The vertically Integrated Connectionist/Symbolic Cognitive Architecture (ICS) treats symbolic and connectionist theories as valid descriptions, at different levels of organization, of a single system: the mind/brain. How then is the wide range of data in cognitive science to be understood in ICS: via symbolic explanations, or connectionist ones?

Sorting out the subtleties underpinning this question demands a clear understanding of the multilevel structure characteristic of computational systems. These systems are analyzed here into three overall levels of organization. In ICS, the highest, functional-level description is a symbolic account of mental structure; the lowest, a physical-level description of relevant neural structure. Bridging these is an intermediate computational level that formally reduces a description corresponding to symbols to a description corresponding to neurons.

Standard reductions of symbolic computation have symbols ‘all the way down’: they reduce to a level of elementary symbolic operations that is fundamentally irreconcilable with neural computation. And standard connectionism does not provide a means of organizing elementary connectionist operations to build up symbolic computation.

ICS solves this reduction problem by splitting the computational level into two sublevels, providing two different ways of decomposing a state of the mind/brain. At the higher sublevel, a state is decomposed into superimposed patterns, corresponding to the decomposition of a symbol structure into its constituents. At the lower sublevel, a state is decomposed into individual unit activations, corresponding to the physical decomposition of a brain state at the neural level. These two decompositions are compatible because the reduction from the higher to the lower is not structure-preserving: because of distributed representations, the mapping between symbol-realizing patterns and individual unit activations is many to many. The notion of structure preservation needed here—isomorphism—is investigated, as are the implications of ICS’s split-level computational level for explaining a broad range of data that spans multiple cognitive subdisciplines and several levels of analysis. Symbolic explanation in ICS addresses many important types of phenomena, but ICS models of cognitive processes are connectionist, not symbolic (Figures 8. and 9 of Chapter 2’s ICS map).
Contents

23 Computational Levels and Integrated Connectionist/Symbolic Explanation ........................................ 503

1 Overview ............................................................................................................................................. 505
  1.1 The problem .................................................................................................................................. 505
  Box 1. Reducing symbolic computation to primitives: Symbols all the way down ...................... 506
  Box 2. The Fodorian challenge to connectionism ............................................................................ 513
  1.2 The ICS resolution .................................................................................................................... 514
  1.3 The structure of the argument and the chapter ........................................................................... 516
  Box 3. Is ICS productive? .................................................................................................................. 518

2 Levels of descriptive abstraction in computational theories ............................................................... 519
  2.1 The n-level of descriptive abstraction ......................................................................................... 520
  2.2 The c-level of descriptive abstraction ......................................................................................... 520
  Box 4. C-N isomorphism in arithmetic computers I ........................................................................ 523
  Box 5. C-N isomorphism in arithmetic computers II .................................................................... 526
  2.3 The f-level of descriptive abstraction ......................................................................................... 533
  Box 6. F-C isomorphism in arithmetic computers ........................................................................... 539

3 Levels of description in a (representational) cognitive theory ................................................................. 542
  3.1 The f-level in a cognitive theory: Mental representation ............................................................. 543
  Box 7. Explanation by mental representation .................................................................................. 545
  3.2 The c-level in a cognitive theory: Process modeling ................................................................. 548
  3.3 The n-level in a cognitive theory: Neural realization ................................................................. 549
  3.4 Process-relevance and n-isomorphism for data-dependency descriptions ................................ 549

4 Requirements for a cognitive architecture .............................................................................................. 556

5 Putative incompatibility of the architectural requirements ........................................................................ 558

6 Reconciling the architectural requirements via ICS ............................................................................. 563
  6.1 ICS's realization of F in C: From symbols and constraints to vectors and matrices ................. 564
  6.2 ICS's C-internal realization: From vectors and matrices to units and connections .................... 567
  6.3 ICS's realization of C in N: From units and connections to neurons and synapses .................. 568

7 Cognitive explanation in ICS ............................................................................................................. 569
  7.1 Core explanatory problems of cognitive science ......................................................................... 569
  Box 8. Impairment to parts versus wholes ....................................................................................... 575
  Box 9. Competence and the limits of performance ......................................................................... 578
  7.2 Cognitive data at multiple levels ................................................................................................ 580

8 Summary ............................................................................................................................................. 589

References .............................................................................................................................................. 591
1 OVERVIEW

1.1 The problem

As in many other sciences, in cognitive science one central approach to explanation is a type of reductionism. The explanatory strategy of computationalism demystifies cognition by characterizing it in terms of abstract functions and then showing how such complex cognitive functions can be reduced to a set of simple, mechanical operations. These primitive operations are not conceptually mysterious; furthermore, they are realizable in physical systems. Computational reductions give us precise ways of understanding not only what makes cognition possible at all, but also what makes cognition possible in a physical system.

A set of primitive operations, and their modes of interaction, define a computational architecture upon which a computational reduction is founded. If reduction to primitive operations is to explain how cognition is possible in a brain, then the computational architecture employed must consist of operations and interactions that are not merely simple, but simple in a way that puts them within the capabilities of neural computation.

Connectionist theory undertakes to characterize just such an architecture. Connectionist networks are, however, often criticized as naïvely simple, given contemporary knowledge of neuroscience. Such a criticism makes little sense, though, when connectionism is properly treated: it is a target of computational reduction for theories of cognition, not a tool for explaining the function of fine-grained neural structure. The real danger for a target of cognitive reduction is that it will be too computationally powerful—not too simple. Accusations of excessive computational power in connectionist architectures are thus appropriate—and they are recognized as such by connectionist theorists, who often explore architectural modifications motivated entirely by such concerns.

So while an enormous amount of difficult work remains to be done, connectionist theory appears to have given cognitive science, at the very least, an effective vehicle for launching the formal study of neural computation. At the other extreme of the reductive scale, contemporary cognitive science also possesses an effective means of precisely characterizing many important, high-level mental functions: symbolic cognitive theory. This theory is computational in that it provides a means of reduction to a set of simple, mechanical operations.

The conceptual crisis currently facing cognitive science arises from the simple fact that the elementary, mechanical operations to which symbolic theory reduces cognitive functions do not constitute a computational architecture that is consistent with today’s conception of neural computation (see Box 1). The neurally plausible architectures provided by connectionist theory do not appear to support the kinds of cognitive functions described by symbolic theory. It is of course exactly this crisis that
motivates the search for an architecture that can successfully integrate symbolic and connectionist computation—that is, solve what was called in Chapter 3 (1) the central paradox of cognitive architecture.

The tensions forming the central paradox can be analyzed as the result of incompatibilities among the demands placed on a satisfactory computational architecture for human cognition. These are summarized in (1).

(1) Architectural requirements (initial formulation)

A complete cognitive architecture must provide a formal foundation for developing, within a single unified framework, theories that promise to

a. formally explain central aspects of higher cognition, such as grammatical universals and the productivity of cognition;

b. provide a formal framework for theories of cognitive processes that
   i. explains actual human behavior and
   ii. shows explicitly how cognition can be reduced to basic computational operations;

c. reduce mental computation to neural computation, showing how the basic computational operations can be, and are, physically realized in the brain.

Simultaneously meeting all these challenges within a unified theory is the goal of ICS research.

The incompatibility of symbolic and connectionist theory has been formulated as a challenge to connectionism by Fodor and Pylyshyn (1988) (see Box 2). Connectionism, they assert, faces a serious dilemma. On one horn, connectionist computation could be used to eliminate symbols from cognitive theory; but then connectionism couldn’t explain central aspects of higher cognition, for which symbols seem necessary. On the other horn, connectionist computation could be used to literally implement symbolic computation; but then connectionism can teach us nothing new about cognition proper, only (at best) something about the neuroscience underlying the “classical” symbolic theory of cognition that we already have.

Box 1. Reducing symbolic computation to primitives: Symbols all the way down

Symbolic computation enables highly abstract conceptions of mental representations, processes, and knowledge to be characterized with formal precision. It also shows how to reduce these abstractions to extremely simple and concrete elements that can be physically realized.

How is this reduction carried out? In this box, I will consider a conventional computer running Lisp—a Lisp machine (roughly along the lines proposed in Abelson, Sussman, and Sussman 1985; see also Touretzky 1989). This (hypothetical) computer is programmed to generate sentences from a context-free grammar (see

Box 1
Box 2. The Fodorian challenge to connectionism

In a highly influential paper, Fodor and Pylyshyn (1988) argue that connectionists have two choices: implement the ‘classical’ symbolic cognitive architecture in their networks, or leave unexplained several fundamental properties of cognition that are explained by the classical architecture. These properties are as follows:

**Systematicity.** The thoughts a cognitive system is capable of entertaining are not a random collection (like the phrases in a tourist’s foreign-language phrasebook) but a systematic set (like the sentences that can be produced by a fluent speaker of a language). For example, if a cognitive system can entertain the thought expressed by *Sandy loves Kim*, then it can entertain the thought expressed by *Kim loves Sandy*.

**Productivity.** The thoughts a cognitive system is capable of entertaining are unlimited; arbitrarily complex thoughts can be created by composing simpler thoughts.

**Compositionality.** Thoughts have combinatorial structure, and the semantic content of a thought is determined compositionally by the semantic content of its parts. The thought expressed by *Sandy loves Kim* means what it does because its parts—for instance, those expressed by *Sandy* and *Kim*—mean what they do.

**Inferential coherence.** The inferences a cognitive system can draw are not an arbitrary set, but those that arise from the coherent application of certain inference rules. For example, a system able to deduce the proposition *P* from the proposition *P* & *Q* & *R* possesses an inference rule that would enable it also to deduce *P* from *P* & *Q*.

Fodor and Pylyshyn argue that these are necessary properties of a cognitive system and that they must be entailed by any adequate theory of cognition. They claim that such entailments follow from the classical symbolic architecture, and presumably would also follow from a connectionist architecture that implemented a classical architecture (although a “mere implementation” of a classical architecture would add nothing to its status as a cognitive architecture). The connectionist’s alternative to implementing a classical architecture, on Fodor and Pylyshyn’s analysis, would be some neo-Humean associationist architecture that would entail none of their fundamental properties of cognition and would thus fail a fundamental criterion of adequacy.

Fodor and McLaughlin (1990) followed up this argument in a critique of a connectionist architecture based on tensor product representations. A main claim was that these representations do not actually possess constituent structure: the alleged constituent vectors aren’t “really there” and so can’t have the sort of causal role in mental processing needed to explain Fodor and Pylyshyn’s fundamental cognitive properties.

---

4 Here is a brief synopsis of a main thread of my debate with Fodor and colleagues, with no pretense of objectivity. In Smolensky 1988, I argued that the proper role of connectionism is to explain the successes and repair the failures of symbolic theory by reducing it to a level closer to that of neurons. Fodor and Pylyshyn (1988) asserted that connectionist representations are inadequate for explaining
1.2 The ICS resolution

In this chapter, I argue that ICS furnishes a unified cognitive architecture within which all the challenges of (1) can be faced; among many other things, it resolves the alleged dilemma posed by Fodor and Pylyshyn (1988). The crucial innovation is tensor networks, or tensorial computation, the topic of Part II of this book (especially Chapter 5). Tensorial computation furnishes the bridge that allows ICS to cross the chasm separating the high ground of symbolic theories of mind from the low ground of connectionist theories of brain. On the higher side, tensorial computation mirrors—is isomorphic to—key aspects of symbolic computation. On the lower side, tensorial computation is simply parallel, distributed connectionist computation, which is plausibly isomorphic to neural computation. Crucially, I will argue, the higher and lower sides of tensorial computation are not isomorphic with one another in the relevant respects. Yet the lower is a formal realization of the higher. The result is a computational theory that reduces symbolic computation to connectionist computation: it provides a formal realization mapping from one to the other. However, a crucial transformation occurs within the level of tensorial computation, and this is what enables the symbolic to reduce to the connectionist. This transformation is achieved by distributed representations, which take distinct symbols and realize them in a common set of units, and which enable massively parallel processing. Because of this transformation, in ICS, unlike the purely symbolic architecture, it’s not ‘symbols all the way down’ (see Box 1).

An isomorphism maps each part of one system to a part of another system in such a way as to preserve the relationships among the parts within each system. Tensorial computation can be isomorphic to both symbolic and neural descriptions because it has two different modes of decomposition into parts. A tensorial representation is a vector with a special structure that enables it to be decomposed into components because they are atomic—a compound proposition such as \( p \land q \), they claimed, is represented by a single unit, connected to a different single unit for \( p \) and another unit for \( q \). In Smolensky 1987, I observed that this is an incorrect characterization of PDP connectionism, in which a composite has a distributed representation that is the superposition of the distributed representations of its parts. How this invalidates Fodor and Pylyshyn’s primary argument was spelled out in detail in Smolensky 1991. Fodor and McLaughlin (1990) claimed that it is not sufficient to show that a network can have the combinatorial properties needed for an adequate theory of mind; that a network must have such properties is necessary. In Smolensky 1995, I pointed out that a viable connectionist theory of mind is not the theory that any network is a mind; a viable theory posits a particular set of principles, and it is networks satisfying these principles that realize minds. The principles of ICS were explicitly presented, and shown to logically entail the right combinatorial properties, reducing them to neural computation. The “classical” theory that Fodor and colleagues defend does not derive those combinatorial properties at all; it merely stipulates them in its symbolic definition of mind. Against the superpositional representations of ICS, Fodor (1997) claimed that a realization of a symbolic structure must have physically separable realizations of the structure’s symbol tokens in order for the symbols to be sufficiently “real” to qualify as a legitimate basis for explaining cognitive generalizations. The falsity of this claim is documented in detail in the present chapter.

The debate through 1995 is reprinted, with enlightening commentary by the editors, in Macdonald and Macdonald 1995. See also van Gelder 1990; Cammim 1991, 1996; Mathews 1997; Phillips 2000; Aizawa 2003; and especially Horgan and Tienson 1996. For learning-oriented conceptions of connectionist systematicity, see, for example, Hadley 1994a, b, 1997a, b; Niklasson and van Gelder 1994; Phillips 1994; Aizawa 1997; Hadley and Hayward 1997; Bodén and Niklasson 2000.
stituent vectors; for example, \( s = A \otimes r_0 + B \otimes r_1 \) expresses a decomposition of \( s \) into \( A \)- and \( B \)-constituents. This decomposition is isomorphic to the decomposition of a symbol structure \( s = [A, B] \) into its \( A \)- and \( B \)-constituents. This is the sense in which the higher-level description of tensorial computation is ‘symbolic’.

At the same time, since a tensorial representation \( s \) is an activation vector, it can be decomposed into a list of numbers \((s_1, s_2, \ldots)\), each the activation value of a single connectionist unit. This lower-level decomposition of \( s \) is isomorphic to the decomposition of a biological pattern of neural activity into a list of numerical activation values of individual neurons.\(^5\)

Tensorial computation thus interfaces isomorphically with symbolic computation at its higher level, and with neural computation at its lower level. \textit{Because of distributed representations, however, the higher and lower levels are not isomorphic with each other}: the two modes of decomposition cannot be put into one-to-one correspondence (see Figure 1).\(^6\) A higher-level part like \( A \otimes r_0 \) corresponds to many lower-level parts: the activation values of the individual units in the pattern of activity \( A \otimes r_0 \). A lower-level part, the activation of unit \( k \), corresponds to many higher-level parts, since it is part of the pattern defining many constituent vectors (e.g., both \( A \otimes r_0 \) and \( B \otimes r_1 \)). The higher- and lower-level decompositions crosscut one another.

\textbf{Figure 1. Interlevel relations in ICS}

 Yet tensorial computation furnishes a fully satisfactory realization mapping from its higher- to its lower-level description. The tensor product operation, for example, has a formal definition that allows exact computation of the unit activations constituting \( A \otimes r_0 \). Likewise, the higher-level tensorial operations such as multiplying representational vectors by weight matrices are fully formally defined with respect to individual unit activations and connection weights. The representational-vector input-output function defined by multiplying an input vector by a matrix corresponds exactly to the input-output function of the connectionist network that realizes it. But the internal causal structure, described by the connectionist network description, has no

\(^5\) A one-to-one mapping from connectionist units to neurons is one rather plausible connectionist-neural isomorphism; less obvious mappings may ultimately prove more enlightening, however.

\(^6\) This figure will be elaborated in Figures 7 and 12 below.
corresponding description at the higher, ‘symbolic’ level. ICS employs symbolic explanation for cognitive functions, but connectionist explanation for cognitive processes.

1.3 The structure of the argument and the chapter

Integral to the preceding discussion is the notion level of description of a computational system. This chapter develops the general analysis of computational levels roughly synopsized in (2). (Section numbers indicate where in the chapter each topic is addressed.)

(2) Computational levels (Sections 2 and 3)

Three levels of descriptive abstraction for physical computational systems (rough)

a. The f-level: describes the functions computed
b. The c-level: describes the algorithms that compute this function
c. The n-level: describes the neural (or other physical) processes that realize the algorithm

The c-level generally consists of multiple sublevels; this is illustrated by the higher- and lower-sublevel descriptions of tensorial nets discussed above (see Figure 1).

Given formal descriptions at two levels, it may be that there is a structure-preserving one-to-one mapping from one description to the other—an isomorphism. Particularly important is isomorphism with the n-level, for that is the level at which the theory makes contact with such observable quantities as reaction times and capacity limitations. For example, in order for the number of steps in an algorithm to necessarily predict the actual time required to perform a computation, the step-by-step structure of the algorithm must be isomorphic to the moment-by-moment structure of the physical system realizing that algorithm. I will argue that the general notion process-relevance can be characterized as in (3).

(3) Process-relevance of a computational description (Section 2.2.1)

In order for a computational description to account for the time, space, or other resource requirements of a process in a physical system, that description must be isomorphic to the n-level description of the system, with respect to the structural decomposition relevant to the given resource.

With this notion, the problem posed by the incompatibility of symbolic and neural computation can be succinctly stated: symbolic theory postulates the process-relevance of its symbolic descriptions of cognitive functions and algorithms, yet the methods of reduction in symbolic theory do not establish a plausible isomorphism with the neural level. (4) states this in the language to be developed in this chapter.

(4) The central paradox of cognition (Section 5)

\[ f\text{-level} \not\equiv n\text{-level} \]

The symbolic f-level description is not isomorphic to the n-level.
Throughout this chapter, the symbol ‘≈’ will denote the isomorphism relation, and the symbol ‘∼’ the weaker relation, realization.

Exploiting the level distinctions of (2), the argument of this chapter can be summarized with the claims (5)–(9). The comments in square brackets allude to the rationale for individual claims.

The argument begins with a refinement of (1).

(5) Architectural requirements (interim formulation; final formulation in Section 4)
A complete cognitive architecture must have the following three properties:
  a. The f-level employs symbolic computation.
     [to explain the combinatorial structure of the representations and functions of higher cognition]
  b. The c-level
     i. The c-level provides a realization of the f-level.
        [required for the f- and c-level descriptions to be of the same system]
     ii. The c-level is isomorphic to the physical (n-)level.
        [the c-level of a complete cognitive theory must provide algorithms that are process-relevant—that is, isomorphic to the n-level (3)]
  c. The physical level is neural computation.
     [this must be the target of reduction in cognitive science]

(6) Putative incompatibility of the requirements (Section 5)
The three requirements (5), plus (4), are prima facie mutually contradictory.
  a. They apparently imply that the symbolic f-level must be isomorphic to the neural n-level, in contradiction to the central paradox (4).
  b. The Purely Symbolic Architecture satisfies only (5a) and (5b.i).
  c. The Eliminativist Connectionist Architecture satisfies only (5c) and (5b.ii).

(7) Reconciling the requirements via ICS (Section 6)
Appearances of incompatibility (6) notwithstanding, ICS satisfies all three requirements of (5).
  a. The f-level is symbolic.
  b. The c-level is split into higher and lower sublevels, the lower realizing the higher.
     i. The higher sublevel is isomorphic to the f-level.
     ii. The lower sublevel is connectionist, isomorphic to the n-level.
  c. The physical level is neural.

(8) ICS explanation (Section 7)
a. ICS explanation is symbolic for f-level problems.
   [cognitive functions have symbolic descriptions]
b. ICS explanation is connectionist for c-level problems.
[cognitive processes require connectionist descriptions: only the connectionist algorithms of the lower sublevel of the c-level are process-relevant]

(9) Symbolic explanation in ICS (Section 7.2)

a. Symbolic explanation in ICS addresses not only “competence data,” but important types of “performance data” as well.
[these data actually address f-level problems—explained with functions, not algorithms]

b. Symbolic explanation in ICS does not address other important types of “performance data.”
[these are uniquely c-level problems—explanations require algorithms]

The last point, (9b), identifies a major difference between ICS and the Purely Symbolic Architecture. A defining assumption of the (purely) symbolic theory is the existence of (purely) symbolic algorithms that model cognitive processes. The ICS architecture entails that in general these do not exist.

Points (2)–(9) are taken up in turn in the following sections. The lengthiest section is the next one, which attempts a rather careful and thorough analysis of levels of computational description. This analysis provides the foundation for everything that follows.

In Chapter 1, the problem of explaining the productivity of higher cognition was identified as a central challenge for cognitive science. Readers interested in how ICS explains this productivity will find in Box 3 a relatively simple synopsis of the argument, in question-and-answer format.

---

**Box 3. Is ICS productive?**

**Q1.** How does ICS explain the productivity of cognition?

A1. Symbolic combinatorics. Productivity is an f-level problem, and at the f-level, ICS is symbolic.

**Q2.** Why is ICS then not just a ‘classical architecture’ (Fodor and Pylyshyn 1988), that is, a purely symbolic architecture?

A2. The differences lie beneath the f-level. At the c-level, the ‘classical’ Purely Symbolic Architecture (PSA) assumes process-relevant symbolic algorithms; these do not exist in ICS.

**Q3.** If there are no such algorithms, how can ICS be a computational theory?

A3. ICS is computational because the symbolic combinatorics of the f-level are reduced, by fully formal realization mappings, to primitive computational operations—connectionist operations. In ICS, the process-relevant algorithms are not symbolic, and the symbolic algorithms are not process-relevant.

**Q4.** The alleged c-level distinction between ICS and PSA is a mere detail, too low-level to bear on the central issues of cognitive architecture. Who cares about what’s beneath the f-level anyway?
Section 2

2 Levels of Descriptive Abstraction in Computational Theories

This section lays the groundwork for the main argument of the chapter outlined in (5)-(9) by developing a detailed analysis of the level structure of physical computational systems (2).7

In the description of a general physical computational system $S$, three levels of abstraction can be distinguished. These level distinctions derive from those introduced in the seminal work Marr 1982, but are not identical to them. In particular, terminologically, ‘computational level’ refers to the highest level in Marr 1982 but to the middle level here. I will in fact generally avoid ‘computational level’ in favor of $c$-level, which emphasizes the technical nature of the level terminology developed here; the label ‘computational’ for the $c$-level should be construed as a rough gloss, intended more as a mnemonic than as a meaningful label. The glosses ‘functional level’ and ‘neural level’ for $f$-level and $n$-level should be taken in the same spirit.

According to the foundational premise of modern cognitive science—cognition is computation—the ultimate theory of cognition will be a complete, three-level description of a physical computational system, the brain. Available at present are only incomplete theories, which usually specify just one or two levels. Nonetheless, I will argue, these theories typically make implicit but critical assumptions concerning the unspecified levels, so in this chapter it will prove useful to consider every cognitive theory to provide three levels of description—even if, at present, some of these descriptions are rather vague, and the theory largely noncommittal about them.

Figure 2 depicts the three levels, indicating the properties I will now develop.

---

A4. Well, all of ‘classical’ symbolic cognitive theory must care: this theory is the hypothesis that there are process-relevant symbolic algorithms at the $c$-level. Thus, ICS differs crucially from PSA.

Q5. How can ICS have symbolic $f$-level accounts of productivity, but no process-relevant symbolic algorithms?

A5. The higher sublevel of the ICS $c$-level is isomorphic to the symbolic $f$-level account: symbols are computationally relevant. But only the lower $c$-sublevel is process-relevant, and at this level there are connectionist but not symbolic algorithms. The reduction from the higher to lower $c$-sublevel is accomplished by a realization mapping that is not an isomorphism; this mapping does allow a reduction from symbol structures to connectionist units—but, because representations are distributed, it is a holistic, not a part-by-part, reduction. This mapping is formal, as required for a fully computational reduction, but the lack of part-by-part correspondence means symbolic decompositions do not map onto process-relevant decompositions.

---

7 For an explication of the notion of computational level focused on algorithms, see Foster 1992.