The Foundations of Computing

Brian Cantwell Smith*

Computer and Cognitive Science Departments
Indiana University, Bloomington, in 47405 usa

Will computers ever be conscious? Is it appropriate—correct, illuminating, ethical—to understand people in computational terms? Will quantum, dna, or nanocomputers radically alter our conception of computation? How will computing affect science, the arts, intellectual history?

I have never known how to answer these questions, because I have never been sure what computation is. More than twenty-five years ago, this uncertainty led me to undertake a long-term investigation of the foundations of computer science and artificial intelligence. That study is now largely complete. My aim in this paper is to summarise some of the results.[1]

1. Project

The overall goal has been to develop a comprehensive theory of computation. Since the outset, I have assumed that such an account must meet three criteria:

1. Empirical: It must do justice to computational practice (e.g., be capable of explaining Microsoft Word: the program, its construction, maintenance, and use);
2. Conceptual: It must discharge all intellectual debts (e.g., to semantics), so that we can understand what it says, where it comes from, what it costs; and
3. Cognitive: It must provide a tenable foundation for the computational theory of mind—the thesis, sometimes known as "cognitivism," that underlies artificial intelligence and cognitive science.

The first, empirical, requirement, of doing justice to practice, helps to keep the analysis grounded in real-world examples. It is humbling, too, since the computer revolution so reliably adapts, expands, dodges expectations, and in general outstrips our theoretical grasp. But the criterion's primary advantage is to provide a vantage point from which to question the legitimacy of all extant theoretical perspectives. For I take it as a tenet that what Silicon Valley *treats* as computational, *is* computational; to deny that would be considered sufficient grounds for rejection. But no such *a priori* commitment is given to any story about computation—including the widely-held Turing-theoretic conception

[1]This paper is largely excerpted from, and is intended to serve as an introduction to, a series of books that collectively report, in detail, on the study identified in the opening paragraphs. The study of computing is presented in *The Age of Significance: Volumes I–VI* (Smith, forthcoming); the metaphysical territory to which that study leads, in *On the Origin of Objects* (Smith 1996).

 *Brian Cantwell Smith <bcsmith@phil.indiana.edu>

of computability that currently goes by the name "the theory of computation." I also reject all proposals that assume that computation can be defined. By my lights, an adequate theory must make a substantive empirical claim about what I call *computation in the wild:*[2] that eruptive body of practices, techniques, networks, machines, and behavior that has so palpably revolutionized late twentieth century life.

The second, "conceptual" criterion, that a theory own up to—and as far as possible repay—its intellectual debts, is in a way no more than standard theoretical hygiene. But it is important to highlight, for two intertwined reasons. First, it turns out that several candidate theories of computing (including the official mathematical "theory of computation" taught in computer science departments), as well as many of the reigning but largely tacit ideas about computing held in surrounding disciplines,[3] implicitly rely, without explanation, on such substantial, recalcitrant notions as representation and semantics. Second, which only makes the matter worse, there is a wide-spread tendency throughout the surrounding intellectual terrain to point to computation as a possible *theory of those very recalcitrant notions.* Unless we ferret out all such dependencies, and lay them in plain view, we run the risk of endorsing accounts that are either based on, or give rise to, vicious circularity.

The third "cognitive" condition, that an adequate theory of computation must provide a tenable foundation for a theory of mind, is of a somewhat different character. Like the second, it is more a metatheoretic requirement on the form or status of the theory than a constraint on substantive content. In committing myself to honor the criterion, however, I make no advance commitment to cognitivism's being true or false. I just want to know out what it says.

That is not to say that the content of cognitivism is left open. Cognitivism's fundamental thesis—that the mind is computational—is given substance by the first, empirical criterion. Cognitivism, that is—at least as I read it—is not a theory-laden proposal, in the sense of framing specific hypotheses about what computers are. Rather, it has more an ostensive character: that people (i.e., us) are computers in whatever way that computers (i.e., those things over there) are computers, or at least in whatever way *some* of those things are computers.

It follows that any theoretical formulation of cognitivism is doubly contingent. Thus consider Newell and Simon's (1976) popular "physical symbol system hypothesis," according to which human intelligence is claimed to consist of physical symbol manipulation, or Fodor's (1975, 1980) claim, that thinking consists of formal symbol manipulation, or Dreyfus' (1993) assertion that cognitivism (as opposed to connectionism) requires the explicit manipulation of explicit symbols. Not only do these writers make a hypothetical statement about *people*, that they are physical, formal, or explicit symbol manipulators, respectively; they do so by making a hypothetical statement about *computers*, that they are in some essential or illuminating way characterisable in the same way. Because I take the latter claim to be as subservient to empirical adequacy as the former, there are two ways in which these writers could be wrong. In

[2]Borrowed from Hutchins' *Cognition in the Wild* (1995).

[3]A notable example of such a far-from-innocent assumption is the common idea that "computation" is the fundamental notion, with a "computer" simply being any physical device that carries out a computation. It turns out, on spection, that this assumption builds in a residually dualist stance towards the mind/body problem—something I eventually want to argue against, and probably not a claim that anyone should want to build into their theories as a presumptive but inexplicit premise.

claiming that people are formal symbol manipulators, for example, Fodor would naturally be wrong if computers were formal symbol manipulators and people were not. But he would also be wrong, *even though cognitivism itself might still be true*, if computers were not formal symbol manipulators, either.

In sum, cognitive science is, like computer science, hostage to what I call the *constitutive project*: formulating a true and satisfying theory of computing that honors these three criteria. Needless to say, no one of them is easy to meet.

2. Six Construals of Computation

Some might argue that we already know what computation is. That in turn breaks into two questions: (i) is there a story—an account that people think answers the question of what computers are; and (ii) is that story right?

With regards to the first question, the answer is not *no*, but it is not a simple *yes*, either. More than one idea is at play in current theoretic discourse. Over the years I have found it convenient to distinguish six *construals* of computation, each requiring its own analysis:

1. Formal symbol manipulation (FSM): the idea, derivative from a century's work in formal logic & metamathematics, of a machine manipulating symbolic or (at least potentially) meaningful expressions without regard to their interpretation or semantic content;
2. Effective computability (EC): what can be done, and how hard it is to do it, mechanically, as it were, by an abstract analogue of a "mere machine";
3. Execution of an algorithm (ALG): what is involved, and what behavior is thereby produced, in following a set of rules or instructions, such as when making dessert;
4. Digital state machines (DSM): the idea of an automata with a finite disjoint set of internally homogeneous machine states—as parodied in the "clunk, clunk, clunk" gait of a 1950's cartoon robot;
5. Information processing (IP): what is involved in storing, manipulating, displaying, and otherwise trafficking in information, whatever that might be; and
6. Physical symbol systems (PSS): the idea, made famous by Newell and Simon, that, somehow or other, computers interact with (and perhaps also are made of) symbols in a way that depends on their mutual physical embodiment.

By far the most important step in getting to the heart of the foundational question, I believe, is to recognize that these construals are all conceptually distinct. In part because of their great familiarity (we have long since lost our innocence), and in part because "real" computers seem to exemplify more than one of them—including those often-imagined but seldom-seen Turing machines, complete with controllers, read-write heads, and long tapes—it is sometimes uncritically thought that all six can be viewed as rough synonyms, as if they were different ways of getting at the same thing. Indeed, this conflationary tendency is rampant in the literature, much of which moves around among them as if doing so were intellectually free. But that is a mistake. The supposition that *any two* of these construals amount to the same thing, let alone all six, is simply false.

Thus the formal symbol manipulation construal (fsm) is explicitly characterized in terms of a semantic or intentional aspect of computation, if for no other reason than that without some such intentional character there would be no warrant in calling it *symbol* manipulation.

The digital state machine construal (dsm), in contrast, makes no such reference to intentional properties. If a Lincoln-log contraption were digital but not symbolic, and a system manipulating continuous symbols were formal but not digital, they would be differentially counted as computational by the two construals. Not only do fsm and dsm *mean* different things, in other words; they have overlapping but distinct extensions.

The second and third construal—effective computability (ec) and algorithm execution (alg)—similarly differ on the crucial issue of semantics. Whereas the effective computability construal seems free of intentional connotation, the idea of algorithm execution, as I have characterized it, seems not only to involve rules or recipes, which presumably do mean something, but also to require some sort of understanding on the part of the agent producing the behavior.

Semantics is not the only open issue. It is similarly unclear whether the notions of "machine" and "taking an effective step" internal to the ec construal make fundamental reference to causal powers, material realization, or other physical properties, or whether, as most current theoretical discussions suggest, effective computability should be taken as an abstract mathematical notion. This is no small question. If we do not yet understand the *mind/body problem for machines*, how can we expect computational metaphors to help us in the case of people?

There are other differences among the construals. They differ on whether they inherently focus on internal structure or external input/output, for example—i.e., on whether (i) they treat computation as fundamentally *a way of being structured or constituted*, so that surface behavior is derivative (the formal symbol manipulation and digital state machine construals are of this type); or whether (ii) the *having of a particular surface behavior* is the essential locus of being computational, with questions about how that is achieved left unspecified and uncared about (effective computability is like this, with algorithm execution somewhere in the middle).

Not only must the six construal be differentiated one from another; additional distinctions must be made within each one. Thus the idea of information processing (ip)—by far the most likely characterization of computation to appear in the *Wall Street Journal*, and the idea responsible for such popular slogans as 'the Information Age' and 'the information highway'—needs to be broken down, in turn, into at least three subreadings, depending on how 'information' is understood: (i) as a *lay* notion, perhaps dating from the nineteenth-century, of something like an abstract publicly-accessible commodity, carrying a certain degree of authority; (ii) so-called "information theory," a semantics-free notion that originated with Shannon & Weaver (1949), spread out through much of cybernetics and communication theory, is implicated in Kolmogorov, Chaitin, and similar complexity measures, and has more recently been tied to notions of energy and, particularly, entropy; and (iii) the semantical notion of information advocated by Dretske (1981), Barwise & Perry (1983), Halpern (1987), and others, that in contrast to the second deals explicitly with semantic content and veridicality.

Clarifying all these issues, bringing the salient assumptions to the fore, showing where they agree and where they differ, tracing the roles they have played in the last forty years—questions like this must be part of any foundational reconstruction. But in a sense these issues are all secondary. For none have the bite of the second question raised at the beginning of the section: of whether any of the enumerated accounts is

right. Naturally, one has to say just what this means—has to answer the question, that is, "Right of what?," in order to avoid the superficial response: "Of course such and such an analysis is right; that's how computation is *defined!*" This is where the empirical criterion takes hold. More seriously, I am prepared to argue for a much more radical conclusion: that, when subjected to the empirical demands of practice and the conceptual demands of cognitive science, *all six construals fail*—for deep, overlapping, but distinct, reasons.

3. Diagnosis I: General

What is the problem? Why do these theories all fail?

The answers come at many levels. In the next section, I discuss some construal-specific problems. But a general thing can be said first. Throughout, the most celebrated difficulties have to do with semantics. It is widely (if tacitly) recognized that computation is in one way or another a symbolic or representational or information-based or semantical—i.e., as philosophers would say, an *intentional*—phenomenon.[4] Somehow or other, though in ways we do not yet understand, the states of a computer can model or simulate or represent or stand for or carry information about or signify other states in the world (or at least can be taken by people to do so). This semantical or intentional character of computation is betrayed by such phrases as *symbol* manipulation, *information* processing, programming *languages*, *knowledge representation*, *data* bases, etc. Furthermore, and this is important to understand, it is the intentionality of the computational that motivates the cognitivist thesis. The only compelling reason to suppose that we (or minds or intelligence) might be computers stems from the fact that we, too, deal with representations, symbols, meaning, information, and the like.

For someone with cognitivist leanings, therefore—as opposed, say, to an eliminativist materialist, or to some types of connectionist—it is natural to expect that a comprehensive theory of computation will have to focus on its semantical aspects. This raises problems enough. Consider just the issue of representation. In order to meet the first criterion, of empirical adequacy, a successful candidate will have to make sense of the myriad kinds of representation that saturate practical systems—from bit maps and images to knowledge representations and data bases; from caches to backup tapes; from low-level finite-element models used in simulation to high-level analytic descriptions supporting reasoning and inference, from text to graphics to audio to video to virtual reality. As well as being vast in scope, it will also have to combine decisive theoretical bite with exquisite resolution, in order to distinguish: models from implementations; analyses from simulations; and virtual machines at one level of abstraction from virtual machines at another level of abstraction, in terms of which the former may be implemented.

In order to meet the second, conceptual, criterion, moreover, any account of this profusion

[4]Although the term 'intentional' is philosophical, there are many philosophers, to say nothing of some computer and cognitive scientists, who would deny that computation is an intentional phenomenon. Reasons vary, but the most common goes something like this: (i) that computation is both *syntactic* and *formal*, where 'formal' means 'independent of semantics'; and (ii) that intentionality has fundamentally to do with semantics; and therefore (iii) that computation is thereby not intentional. I believe this is wrong, both empirically (that computation is purely syntactic) and conceptually (that being syntactic is a way of not being intentional); I also disagree that being intentional has *only* to do with semantics, which the denial requires.

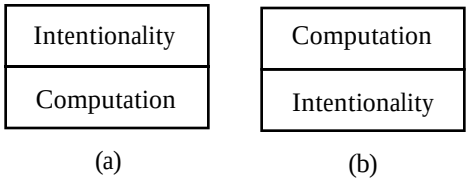

Figure 1 — Explanatory dependence

of representational practice must be grounded on, or at least defined in terms of, a theory of semantics or content, partly in order for the concomitant psychological theory to avoid vacuity or circularity, and partly so that even the computational part of the theory meet a minimal kind of naturalistic criterion: that we understand how computation is part of the natural world. This is made all the more difficult by the fact that the word 'semantics' is used in an incredible variety of different senses across the range of the intentional sciences. Indeed, in my experience it is virtually impossible, from any one location within that range, to understand the full significance of the term, so disparate is that practice *in toto*.[4]

Genuine theories of content,[5] moreover—of what it is that makes a given symbol or structure or patch of the world be *about* or *oriented towards* some other entity or structure or patch—are notoriously hard to come by.[6] Some putatively foundational construals of computation are implicitly defined in terms of just such a background theory of semantics, but do not explain what semantics is, and thus fail the second conceptual criterion. This group includes the first formal symbol manipulation construal so favored in the cognitive sciences, in spite of its superficial formulation as being "independent of semantics."[7] Other construals, such as those that view computation as the behavior of dis-

[5]In computer science, to take a salient example, the term "the semantics of α," where α is an expression or construct in a programming language, means approximately the following: the topological (as opposed to geometrical) temporal profile of the behavior to which execution of this program fragment gives rise. By 'topological' I mean that the overall temporal order of events is dictated, but that their absolute or metric time-structure (e.g., exactly how fast the program runs) is not. As a result, a program can usually be sped up, either by adjusting the code or running it on a faster processor, without, as is said, "changing the semantics."

[6]Best known are Dretske's semantic theory of information (1981), which has more generally given rise to what is known as "indicator semantics"; Fodor's "asymmetrical-dependence" theory (1987), and Millikan's "teleosemantics" or "biosemantics" (1984, 1989). For comparison among these alternatives see e.g. Fodor (1984) and Millikan (1990).

[7]Because formal symbol manipulation is usually defined as "manipulation of symbols independent of their interpretation," some people believe that the formal symbol manipulation construal of computation does not rest on a theory of semantics. But that is simply an elementary, though apparently common, conceptual mistake. As discussed further in §4, the "independence of semantics" postulated as essential to the formal symbol construal is independence at the level of the phenomenon; it is a claim about how symbol manipulation *works*. Or so at least I believe, based on many years of investigating what practitioners are actually committed to (whether it is *true*—i.e., holds of computation-in-the-wild—is a separate issue). The intuition is simple enough: that semantic properties, such as referring to the Sphinx or being true, are not of the right sort to do effective work. So they cannot be the sort of property in virtue of the manifestation of which computers run. At issue in the present discussion, in contrast, is independence *at the level of the theory* (or, perhaps, to put it less epistemically and more ontologically, independence at the level of the *types*). And here the formal symbol manipulation construal is as dependent on semantics as it is possible to be: *it is defined in terms of it*. And defining yourself in terms of something is not a way to be independent of it, as the parent of any teenager knows. Symbols must have a semantics, in other words (have an actual interpretation, or be interpretable, or whatever), in order for there to be something substantive for their formal manipulation to proceed independently of. Without a semantic character to be kept crucially in the wings, the formal symbol manipulation construal would collapse in vacuity—would degenerate into something like "the manipulation of structure" or "stuff manipulation" — i.e., materialism.

crete automata—and also, I will argue below, even if this is far from immediately evident, the recursion-theoretic one that describes such behavior as the calculation of effective functions—fail to deal with computation's semantical aspect at all, in spite of sometimes using semantical vocabulary, and so fail the first empirical criterion. In the end, one is inexorably driven to the conclusion represented in Figure 1 (b): that, in spite of the advance press, especially from cognitivist quarters, computer science, far from supplying answers to the fundamental intentional mysteries, as suggested in Figure 1 (a), must, like cognitive science, await the development of a satisfying theory of intentionality.[8]

4. Diagnosis II: Specific

So none of the six construals provide an account of semantics. Since I take computation to be semantic, that means they fail as theories of computation, as well. And that is just the beginning of the problems. They also fail for detailed structural reasons—different reasons per construal, but reasons that add up, overall, to a remarkably coherent overall picture.

In this section I summarize just a few of the problems, to convey a flavor of what is going on. In each case, to put this in context, the aim is to explicate:

1. What the construal says or comes to—what claim it makes about what it is to be a computer;
2. Where it derives from, historically;
3. Why it has been held;
4. What's right about it—what insights it gets at;
5. What is wrong with it, conceptually, empirically, and explanatorily;
6. Why it must ultimately be replaced; and
7. What about it should nevertheless be retained in a "successor" more adequate theory.

4.a. Formal Symbol Manipulation

I sometimes call the fsm construal *antisemantical*, to capture its underlying idea that computation is the "manipulation of symbols independent of their semantics." On analysis, it turns out to be motivated by two entirely different, ultimately incompatible, intuitions. The first motivation is at the level of the theory, and is reminiscent of a reductionist desire for a "semantics-free" account. It takes the fsm thesis as a claim that computation can be *described* or *analysed* in a semantics-free way. If that were true, or so the argument goes, that would go some distance towards naturalizing intentionality (as Haugeland, primary adherent of this interpretation, says, it would help "make the world safe for semantics").

There is a second motivating intuition, different in character, that holds at the level of the phenomenon. Here the ideas is simply the familiar observation that intentional phenomena, such as reasoning, hoping, or dreaming, carry on in relative independence of their subject matters or referents. Reference and truth, it is recognized, are just not the sorts of properties that can play a causal role in engendering behav-

[8]As suggested in the preceding footnote, philosophers are less likely than computer scientists to expect a theory of computation to be, or to supply, a theory of intentionality. I.e., they would not expect the metatheoretic structure to be as expected by most computer scientists and artificial intelligence researchers—namely, as indicated in Figure 1 (a), with a theory of intentionality resting on a theory of computation. But that does not mean they would necessarily agree with Figure 1 (b). Many philosophers seem to think that a theory of computation can be *independent* of a theory of intentionality. Clearly, I do not believe this is correct.

ior—essentially because they involve some sort of relational coordination with things that are *too far away* to make a difference. This relational characteristic of intentionality—something I call semantic *disconnection*—is such a deep aspect of the phenomenon that it is hard to imagine it being false. Without it, fantasy lives would be metaphysically banned; you would not be able to think about continental drift without bringing the tectonic plates along with you.

For discussion, I label the two readings of the formal symbol manipulation construal *conceptual* and *ontological*, respectively. The ontological reading is natural, familiar, and based on a deep insight. But it is too narrow. Many counter-examples can be cited against it. Space does not permit rehearsing them here.[9] Instead, to get to the heart of the matter, it helps to highlight a distinction between two kinds of "boundary" thought to be relevant or essential—indeed, often assumed *a priori*—in the analysis of computers and other intentional systems:

1. Physical: A physical boundary between the system and its surrounding environment—i.e., between "inside" and "outside"; and
2. Semantic: A semantic boundary between symbols and their referents.

In terms of these two distinctions, the ontological reading of the formal symbol manipulation construal can be understood as presuming the following two theses:

1. Alignment: That the physical and semantic boundaries line up, with all the symbols inside, all the referents outside; and
2. Isolation: That this allegedly aligned boundary is a barrier or gulf across which various forms of dependence (causal, logical, explanatory) do not reach.

[9]See *The Age of Significance: Volume II.*

The fundamental idea underlying the formal symbol manipulation thesis, that is, is that a barrier of this double allegedly aligned sort can be drawn around a computer, separating a pristine inner world of symbols—a private kingdom of thought, as it were—thought both to work (ontologically) and to be analysable (theoretically) in isolation, without distracting influence from the messy, unpredictable exterior.

Not surprisingly, the traditional motivating examples motivating the fsm construal, such as theorem proving, meet this complex condition. First, they involve internal symbols designating external situations, thereby satisfying alignment: data bases representing employee salaries, differential equations modeling the perihelion of Mercury, first order axioms designating Platonic numbers or purely abstract sets. Second, especially in the paradigmatic examples of formal axiomatizations of arithmetic and proof systems of first-order logic (and, even more especially, when those systems are understood in classical, especially model-theoretic, guise), the system is assumed to exhibit the requisite lack of interaction between the syntactic proof system and the model-theoretic interpretation, satisfying isolation. In conjunction, the two assumptions allow the familiar two-part picture of a formal system to be held: a locally contained syntactic system, on the one hand, consisting of symbols or formulae in close causal intimacy with a proof-theoretic inference regimen; and a remote realm of numbers or sets or "ur-elements," in which the symbols or formulae are interpreted, on the other. It is because the formality condition relies on both theses that the classical picture takes computation to consist exclusively of symbol–symbol transformations, carried on entirely within the confines of a machine.

The first—and easier—challenge to the anti-semantical thesis comes when one retains the

first alignment assumption, of coincident boundaries, but relaxes the second isolation claim, of no interaction. This is the classical realm of input/output, home of the familiar notion of a transducer. And it is here that one encounters the most familiar challenges to the fsm construal (such as the "robotic" reply to Searle's (1980) Chinese room argument, and Harnad's (1991) "Total Turing Test" as a measure of intelligence). Thus imagine a traditional perception system—for example one that on encounter with a mountain lion constructs a symbolic representation of the form mountain-lion-043. There is interaction (and dependence) from external world to internal representation. By the same token, an actuator system, such as one that would allow a robot to respond to a symbol of the form cross-the-street by moving from one side of the road to the other, violates the independence assumption in the other direction, from internal representation to external world.

Note, in spite of this interaction, and the consequent violation of isolation, that alignment is still preserved, in both cases: the transducer is imagined to mediate between an internal symbol and an external referent. Yet the violation of isolation alone is enough to defeat the formality condition. This is why transducers and computation are widely recognized to be uneasy bedfellows, at least when formality is at issue. It is also why, if one rests the critique at this point, defenders of the antisemantical construal are tempted to wonder, given that the operations of transducers violate *formality*, whether they should perhaps be counted as *not being computational*.[10] Given the increasing role of environmental interaction within computational practice, it is not at all clear that this would be possible, without violating the condition of empirical adequacy embraced at the outset. But it does not matter, ultimately, because the critique is only half way done.

More devastating to the fsm construal are examples that challenge the alignment thesis. It turns out, on analysis, that far from lining up on top of each other, real-world computer systems' physical and semantic boundaries *cross-cut*, in rich and productive interplay. It is not just that computers are involved in an engaged, participatory way with *external* subject matters, in other words, as suggested by some recent "situated" theorists. They are participatorily engaged in the world *as a whole*—in a world that indiscriminately includes themselves, their own internal states and processes. This integrated participatory involvement, blind to any *a priori* subject-world distinction, and concomitantly intentionally directed towards both internally and externally exemplified states of affairs, is not only architecturally essential, but is also critical, when the time comes, in establishing and grounding a system's intentional capacities.

From a purely structural point of view, four types of case are required to demonstrate this non-alignment: (i) where a symbol and referent are both internal; (ii) where a symbol is internal and its referent external; (iii) where symbol and referent are both external; and (iv) where symbol is external and referent internal. The first is exemplified in cases of quotation, meta-structural designation, window systems, e-mail, compilers, loaders, network routers, and at least arguably all programs (as opposed, say, to databases). The second, of internal symbols with external referents, can be considered as something of a theoretical (though not necessarily practical) default, as for example when one re-

[10]Thus Devitt (1991) restricts the computational thesis to what he calls "thought-thought" (t-t) transactions; for him output (t–o) and input (i–t) count as non-computational.

members the sun's setting over a lake. The third and fourth are neither more nor less than a description of ordinary written text, public writing, etc.—to say nothing of pictures, sketches, conversations, and the whole panoply of other forms of external representation. Relative to any particular system, they are distinguished by whether the subject matters of those external representations are similarly external, or are internal. The familiar red skull-and-cross-bones signifying radioactivity is external to both man and machine, and also denotes something external to man and machine, and thus belongs to the third category. To a computer or person involved, on the other hand, an account of how they work (psychoanalysis of person or machine, as it were) is an example of the fourth.

By itself, violating alignment is not enough to defeat formality. What it does accomplish, however, is to radically undermine isolation's plausibility. In particular, the antisemantical thesis was challenged not only because these examples show that the physical and semantic boundaries cross-cut, thereby undermining the alignment assumption, but because they illustrated the presence, indeed the prevalence, of effective traffic across both boundaries—between and among all the various categories in question—thereby negating isolation.

And this, in turn, shows up, for what it is, the common suggestion that transducers, because of violating the antisemantical thesis, should be ruled "out of court"—i.e., taken not to be computational (à la Devitt (1991)). It should be clear that this maneuver is ill-advised; even a bit of a cop-out. For consider what a proponent of such a move must face up to, when confronted with boundary non-alignment. *The notion of a transducer must be split in two*. I.e., someone interested in transducers would have to distinguish:

1. Physical transducers, for operations or modules that cross between the inside and outside of a system;
2. Semantic transducers, for operations or modules that mediate between symbols and their referents.

And it is this bifurcation, finally, that irrevocably defeats the antisemantical claim. For the only remotely plausible notion of transducer, in practice, is the physical one. That is what we think of when we imagine vision, touch, smell, articulation, wheels, muscles, and the like: systems that mediate between the internals of a system and the "outside" world. Transducers, that is, at least in informal imagination of practitioners, are for connecting systems to their (physical) environments.[11] What poses a challenge to the formal (antisemantical) symbol manipulation construal of computation, on the other hand, is the *semantic* transducer: those aspects of a system that involve trading between an occurrent state of affairs, on the one hand, and a representation of it, on the other. Antisemantics is challenged as much by disquotation as by driving around.

As a result, the only way to retain the ontological version of the fsm construal is to disallow (i.e., count as non-computational) the operations of semantic transducers. *But that is absurd!* It makes it clear, ultimately, that distinguishing that subset of computation that satisfies the ontological version of the antisemantical claim is not only unmotivated, solving the problem by fiat (making it uninteresting), but is a spectacularly infeasible way to draw and quarter any ac-

[11]This statement must be understood within the context of cognitive science and the philosophy of mind. It is telling that the term 'transducer' is used completely differently in engineering and biology (its natural home): to signify mechanisms that mediate changes in *medium*, not that cross the inside/outside *or* symbol/referent boundary.

tual, real-life, system. For no one who has ever built a computational system has ever found any reason to bracket reference-crossing operations, or to treat them as a distinct type. Not only that; think of how many different kinds of examples of semantic transducers one can imagine: counting, array indexing, e-mail, disquotation, error-correction circuits, linkers, loaders, simple instructions, data base access routines, pointers, reflection principles in logic, index operations into matrices, most Lisp primitives, and the like. Furthermore, to *define* a species of transducer in this semantical way, and then to remove them from consideration as not being genuinely computational, would make computation (minus the transducers) antisemantical *tautologically*. It would no longer be an interesting claim on the world that computation was antisemantical—an insight into how things are. Instead, the word 'computation' would simply be shorthand for antisemantical symbol manipulation. The question would be whether anything interesting was in this named class—and, in particular, whether this conception of computation captured the essential regularities underlying practice. And we have already seen the answer to that: it is *no*. In sum, introducing a notion of a semantical transducer solves the problem tautologically, cuts the subject matter at an unnatural joint, and fails to reconstruct practice. That is quite a lot to have against it.

Furthermore, to up the ante on the whole investigation, not only are these cases of "semantic transduction" all perfectly well-behaved; they even seem, intuitively, to be as "formal" as any other kind of operation. If that is so, then those systems either are not formal, after all, *or else the word 'formal' has never meant independence of syntax and semantics in the way that the* fsm *claim construes it*. Either way, the ontological construal does not survive.

4.b. Effective Computability

Although different in detail, the arguments against the other five construals are similar in style. In each case, I have tried to develop a staged series of counterexamples, not simply in order to show the construal false, but to serve as strong enough intuition pumps on which to base a positive alternative. Space precludes spelling out details, except for a few words about effective computability—the idea that underwrites recursion theory, complexity theory, and the official (mathematical) "theory of computation." Note, for starters, that whereas the first formal symbol manipulation construal is predominant in artificial intelligence, cognitive science, and philosophy of mind, it is this second, effective computability construal, in contrast, that underlies most theoretical and practical computer science.

Fundamentally, it is widely agreed, the theory of effective computability focuses on "what can be done by a mechanism." But two conceptual problems have clouded its proper appreciation. First, in spite of its subject matter, it is almost always characterized *abstractly*, as if it were a branch of mathematics. Second, it is imagined to be a theory defined over (for example) the numbers. Specifically, the marks on the tape of the paradigmatic Turing machine are viewed as *representations*—representations, in general, of numbers, functions, or other Turing machines.

In contrast to the received view, I argue two things. First, I claim that the theory of effective computability is fundamentally a theory about the *physical* nature of patches of the world. In underlying character it is no more "mathematical" than anything else in physics—even if we use mathematical structures to that physical reality. Second—and this is sure to be contentious—I argue that recursion theory is funda-

mentally a *theory of marks*. More specifically:

> *The representation relation, alleged to go from marks to numbers, in fact runs the other way, from numbers to marks! I.e., the truth is 180° off what we have all been led to believe.*

All sorts of evidence are cited in defense of this non-standard claim. For example:

1. Unless one understands it this way, one can solve the halting problem;
2. An analysis of history, through Turing's paper and subsequent work, especially including the development of the universal Turing machine, shows how and why the representation relation was inadvertently turned upside down;
3. The analysis makes sense of a number of otherwise-inexplicable practices, including, among other examples: (i) the use of the word "semantics" in practicing computer science to signify the behavior engendered by running a program, (ii) the rising popularity of Girard's linear logic, and (iii) the close association between theoretical computer science and constructive mathematics.

It follows from this analysis that all use of semantical vocabulary in the "official" theory of computation is meta-theoretic. As a result, *the theory is not a theory of intentional phenomena*—that is, is not a theory that deals with them *as* intentional phenomena. Whereas the formal symbol manipulation construal fails to meet its own criterion, in other words, of being "defined independent of semantics," this second construal *does* meet (at least the conceptual reading of) that condition. Ironically, however, it is in achieving that success that the recursion-theoretic tradition ultimately fails. For computation, as was said above, and as I am prepared to argue, *is* an intentional phenomenon. Which leads inexorably to the following very strong conclusion: that what goes by the name "theory of computation" fails not because it makes false claims about computation, but because *it is not a theory of computation at all.*[12]

In sum: I recommend redrawing the intellectual map. What has been called a "theory of computation" is in fact a general theory of the physical world—specifically, a theory of how hard it is, and what is required, for patches of the world in one physical configuration to change into other physical configuration. It applies to *all* physical entities, not just to computers. It is no more mathematical than the rest of physics. And thus it should be joined with physics—because in a sense it *is* physics. It is a spectacular achievement of which the century should be proud: a thoroughly-modern mathematical theory of causality.

5. Method

Similarly strong conclusions can be arrived at by pursuing each of the other construals. Rather than go into them here, I instead want to say a word about method. Specifically, about *formality*. For there is a theme underlying all six of these critiques: that part of what has blinded us to the true nature of computation has to do with the often pretheoretic assumption that *computers are formal*.

In one way or another, no matter what construal they pledge allegiance to, just about everyone thinks that computers are formal—that they manipulate symbols formally, that pro-

[12]Even if it is *derivatively* intentional. I do not believe that computation's intentionality is inherently derivative, as it happens, but even those who do think that must admit that it is an intentional phenomenon of some sort. For *derivative* does not mean *fake*. And if "derivatively intentional" is taken not to be a constraint at all, then one is forced to say just what *does* characterize computation.

grams specify formal procedures, that data structures are a kind of formalism, that computational phenomena are uniquely suited for analysis by formal methods. In fact the computer is often viewed as the crowning achievement of an entire "formal tradition"—an intellectual orientation, reaching back through Galileo to Plato, that has been epitomized in this century in the logic and metamathematics of Frege, Russell, Whitehead, Carnap, Turing, etc.

This history would suggest that formality is an essential aspect of computation. But since the outset, I have not believed that this is necessarily right. For one thing, it has never been clear what the allegiance to formality is an allegiance to. It is not as if "formal" is a technical or theory-internal predicate, after all. People may believe that developing an idea means formalizing it, and that programming languages are formal languages, and that theorem provers operate on formal axioms—but few write formal(x) in their daily equations. Moreover, a raft of different meanings and connotations lie just below the surface. Far from hurting, this apparent ambiguity has helped to cement popular consensus. Freed of the need to be strictly defined, formality has been able to serve as a lightning rod for a cluster of ontological assumptions, methodological commitments, and social and historical biases.

Because it is tacit, goes deep, has historical roots, and permeates practice, formality has been an ideal foil, over the years, with which to investigate computation.

Almost a dozen different readings of "formal" can be gleaned from informal usage: *precise*, *abstract*, *mathematical*, *a-contextual*, *digital*, *explicit*, *syntactic*, *non-semantic*, etc.[13]

They are alike in foisting recalcitrant theoretical issues onto center stage. Consider explicitness, for example, of the sort that might explain such a sentence as "for theoretical purposes we should lay out our tacit assumptions in a formal representation." Not only has explicitness—and its partner in crime, implicitness—stubbornly resisted theoretical analysis, but both notions are parasitic on something else we do not understand: general representation.[14] Or consider "a-contextual." Where is an overall theory of context in terms of which to understand what it would be to say of something (a logical representation, say) that it was not contextually dependent?

Considerations like this suggest that particular readings of formality can be most helpfully pursued within the context of the general theoretical edifices that have been constructed (more or less explicitly) in their terms. Five are particularly important:

1. The *antisemantical* reading mentioned above: the idea that a symbolic structure (representation, language, symbol system, etc.) is formal just in case it is manipulated *independent of its semantics*. Paradigmatic cases include so-called formal logic, in which it is assumed that a theorem (such as mortal(socrates)) is derived by an automatic inference regimen

[13]At one stage I asked people what they thought "formal" meant—not just computer scientists, but also mathematicians, physicists, sociologists, etc. It was clear from the replies that the term has very different connotations in different fields. Some mathematicians and logicians, for example, take it to be pejorative, in contrast to the majority of theoretical computer scientists, for whom it has almost the opposite connotation.

[14]On its own, an eggplant cannot exactly be either formal or explicit, at least not in its ordinary culinary role, since in that role it is not a representation at all. In fact the only way to make sense of calling something non-representational explicit is as short-hand for saying that it is explicitly represented (e.g., calling eggplant an explicit ingredient of moussaka as a way of saying that the recipe for moussaka mentions eggplant explicitly).

without regard to the reference, truth, or even meaning of any of its premises.

2. A closely-allied grammatical or *syntactic* reading, illustrated in such a sentence as "inference rules are defined in terms of the *formal* properties of expressions." Note that whereas the antisemantical reading is negatively characterized, this one has a positive sense.
3. A reading meaning something like *determinate* or *well-defined*—i.e., as ruling out all ambiguity and vagueness. This construal turns out to be related to a variant of the computationally familiar notion of digitality or discreteness.
4. A construal of "formal" as essentially equivalent to *mathematical*.
5. A reading that cross-cuts the other four: formality as applied to analyses or *methods*, perhaps with a derivative ontological implication that some subject matters—such as computation, perhaps?—are uniquely suited to such analytic techniques.

The first two are often treated as conceptually equivalent, but to do that is to assume that a system's syntactic and semantic properties are *necessarily disjoint*—which is almost certainly false. The relationship between the third (determinate) reading and digitality does not have so much to do with what Haugeland (1982) calls "first order digitality": the ordinary assumption that a system's states can be partitioned into a determinate set, such as that its future behavior or essence stems solely from membership in one element of that set, without any ambiguity or matter of degree. Rather, vagueness and indefiniteness (as opposed to simple continuity) are excluded by a *second-order* form of digitality—digitality at the level of concepts, in the sense of there being a binary "yes/no" fact of the matter about whether any given situation falls under (or is correctly classified in terms of) the given concept. And finally, the fourth view—that to be formal has something to do with being mathematical, or at least with being mathematically characterisable—occupies something of an ontological middle-realm between the subject-matter orientation of the first three and the methodological orientation of the fifth.

The ultimate moral for computer and cognitive science is similar to the claim made earlier about the six construals: *not one of these readings of 'formal' correctly applies to the computational case*. It can never be absolutely proved that computation is not formal, of course, given that the notion of formality is not determinately tied down. But I am prepared to argue that no standard construal of formality, including any of those just enumerated, is both (i) substantive and (ii) true of extant computational practice. Some readings reduce to vacuity, or to no more than physical realisability; others break down in internal contradiction; others survive the test of being substantial, but are demonstrably false, even of current systems. One is inescapably led to the following conclusion, in all its historical irony: that the computer, darling child of the formal tradition, outstrips the bounds of the very tradition that gave rise to it.

6. The Ontological Wall

Where does this all leave us? Over time, investigations of the sort described above, and consideration of the conclusions reached in them, have convinced me that none of the reigning theories of computation, nor the reigning methodological attitude to computation, will ever lead to an analysis strong enough to meet the two criteria laid down at the outset. More specifically, I was led to the following position, which in various ways I held for almost twenty years: I was

1. In awe of the depth, texture, scope, pluck, and impact of computational practice;
2. Critical of the inadequate state of the current theoretical art;
3. Convinced that a formal methodological stance stood in the way of getting to the heart of the computational question; and
4. Sure in my belief that what was needed, above all else, was a (situated, embodied, embedded, indexical, critical, reflexive, all sorts of other things—it changed some, over the years) theory of representation and semantics.

In line with this metatheoretic attitude, as will already have been indicated, I kept semantical and representational issues in primary theoretical focus. Since, as indicated in the last section, the official "theory of computation," derived from recursion and complexity theory, pays no attention to such intentional problems, to strike even this much of a semantical stance was to part company with the center of gravity of the received theoretical tradition.

You might think that this would be conclusion enough. And yet, in spite of the importance and magnitude of these intentional difficulties, and in spite of the detailed conclusions suggested above, I have gradually come to believe something much more sobering: that the most serious problems standing in the way of developing an adequate theory of computation are as much *ontological* as they are semantical. It is not that the semantic problems go away; they remain as challenging as ever. It is just that they are joined—on center stage, as it were—by even more demanding problems of ontology.

Except that to say "joined" is misleading, as if it were a matter of simple addition—i.e., as if now there were two problems on the table, where before there had been just one. No such luck. The two issues are inextricably entangled—a fact of obstinate theoretical and metatheoretical consequence.

A methodological consequence will illustrate the problem. Especially within the analytic tradition (by which I mean to include not just analytic philosophy, e.g. of language and mind, but most of modern science as well, complete with its formal/mathematical methods), it is traditional to analyse semantical or intentional systems, such as computers or people, under the following presupposition: (i) that one can parse or register the relevant theoretical situation in advance into a set of objects, properties, types, relations, equivalence classes, and so one (e.g., into people, heads, sentences, real-world referents, etc.)—as if this were theoretically innocuous—and then (ii), with that ontological parse in hand, go on to proclaim this or that or the other thing as an empirically justified result. Thus for example one might describe a mail-delivering robot by first describing an environment of offices, hallways, people, staircases, litter, and the like, through which it is supposed to navigate, and then, taking this characterization of its context as given, ask how or whether the creature represents routes, say, or offices, or the location of mail delivery stations.

If one adopts a reflexively critical point of view, however, as I have systematically tried to do, one is inexorably led to the following conclusion: that, in that allegedly innocent pretheoretical "set-up" stage, one is liable, even if unwittingly, to project so many presuppositions, biases, blindnesses, and advance clues about the "answer," and in general so thoroughly prefigure the target situation, without either apparent or genuine justification, that *one cannot, or at least should not, take any of the subsequent "analysis" seriously.* And that is problematic, in turn, not just because it rejects standard analyses, but be-

cause it seems to shut all inquiry down. What else can one do, after all? How can one not parse the situation in advance (since it will hardly do to merely whistle and walk away)? And if, undaunted, one were to go ahead and parse it anyway, what kind of story could possibly serve as a justification? It seems that any conceivable form of defense would devolve into another instance of the same problem.

In sum, the experience is less one of facing an ontological challenge than of running up against an ontological wall. Perhaps not quite of slamming into it, at least in my own case; recognition dawned slowly. But neither is the encounter exactly gentle. It is difficult to exaggerate the sense of frustration that can come, once the conceptual fog begins to clear, from seeing one's theoretical progress blocked by what seems for all the world to be an insurmountable metaphysical obstacle.

Like many prior claims, such as that all extant theories of computation are inadequate to reconstruct practice, this last claim, that theoretical progress is stymied for lack of an adequate theory of ontology, is a strong statement, in need of correspondingly strong defense. In my judgment, to make it perfectly plain, despite the progress that has been made so far, and despite the recommended adjustments reached in the course of the six specific analyses enumerated above, we are not going to get to the heart of computation, representation, cognition, information, semantics, or intentionality, until the ontological wall is scaled, penetrated, dismantled, or in some other way defused.

One reaction to the wall might be depression. Fortunately, the prospects are not so bleak. For starters, there is some solace in company. It is perfectly evident, once one raises one's head from the specifically computational situation and looks around, that computer scientists, cognitive scientists, and artificial intelligence researchers are not the only ones running up against the demands of ontology. Similar conclusions are being reported from many other quarters. The words are different, and the perspectives complementary, but the underlying phenomena are the same.

Perhaps the most obvious fellow travelers are literary critics, anthropologists, and other social theorists, vexed by what analytic categories to use in understanding people or cultures that, by such writers' own admission, comprehend and constitute the world using concepts alien to the theorists' own. What makes the problem particularly obvious, in these cases, is the potential for *conceptual clash* between theorist's and subject's world view—a clash that can easily seem paralysing. One's own categories are hard to justify, and reek of imperialism; it is at best presumptuous, and at worst impossible, to try to adopt the categories of one's subjects; and it is manifestly impossible to work with no concepts at all. So it is unclear how, or even whether, to proceed.

But conceptual clash, at least outright conceptual clash, is not the only form in which the ontological problem presents. Consider the burgeoning interest in "complex systems" coalescing in a somewhat renegade subdiscipline at the intersection of dynamics, theoretical biology, and artificial life. This community debates the "emergence of organisation," the units on which selection operates, the structure of self-organizing systems, the smoothness or roughness of fitness landscapes, and the like. In spite of being disciplinarily constituting, however, these discussions are conducted in the absence of adequate theories of what organization is, of what a "unit" consist in, of how "entities" arise (as opposed to how they survive), of how it is determined what predicates should figure in

characterizing a fitness landscape as rough or smooth, etc. The ontological lack is to some extent recognized in increasingly vocal calls for "theories of organization" (a theory of organization is metaphysics with a business plan). But the calls have not yet been answered.

Ontological problems have also plagued physics for years, at least since foundational issues of interpretation were thrown into relief by the developments of relativity and quantum mechanics (including the perplexing wave-particle duality, and the distinction between "classical" and "quantum" world views). They face connectionist psychologists, who, proud of having developed architectures that do not rely on the manipulation of formal symbol structures encoding high-level concepts, and thus of having thereby rejected propositional content, are nevertheless at a loss as to say what their architectures *do* represent. And then of course there are communities that tackle ontological questions directly: not just philosophy, but poetry and art, where attempts to get in, around, and under objects a have been pursued for centuries.

So there are fellow-travelers. But no one, so far as I know, has developed an alternative ontological/metaphysical proposal in sufficient detail and depth to serve as a practical foundational for a revitalized practice. Unlike at least some arguments for realism or irrealism, and also unlike some treatises *pro* or *con* this or that philosophy of science, the task is not a *meta*-metaphysical one—of merely arguing for or against a way of proceeding, if one were to proceed. Rather, the concrete demand is for a detailed, worked-out account—an account, as one says, that "goes the distance." And for this purpose, with respect to the job of developing an alternative metaphysics, the computational realm has unparalleled advantage. Midway between matter and mind, computation stands in excellent stead as a supply of concrete cases of middling complexity—what in computer science is called an appropriate "validation suite"—against which to test specific metaphysical hypotheses. "Middling" in the sense of neither being so simple as to invite caricature, nor so complex as to defy comprehension. Crucially, too, they are examples with which we are as much practically as theoretically familiar (we build systems better than we understand them).

7. Summary

Thus the ante is upped one more time. Not only must an adequate account (any account that meets the three criteria with which we started) include a theory of semantics; it must also include a theory of ontology. It is not just intentionality that is at stake, in other words; so is metaphysics. But still we are not done. For on top of these two very strong conclusions lies one final one—if anything even stronger:

Computation is not a (determinate, autonomous) subject matter.

In spite of everything I said about a "theory of computing," that is, and in spite of everything I myself thought for twenty years, I no longer believe that there is a distinct ontological category of computing or computation, one that will be the subject matter of a deep and explanatory and intellectually satisfying theory. Close and sustained analysis, that is, suggests that the things that the things that Silicon Valley calls computers, the things that perforce *are* computers, do not form a coherent intellectually delimited class. Computers turn out in the end to be rather like cars: objects of inestimable social and political and economic and personal importance, but not in and of themselves, *qua* themselves, the focus of enduring scientific or intellectual inquiry.

Needless to say, this is another extremely strong claim—one over which some readers may be tempted to rise up in arms. For if I am right, it implies that there will never be a satisfying and intellectually productive "theory of computing" of the sort I initially set out to find. So all the previous conclusions must be revised. It is not just that a theory of computation will not *supply* a theory of semantics, for example, as Newell has suggested. Or that it will not *replace* a theory of semantics. Or even that it will not *depend or rest on* a theory of semantics, as intimated in Figure 1 (b). It will do none of these things because *there will be no theory of computation at all.*

Given the weight that has been rested on the notion of computation—not just by me, or by computer science, or even by cognitive science, but by a large fraction of the surrounding intellectual landscape—this might seem like a negative conclusion. Indeed, you might conclude that I (and no doubt others as well) have spent these last twenty-five years in vain. But in fact there is no need to worry. For I firmly believe almost exactly the opposite:

> *The superficially negative conclusion (that computing is not an autonomous subject matter) makes the 20th-century arrival of computation onto the intellectual scene a far more interesting and important phenomenon than it would otherwise have been.*

On reflection, in fact, it is clear that the fact that neither computing nor computation will sustain the development of a distinct theory is by far the most exciting conclusion that the computer and cognitive sciences could possibly hope for.

Why so? Because I am not saying that computation in the wild is intrinsically a-theoretical—and thus that there will be no theory of these machines, at all, when day is done. Rather, the claim is that such theory as there is—and I take it that there remains a good chance of such a thing, as much as in any domain of human activity—will not be a theory of computation or computing. It will not be a theory of computation because *computers per se*, as I have said, do not constitute a distinct, delineated subject matter. Rather, what computers are, I now believe, and what the considerable and impressive body of practice associated with them amounts to, is neither more nor less than *the full-fledged social construction*[15] *and development of intentional artifacts*. That means that the range of experience and skills and theories and results that have been developed within computer science—astoundingly complex and far-reaching, if still inadequately articulated—is best understood as practical, synthetic, raw material for no less than full theories of causation, semantics, and ontology—i.e., for metaphysics full bore.

Where does that leave things? Substantively, it leads inexorably to the conclusion that metaphysics, ontology, and intentionality are the only integral intellectual subject matters in the vicinity of either computer or cognitive science. Methodologically, it means that our experience with constructing computational (i.e., intentional) systems may open a window onto something to which we would not otherwise have any access: the chance to witness, with our own eyes, how intentional capacities can arise in a "merely" physical mechanism. It is sobering, in retrospect, to realize that *the fact that computers are computational* has placed a major theoretical block in the way of our understanding how important they are! They are computational, of course; that much is tautological. But only when we let go of the conceit that that fact is theoretically important will we finally be able to see, *without distraction*—and thereby, perhaps, at least partially to understand—how a structured

lump of clay can sit up and think.

And so that, in these last five years, has been my project: to take, from the ashes of computational critique, enough positive morals to serve as the inspiration for, basis of, and test of, an entirely new metaphysics. A story of objects; a story of reference; a story of history; a story of society. For sheer ambition, physics does not hold a candle to computer or cognitive—or rather, as we should now call it, in order to recognise that we are dealing with something on the scale of natural science—*epistemic* or *intentional* science.

Hawking (1988) is wrong. It is we, not the natural scientists, who must develop a theory of everything.

References

Barwise, Jon & Perry, John (1983): *Situations and Attitudes*. Cambridge, Mass.: MIT Press.

Devout, Michael (1991), "Why Fodor Can't Have It Both Ways," in Loewer, Barry and Rey, Georges (eds.), *Meaning in Mind: Fodor and His Critics*, Oxford: Basil Blackwell, pp. 95–118.

Dretske, Fred I. (1981): *Knowledge and the Flow of Information*, Cambridge, Mass.: MIT Press.

Dreyfus, Herbert (1993), *What Computers Still Can't Do* Cambridge, Mass.: MIT Press.

Fodor, Jerry (1980), "Methodological Solipsism Considered as a Research Strategy in Cognitive Psychology," *Behavioral and Brain Sciences*, Vol. 3, No. 1; March 1980, 63–73. Reprinted in Fodor, Jerry: *Representations*, Cambridge, ma.: MIT Press, 1981.

——— (1984), "Semantics, Wisconsin Style," *Synthese*, 59, pp. 231–250.

——— (1987), *Psychosemantics*, Cambridge, Mass.: MIT Press.

Halpern, Joe (1987): "Using Reasoning about Knowledge to Analyze Distributed Systems," *Annual Review of Computer Science*, Vol. 2, pp. 37–68.

Harnad, Stevan (1991), "Other Bodies, Other Minds: A Machine Reincarnation of an Old Philosophical Problem," *Minds and Machines* 1, pp. 43–54.

Haugeland, John (1982), "Analog and Analog," *Philosophical Topics* (Spring 1981); reprinted in J. I. Biro & Robert W. Shahan, eds., *Mind, Brain, and Function: Essays in the Philosophy of Mind*, Norman, Oklahoma: University of Oklahoma Press (1982), pp. 213–225.

Hawking, Stephen W. (1988) *A Brief History of Time*, Toronto: Bantam Books.

Hutchins, Ed (1995), *Cognition in the Wild*, Cambridge, Mass: MIT Press.

Millikan, Ruth (1984), *Language, Thought, and Other Biological Categories*, Cambridge, Mass.: MIT Press.

——— (1989), "Biosemantics", *Journal of Philosophy* lxxxvi:6, June 1989 pp. 281–297.

——— (1990), "Compare and Contrast Dretske, Fodor, and Millikan on Teleosemantics," *Philosophical Topics* 18:2, Fall 1990, pp. 151–161.

Newell, Alan and Simon, Herbert A. (1976): "Computer Science as Empirical Inquiry: Symbols and Search," *Communications of the Association for Computing Machinery*, vol. 19 (March 1976), No. 3, pp. 113–126. Reprinted in John Haugeland, ed., *Mind Design*, Cambridge, Mass: MIT Press (1981), pp. 35–66.

Searle, John (1980): "Minds, Brains, and Programs," *Behavioral and Brain Sciences*, Vol. 3, No. 3, Sept. 1980, pp. 417–458.

Shannon, Claude E. & Weaver, Warren (1949), *The Mathematical Theory of Communication*, Urbana, Illinois: The University of Illinois Press.

Smith, Brian Cantwell (1996), *On the Origin of Objects*, Cambridge, Mass: MIT Press.

——— (forthcoming), *The Age of Significance: Volumes* i–vii.

——— *end of file* ———