(3, 0)$ is eliminated. The resulting model is the second from the top. Then the father asks: "Can you tell for sure whether or not you have mud on your forehead?" In our graph, this question means: "Does any of you have only one successor?" All agents commonly know that $(3, 0)$ has just been eliminated. Agent a considers it possible that the actual state is $(2, 1)$, i.e. that two agents are clean and one is muddy, so that he himself would have to be clean. But then he knows that there would have to be an agent whose observational state is $(2, 0)$—there has to be a muddy agent that observes two clean ones. For this hypothetical agent, the uncertainty disappeared just after the quantifier announcement (for $(2, 0)$ there is only one successor left). So, when it becomes clear that no one knows and the father asks the question again, the world $(2, 1)$ gets eliminated and the only possibility for agent a is now $(1, 2)$ via the right successor of $(1, 1)$, and this indicates that he has to be muddy. Agent b is in exactly the same situation. They both can announce that they know. Since c witnessed the whole process, he now knows that the only way for them to know was to be in $(1, 1)$, and therefore he concludes $(1, 2)$ as well.

In general, if there are n agents, we need levels n and $n + 1$ of the triangle to enumerate all possible scenarios (up to isomorphism). Therefore, the total size of the initial model is $2n + 1$. As a result, the proposed representation is exponentially more succinct than the standard DEL approach, which requires on the order of 2^n states. Strategies such as this are attractive when modeling the representations employed by realistic cognitive agents. Computational concerns constitute a plausibility test on such proposed representations.

Classic epistemic logic usually assumes an exponential representation including all possibilities (see e.g. [34]). The representation proposed here exploits a crucial feature of generalized quantifiers, namely closure under isomorphism, in order to

increase the informational power of a message relative to the observational powers of the agents. Such informational "shortcuts" can be extremely powerful, yet have so far rarely been taken into account in the epistemic literature.

30.6.2 Games and Social Cognition

There is a close relationship between epistemic logics and extensive form games. Sequences of DEL models such as those depicted in Fig. 30.5 can be strung together to form a single model for Epistemic Temporal Logic [13, 66]. These models can be pictured as branching tree structures, and a distinguished subset of them, if supplemented with a preference ordering over end states, are equivalent to extensive form games [69]. These close formal relations reflect the fact that DEL is well-suited for modeling the higher-order reasoning (e.g. involving my belief about your belief about my belief, etc.) required to explain game-theoretical arguments [135]. An important concept common to both epistemic logic and game theory is backward induction, the process of reasoning backwards from the end of the game to determine a sequence of optimal actions [9]. Backward induction can be understood as an inductive algorithm defined on a game tree. The backward induction algorithm tells us which sequence of actions will be chosen by agents that want to maximize their own payoffs, assuming common knowledge of rationality. In game-theoretical terms, backward induction calculates subgame perfect equilibria for finite extensive form games.

Games have proved a powerful tool for studying the evolution of cooperation and of social cognition [15, 111, 112, 139]. Games also play a pivotal role in designing experiments for studying social cognition in actual use [17], recently with a particular focus on higher-order social cognition, e.g. the matrix game [63], the race game [58, 62], the road game [41, 97], and the Marble Drop game [85–87].

But how well do games capture the actual dynamics of mental computation during social reasoning? There are natural ways to view games as models of cognitive processes [9], yet we might wonder how well the normative analysis of these processes in traditional game theory captures the flexible dynamics of actual human reasoning. Johan van Benthem's [8], for instance, points out that it is not the realization of perfect solutions, but rather the human ability to use interaction to dynamically recover from mistakes, which is most impressive:

> As Joerg Siekmann once said, the most admirable moments in a mathematics seminar are not when someone presents a well-oiled proof, but when he discovers a mistake and recovers on the spot. Logic should understand this dynamic behaviour, which surely involves many more mechanisms than inference …. And on that view, logic is not the static guardian of correctness, as we still find it defined in most textbooks, but rather the much more dynamic, and much more inspiring *immune system of the mind*!

These considerations motivate a closer look at the fit between game-theoretic strategies and actual human reasoning.

In this spirit, many studies have proven that the application of higher-order social reasoning among adults is far from optimal (see, for example, [63, 134]). However,

Meijering and colleagues [85, 87] report an almost perfect performance by subjects whose reasoning processes have been facilitated by step-wise training, which progresses from simple decisions without any opponent, through games that require first-order reasoning ("he plans to go right at the next trapdoor"), to games that require second-order reasoning ("he thinks that I plan to go left at the last trapdoor"). Surprisingly, an eye-tracking study of the subjects solving the game suggests that backward induction is not necessarily the strategy that participants used; they seemed instead to favour a form of "forward reasoning plus backtracking" [86].

Another application of formal methods in this spirit has been implemented by Ghosh et al. [53] and Gosh and Meijering [54]. They formulate all reasoning strategies on an experimental task in a logical language and compare ACT-R models based on each strategy with the subject's actual performance in a sequence of games on the basis of reaction times, accuracy, and eye-tracking data. These comparisons allowed them to develop a "cognitive computational model" with similar task performance to that of the human subjects. As a final example, consider the recent work of [125]. The authors successfully used structural properties of the game, namely types of turn alternation and pay-off distributions, to predict the cognitive complexity of various trials. This complexity analysis accurately predicted changes in subjects' reaction times during game play.

These examples confirm Johan van Benthem's point: it is not static solution concepts, but dynamic algorithm switching, which characterizes realistic game play. Yet, far from being immune to logical analysis, such dynamic strategic reasoning can be formalized and explored through logical methods.

30.7 Conclusion

We have examined a number of applications of logical methods in cognitive science. These methods assume the computational perspective, which treats cognitive agents as realistic machines solving information processing tasks. The value of the computational perspective is in its fruitfulness as a research program: formal analysis of an information processing task generates empirical predictions, and breakdowns in these predictions motivate revisions in the formal theory. We have illustrated this back-and-forth between logical analyses and empirical results through a number of specific examples, including quantifier processing and higher-order social reasoning.

We have also emphasized the role that logical methods can play in clarifying concepts, with a particular focus on the role of non-monotonic logics in bridging Marr's algorithmic and implementation levels and the role of complexity theory in bridging his computational and algorithmic levels. Typically, formal methods generate "in principle" results, such as the boundary between logical and illogical solutions, tractable and intractable problems, or monotonic and non-monotonic reasoning. However, these in principle results do not constitute brittle hypotheses to be confirmed or disconfirmed: it is not the case that humans are simply "logical" or "illogical." Rather, such results form a bedrock on which a nuanced analysis of

fine structure can be built. An apparent strict boundary between logical and illogical becomes an array of types of logic, an apparent strict boundary between tractable and intractable becomes a hierarchy of algorithmic types which differentially recruit processing resources. It is this fine structure which allows the subtle and fruitful interplay between logical and empirical methods.

There are morals here for both sides of the logical/empirical coin. Logicians can strengthen the relevance of their analyses for science by embracing complexity analysis. Not just formal semantics and logics of agency, but all logical models of cognitive behavior (temporal reasoning, learning, mathematical problem solving, etc.) can strengthen their relevance for empirical methods by embracing complexity analysis and the fine structure of the complexity hierarchy which rests upon it. Likewise, empirical scientists should recognize the degree of nuance a formal analysis can bring to empirical predictions: not just which task can be solved, but how quickly, and using what resources may all be predicted by an algorithmic analysis. The theme which unites these two facets of cognitive science is that *representations matter*. The content and structure of representations constrain performance on cognitive tasks. Both experiments and logical models probe the nature of these representations, and it is in converging on a single analysis through the back-and-forth of theoretical/empirical interaction that cognitive science progresses.

Acknowledgments We would like to thank Alexandru Baltag, Johan van Benthem, Peter Gärdenfors, Iris van Rooij, and Keith Stenning for many comments and suggestions. The first author would also like to thank Douwe Kiela and Thomas Icard for helpful discussions of this material; he was supported by NSF grant 1028130. The second author was supported by NWO Vici grant 277-80-001 and NWO Veni grant 639-021-232. The third author was supported by NWO Vici grant 277-80-001.

References

1. Anderson JR (1990) The adaptive character of thought: Studies in cognition. Lawrence Erlbaum, Mahwah
2. Anderson JR, Lebiere C (1998) The atomic components of thought. Lawrence Erlbaum, Mahwah
3. Arora S, Barak B (2009) Computational complexity: A modern approach (1st edn). Cambridge University Press, Cambridge
4. Ausiello G, Crescenzi P, Kann V, Gambosi G, Marchetti-Spaccamela A, Protasi M (2000) Complexity and approximation: Combinatorial optimization problems and their approximability properties. Springer, Berlin
5. Balkenius C, Gärdenfors P (1991) Nonmonotonic inference in neural networks. In: Allen JA, Fikes R, Sandewall E (eds) Proceedings of the second international conference on knowledge representation and reasoning. Morgan Kaufmann, San Mateo pp 32–39
6. Benacerraf P (1967) God, the devil, and Gödel. Monist 51:9–32
7. van Benthem J (1986) Essays in logical semantics. Reidel, Dordrecht
8. van Benthem J (1991) Language in action: Categories, lambdas and dynamic logic. North-Holland, Amsterdam (paperback edition 1995, MIT Press, Cambridge)
9. van Benthem J (2002) Extensive games as process models. J Logic Lang Inform 11(3):289–313

10. van Benthem J (2007) Cognition as interaction. In: Proceedings of symposium on cognitive foundations of interpretation. KNAW, Amsterdam, pp 27–38
11. van Benthem J (2008) Logic and reasoning: Do the facts matter? Stud Logica 88(1):67–84
12. van Benthem J (2011) Logical dynamics of information and interaction. Cambridge University Press, Cambridge
13. van Benthem J, Gerbrandy J, Hoshi T, Pacuit E (2009) Merging frameworks for interaction. J Philos Logic 38:491–526
14. Blutner R (2004) Nonmonotonic inferences and neural networks. Synthese 142:143–174
15. Bowles S, Gintis H (2011) A cooperative species: Human reciprocity and its evolution. Princeton University Press, Princeton
16. Cadoli M, Schaerf M (1993) A survey of complexity results for nonmonotonic logics. J Log Program 17(2/3&4):127–160
17. Camerer C (2003) Behavioral game theory: Experiments in strategic interaction. Princeton University Press, New Jersey
18. Chalmers D (1990) Why Fodor and Pylyshyn were wrong: The simplest refutation. In: Proceedings of the twelfth annual conference of the cognitive science society. pp 340–347
19. Chater N, Tenenbaum J, Yuille A (2006) Probabilistic models of cognition: Where next? Trends Cogn Sci 10(7):292–293
20. Cheng PW, Holyoak KJ, Nisbett RE, Oliver LM (1986) Pragmatic versus syntactic approaches to training deductive reasoning. Cogn Psychol 18:293–328
21. Chomsky N (1957) Syntactic structures. Mouton de Gruyter, The Hague
22. Clark A (1993) Associative engines. Bradford Books, Cambridge
23. Clark A (2011) Supersizing the mind: Embodiment, action, and cognitive extension. Oxford University Press, Oxford
24. Clark KL (1978) Negation as failure. In: Gallaire H, Minker J (eds) Logic and databases. Plenum Press, New York
25. Cobham J (1965) The intrinsic computational difficulty of functions. In: Proceedings of the 1964 international congress for logic, methodology, and philosophy of science. North-Holland, Amsterdam, pp 24–30
26. Cook SA (1971) The complexity of theorem-proving procedures. In STOC '71: Proceedings of the third annual ACM symposium on theory of computing. ACM Press, New York pp 151–158
27. Cooper BS (2003) Computability theory: CRC mathematics series. Chapman and Hall/CRC, London
28. Cosmides L (1989) The logic of social exchange: Has natural selection shaped how humans reason? studies with the Wason selection task. Cognition 31:187–276
29. Cosmides L, Tooby J (1992) Cognitive adaptations for social exchange. In: Barkow J, Cosmides L, Tooby J (eds) The adapted mind: Evolutionary psychology and the generation of culture. Oxford University Press, Oxford pp 163–228
30. d'Avila Garcez AS, Lamb LC, Gabbay DM (2002) Neural-symbolic learning systems: foundations and applications. Springer, London
31. Dégremont C, Kurzen L, Szymanik J (2014) Exploring the tractability border in epistemic tasks. Synthese 191(3):371–408
32. Dehaene S (1999) The number sense: How the mind creates mathematics. Oxford University Press, New York
33. Dennett DC (1984) Cognitive wheels: The frame problem of AI. In: Hookway C (ed) Minds, machines and evolution: Philosophical studies. Cambridge University Press, Cambridge
34. van Ditmarsch H, van der Hoek W, Kooi B (2007) Dynamic epistemic logic. Springer, Dordrecht
35. Edmonds J (1965) Paths, trees, and flowers. Can J Math 17:449–467
36. van Emde Boas P (1990) Machine models and simulations. In: van Leeuwen J (ed) Handbook of theoretical computer science. MIT Press, Cambridge pp 1–66
37. Fagin R (1974) Generalized first-order spectra and polynomial time recognizable sets. In: Karp R (ed) Proceedings of SIAM—AMS on complexity of computation, vol 7. American Mathematical Society pp 43–73

38. Fagin R, Halpern JY, Moses Y, Vardi MY (1995) Reasoning about knowledge. MIT Press, Cambridge
39. Feferman S (1995) Penrose's Gödelian argument. Psyche 2, online publication, available at http://www.calculemus.org/MathUniversalis/NS/10/04feferman.html
40. Fitting M (2002) Fixpoint semantics for logic programming: A survey. Theoret Comput Sci 278(1–2):25–51
41. Flobbe L, Verbrugge R, Hendriks P, Krämer I (2008) Children's application of theory of mind in reasoning and language. J Logic Lang Inform 17(4):417–442
42. Fodor JA (1983) The modularity of mind: An essay on faculty psychology. MIT Press, Cambridge
43. Fodor JA, Pylyshyn ZW (1988) Connectionism and cognitive architecture: A critical analysis. Cognition 28:3–71
44. Freeman WJ (2000) How brains make up their minds. Columbia University Press, New York
45. Freeman WJ (1972) Waves, pulses and the theory of neural masses. Prog Theor Biol 2:87–165
46. Frixione M (2001) Tractable competence. Mind Mach 11(3):379–397
47. Gärdenfors P (2000) Conceptual spaces: The geometry of thought. MIT Press, Cambridge
48. Garey MR, Johnson DS (1979) Computers and intractability. W. H. Freeman and Co., San Francisco
49. van Gelder T (1990) Compositionality: A connectionist variation on a classical theme. Cogn Sci 14:355–364
50. van Gelder T (1991) Classical questions, radical answers: Connectionism and the structure of mental representations. In: Horgan T, Tienson J (eds) connectionism and the philosophy of mind. Kluwer Academic Publishers, Dordrecht pp 355–381
51. van Gelder T (1995) What might cognition be, if not computation? J Philos 92(7):345–381
52. van Gelder T (1998) The dynamical hypothesis in cognitive science. Behav Brain Sci 21(5):615–628
53. Ghosh S, Meijering B, Verbrugge R (2010) Logic meets cognition: Empirical reasoning in games. In: Boissier O, others (eds) MALLOW, CEUR Workshop Proceedings, vol 627
54. Ghosh S, Meijering B, Verbrugge R (2014), Strategic reasoning: Building cognitive models from logical formulas. J Logic Lang Inform 23:1–29
55. Gierasimczuk N, Szymanik J (2011a) Invariance properties of quantifiers and multiagent information exchange. In: Kanazawa M, Kornai A, Kracht M, Seki H (eds) Proceedings of 12th biennial conference on the mathematics of language (MOL 12, Nara, Japan, September 6–8, 2011), vol 6878. Springer, Berlin pp 72–89
56. Gierasimczuk N, Szymanik J (2011b) A note on a generalization of the muddy children puzzle. In: Apt KR (ed) Proceedings of the 13th conference on theoretical aspects of rationality and knowledge. ACM Digital Library pp 257–264
57. Gierasimczuk N, van der Maas H, Raijmakers M (2013) Logical and psychological analysis of deductive Mastermind. J Logic Lang Inform 22(3):297–314
58. Gneezy U, Rustichini A, Vostroknutov A (2010) Experience and insight in the race game. J Econ Behav Organ 75(2):144–155
59. Gold EM (1967) Language identification in the limit. Inf Control 10:447–474
60. Griggs RA, Cox JR (1982) The elusive thematic-materials effect in Wason's selection task. Br J Psychol 73:407–420
61. Gurevich Y (1993) Feasible functions. London Math Soc Newsl 10(206):6–7
62. Hawes DR, Vostroknutov A, Rustichini A (2012) Experience and abstract reasoning in learning backward induction. Front Neurosci 6(23)
63. Hedden T, Zhang J (2002) What do you think I think you think? Strategic reasoning in matrix games. Cognition 85(1):1–36
64. Hitzler P, Hölldobler S, Seda AK (2004) Logic programs and connectionist networks. J Appl Logic 2(3):245–272
65. Hölldobler S, Kalinke Y (1994) Toward a new massively parallel computational model for logic programming. In: Proceedings of workshop on combining symbolic and connectionist processing. ECAI-94, Amsterdam

66. Hoshi T (2010) Merging DEL and ETL. J Logic Lang Inform 19:413–430
67. Hubbard E, Diester I, Cantlon J, Ansari D, van Opstal F, Troiani V (2008) The evolution of numerical cognition: From number neurons to linguistic quantifiers. J Neurosci 28(46):11,819–11,824
68. Immerman N (1999) Descriptive complexity. Springer Verlag, Berlin
69. Isaac A, Hoshi T (2011) Synchronizing diachronic uncertainty. J Logic Lang Info 20:137–159
70. Johnson-Laird PN, Legrenzi P, Legrenzi MS (1972) Reasoning and a sense of reality. Br J Psychol 63:395–400
71. Kraus S, Lehmann DJ, Magidor M (1990) Nonmonotonic reasoning, preferential models and cumulative logics. Artif Intell 44(1–2):167–207
72. Kugel P (1986) Thinking may be more than computing. Cognition 22(2):137–198
73. Leitgeb H (2003) Nonmonotonic reasoning by inhibition nets II. Int J Uncertainty, Fuzziness and Knowl Based Syst 11(2 (Supplement)):105–135
74. Leitgeb H (2001) Nonmonotonic reasoning by inhibition nets. Artif Intell 128(1–2):161–201
75. Leitgeb H (2005) Interpreted dynamical systems and qualitative laws: From neural networks to evolutionary systems. Synthese 146:189–202
76. Levesque HJ (1988) Logic and the complexity of reasoning. J Philos Logic 17(4):355–389
77. Li M, Vitányi P (2008) An introduction to Kolmogorov complexity and its applications (3rd edn). Springer, Berlin
78. Lucas JR (1961) Minds, machines and Gödel. Philosophy 36(137):112–127
79. Marr D (1983) Vision: A computational investigation into the human representation and processing visual information. W.H. Freeman, San Francisco
80. McCarthy J (1980) Circumscription–a form of non-monotonic reasoning. Artif Intell 13(1–2):27–39
81. McCarthy J, Hayes PJ (1969) Some philosophical problems from the standpoint of artificial intelligence. In: Michie D, Meltzer B (eds) Machine Intelligence, vol 4. Edinburgh University Press, Edinburgh pp 463–502
82. McCulloch WS, Pitts WH (1943) A logical calculus immanent in nervous activity. Bull Math Biophys 5:115–133
83. McMillan CT, Clark R, Moore P, Devita C, Grossman M (2005) Neural basis for generalized quantifier comprehension. Neuropsychologia 43:1729–1737
84. McMillan CT, Clark R, Moore P, Grossman M (2006) Quantifier comprehension in corticobasal degeneration. Brain Cogn 65:250–260
85. Meijering B, van Maanen L, Verbrugge R (2010) The facilitative effect of context on second-order social reasoning. In: Catrambone R, Ohlsson S (eds) Proceedings of the 32nd annual conference of the cognitive science society. Cognitive Science Society, Austin pp 1423–1428
86. Meijering B, van Rijn H, Taatgen NA, Verbrugge R (2012) What eye movements can tell about theory of mind in a strategic game. PLoS ONE 7(9):e45,961
87. Meijering B, van Rijn H, Verbrugge R (2011) I do know what you think I think: Second-order theory of mind in strategic games is not that difficult. In: Proceedings of the 33rd annual meeting of the cognitive science society, Cognitive Science Society, Boston
88. Mostowski M (1998) Computational semantics for monadic quantifiers. J Appl Non-Class Log 8:107–121
89. Mostowski M, Wojtyniak D (2004) Computational complexity of the semantics of some natural language constructions. Ann Pure Appl Logic 127(1–3):219–227
90. Newell A (1981) The knowledge level: Presidential address. AI Mag 2(2):1–20
91. Pagin P (2012) Communication and the complexity of semantics. In: Werning M, Hinzen W, Machery E (eds) Oxford handbook of compositionality. Oxford University Press, Oxford
92. Papadimitriou CH (1993) Computational complexity. Addison Wesley, Boston
93. Penrose R (1994) Shadows of the mind: A search for the missing science of consciousness. Oxford University Press, New York
94. Pinkas G (1995) Reasoning, nonmonotonicity and learning in connectionist networks that capture propositional knowledge. Artif Intell 77:203–247
95. Pratt-Hartmann I (2004) Fragments of language. J Logic Lang Inform 13(2):207–223

96. Pudlák P (1999) A note on applicability of the incompleteness theorem to human mind. Ann Pure Appl Logic 96(1–3):335–342
97. Raijmakers ME, Mandell DJ, van Es SE, Counihan M (2014) Children's strategy use when playing strategic games. Synthese 191:355–370
98. Reiter R (1980) A logic for default reasoning. Artif Intell 13:81–132
99. Ristad ES (1993) The Language complexity game (Artificial intelligence). MIT Press, Cambridge, MA,
100. van Rooij I, Stege U, Kadlec H (2005) Sources of complexity in subset choice. J Math Psychol 49(2):160–187
101. van Rooij I, Wareham T (2007) Parameterized complexity in cognitive modeling: Foundations, applications and opportunities. Comput J 51(3):385–404
102. van Rooij I (2008) The tractable cognition thesis. Cognitive Sci 32(6):939–984
103. Sandler U, Tsitolovsky L (2008) Neural cell behavior and fuzzy logic. Springer, New York
104. Schlotterbeck F, Bott O (2013) Easy solutions for a hard problem? The computational complexity of reciprocals with quantificational antecedents. J Logic Lang Inform 22(4):363–390
105. Shanahan M (1997) Solving the frame problem: A mathematical investigation of the common sense law of inertia. MIT Press, Cambridge
106. Shoham Y (1987) A semantical approach to nonmonotonic logics. In: Proceedings of the 2nd symposium on logic in computer science. CEUR Workshop Proceedings pp 275–279
107. Siegelmann HT, Sontag ED (1994) Analog computation via neural networks. Theoret Comput Sci 131:331–360
108. Siegelmann HT (1995) Computation beyond the Turing limit. Science 268:545–548
109. Siegelmann HT (2003) Neural and super-Turing computing. Mind Mach 13:103–114
110. Simon HA (1957) Models of man: Social and rational. Wiley, New York
111. Skyrms B (1996) The evolution of the social contract. Cambridge University Press, New York
112. Skyrms B (2004) The stag hunt and the evolution of social structure. Cambridge University Press, New York
113. Smolensky P (1987) Connectionism and cognitive architecture: A critical analysis. Southern J Philos 26(supplement):137–163
114. Smolensky P (1988) On the proper treatment of connectionism. Behav Brain Sci 11:1–74
115. Spaan E (1993) Complexity of modal logics. Ph.D. thesis, Universiteit van Amsterdam, Amsterdam
116. Steinert-Threlkeld S, Icard TF (2013) Iterating semantic automata. Linguist Philos 36:151–173
117. Stenning K, van Lambalgen M (2008) Human reasoning and cognitive science. MIT Press, Cambridge
118. Sternberg RJ (2002) Cognitive psychology. Wadsworth Publishing, Stamford
119. Syropoulos A (2008) Hypercomputation: Computing beyond the Church-Turing barrier (Monographs in computer science). Springer, Berlin
120. Szymanik J (2009) Quantifiers in time and space. Computational complexity of generalized quantifiers in natural language. Ph.D. thesis, University of Amsterdam, Amsterdam
121. Szymanik J (2007) A comment on a neuroimaging study of natural language quantifier comprehension. Neuropsychologia 45:2158–2160
122. Szymanik J (2010) Computational complexity of polyadic lifts of generalized quantifiers in natural language. Linguist Philos 33(3):215–250
123. Szymanik J, Zajenkowski M (2010a) Comprehension of simple quantifiers: Empirical evaluation of a computational model. Cognitive Sci 34(3):521–532
124. Szymanik J, Zajenkowski M (2011) Contribution of working memory in parity and proportional judgments. Belgian J Linguist 25(1):176–194
125. Szymanik J, Meijering B, Verbrugge R (2013a) Using intrinsic complexity of turn-taking games to predict participants' reaction times. In: Proceedings of the 35th annual conference of the cognitive science society. Cognitive Science Society, Austin

126. Szymanik J, Steinert-Threlkeld S, Zajenkowski M, Icard TF (2013b) Automata and complexity in multiple-quantifier sentence verification. In: Proceedings of the 12th International Conference on Cognitive Modeling, R. West, T. Stewart (eds.), Ottawa, Carleton University, 2013
127. Szymanik J, Verbrugge R (eds) (2013) Special issue of Journal of Logic, Language, and Information on Logic and Cognition, vol 22, Issue 3, 4
128. Szymanik J, Zajenkowski M (2010b) Quantifiers and working memory. In: Aloni M, Schulz K (eds) Amsterdam colloquium 2009, LNAI, vol 6042. Springer, Berlin, pp 456–464
129. Tenenbaum J, Kemp C, Griffiths T, Goodman N (2011) How to grow a mind: Statistics, structure, and abstraction. Science 331(6022):1279–1285
130. Thorne C (2012) Studying the distribution of fragments of English using deep semantic annotation. In: Proceedings of the ISA8 workshop
131. Tomaszewicz B (2012) Quantifiers and visual cognition. J Logic Lang Inform 22(3):335–356
132. Tsotsos J (1990) Analyzing vision at the complexity level. Behav Brain Sci 13(3):423–469
133. Tversky A, Kahneman D (1983) Extensional versus intuitive reasoning: The conjunction fallacy in probability judgment. Psychol Rev 90(4):293–315
134. Verbrugge R, Mol L (2008) Learning to apply theory of mind. J Logic Lang Inform 17(4):489–511
135. Verbrugge R (2009) Logic and social cognition: The facts matter, and so do computational models. J Philos Logic 38(6):649–680
136. Vogels TP, Abbott LF (2005) Signal propagation and logic gating in networks of integrate-and-fire neurons. J Neurosci 25(46):10786–10795
137. Wason PC (1968) Reasoning about a rule. Q J Exp Psychol 20:273–281
138. Wason PC, Shapiro D (1971) Natural and contrived experience in a reasoning problem. Q J Exp Psychol 23:63–71
139. de Weerd H, Verbrugge R, Verheij B (2013) How much does it help to know what she knows you know? An agent-based simulation study. Artif Intell 199–200: 67–92
140. Zajenkowski M, Styła R, Szymanik J (2011) A computational approach to quantifiers as an explanation for some language impairments in schizophrenia. J Commun Disord 44(6):595–600