PHILOSOPHICAL PSYCHOLOGY, VOL. 15, NO. 2, 2002

Hierarchy helps it work that way

LEE SPECTOR

ABSTRACT *Jerry Fodor argues, in* The mind doesn't work that way *(Cambridge, MA: MIT Press, 2000), that the computational theory of mind is undermined by the pervasive context sensitivity of human cognition. His objections can be easily met, however, by noting the properties of appropriately structured representation hierarchies.*

In *The mind doesn't work that way* (Fodor, 2000) Jerry Fodor argues that the massive modularity thesis, which asserts that most or all of cognition is implemented by encapsulated modules, fails due to the pervasive context sensitivity of human cognition. Without massive modularity, he argues, the computational theory of mind fails to account for "how the mind works" except for peripheral and relatively uninteresting corners of the mind like the module for universal grammar. Without the foundation of the computational theory of mind, he further argues, Darwinian stories about the evolution of mind are ungrounded fantasies and the recent enthusiasm for Darwinian "new synthesis" cognitive science is embarrassingly misplaced.

If it were true that the pervasive context sensitivity of human cognition is in conflict with the massive modularity thesis, then Fodor's argument would indeed be cause for concern among new synthesis cognitive scientists. But it is not true. The apparent conflict is due to the apparent computational intractability of reasoning in the face of a "ruinous holism" of context sensitivity. So far as it goes, this is intuitive computational complexity theory, but real computational complexity theory is not always so intuitive. In fact one can use well-known principles of hierarchical representation to provide tractable and yet broadly context sensitive inference services. Appropriately structured representation hierarchies can rescue the massive modularity thesis and thereby the computational theory of mind. In addition, new evidence from computational simulations shows that modules of the appropriate sort can be produced by strictly Darwinian processes. So the mind may indeed work "that way" after all, and natural selection may have been the agency that brought this about.

To flesh this out we must first clarify terminology. Countless authors in

Lee Spector, Associate Professor of Computer Science, School of Cognitive Science, Hampshire College, Amherst, MA 01002, USA, email: lspector@hampshire.edu; http://hampshire.edu/lspector

ISSN 0951-5089/print/ISSN 1465-394X/online/02/020109–09 © 2002 Taylor & Francis Ltd
DOI: 10.1080/09515080220127080

computer science, psychology, neurology and philosophy have provided countless definitions of "modularity," but Fodor's usage has a long history of which his own earlier work is an important part (particularly Fodor, 1983). In *The mind doesn't work that way* Fodor supplies a computationally explicit description of kind of modularity that matters most for the computational theory of mind:

> Imagine a computational system with a proprietary (e.g., Chomskian) database. Imagine that this device operates to map its characteristic inputs onto its characteristic outputs (in effect, to compute a function from the one to the other) and that, in the course of doing so, its informational resources are restricted to what its proprietary database contains. That is, the system is "encapsulated" with respect to information that is *not* in the database. (This might be for either, or both, of the kinds of reasons considered above: Its operations are defined with less than full generality or its informational exchanges with other processing mechanisms are constrained.) That's what I mean by a module. In my view, it's informational encapsulation, however achieved, that's at the heart of modularity. (p. 63)

The "massive modularity thesis" is then "the idea that most or all of cognition is modular" (p. 55) in this sense; it is the idea that the mind is composed entirely or almost entirely of small sub-systems, each of which has access only to its own, local database.

The importance of the massive modularity thesis derives from computational complexity considerations, the assumption being that inference can be feasible only if limits are placed on the amount of information to which a reasoning process has access:

> The totality of one's epistemic commitments is *vastly* too large a space to have to search if all one's trying to do is figure out whether, since there are clouds, it would be wise to carry an umbrella. Indeed, the totality of one's epistemic commitments is vastly too large a space to have to search *whatever* it is that one is trying to figure out. (p. 31, emphasis in original)

So the computational mind, if it is to compute anything in real time, must be modular. All of its search procedures must be limited to small, localized information sources.

But this will not do, according to Fodor, because cognition is often manifestly global, with seemingly unrelated facts coming jointly to bear in common reasoning tasks. He writes that "the mental processes thus afflicted with globality apparently include some of the ones that are most characteristic of human cognition" (p. 5), and he concludes that the most characteristic processes in human cognition, since they cannot be modular, cannot be explained by the computational theory of mind. This is particularly clear for abductive reasoning processes (reasoning to the best explanation), and in this case Fodor draws the connection to feasibility explicitly:

> Reliable abduction may require, in the limit, that the whole background of epistemic commitments be somehow brought to bear in planning and

> belief fixation. But feasible abduction requires, in practice, that not more than a small subset of even the relevant background beliefs is actually consulted. (p. 37)

The implications of the purported failure of the massive modularity thesis for new synthesis cognitive science are indirect but grave. Psychological Darwinism follows naturally from massive modularity, Fodor argues, but without massive modularity there is little to argue for it. He conjectures that the characteristic processes in human cognition may have arisen not via gradual adaptations shaped by natural selection, but rather via abrupt discontinuities in evolution (saltations).

If one insists on an absolutely strict notion of encapsulation, then the massive modularity thesis is indeed certainly wrong but it is also irrelevant; nobody doubts that various parts of the mind communicate with one another in *some* ways, and nobody's computational theory of mind rests on an assumption of absolute encapsulation. But so long as one allows for *any* communication between modules Fodor's arguments against massive modularity fail. As a result his critique of the computational theory of mind fails as well, along with the subsidiary objections to Darwinian explanations of the origins of human minds. Of course massive modularity may fail for other reasons, particularly if one requires all modules to have all of the other properties (e.g. mandatoriness) for which Fodor has sometimes argued (Fodor, 1983). But the dependence of the computational theory of mind on massive modularity in *The mind doesn't work that way* hinges only on encapsulation and not on these other purported properties of modules.

The massive modularity thesis escapes from Fodor's attack by means of a concept that has been known and appreciated in its general form since antiquity. It has been widely promoted within cognitive science in the specific form required to rebut Fodor's argument at least since Simon's (1969) *The sciences of the artificial*. It is the concept of hierarchy, which Simon calls "one of the central structural schemes that the architect of complexity uses" (p. 87). Simon's notion of hierarchy is based on the concept of "near decomposability," in which:

> Intracomponent linkages are generally stronger than intercomponent linkages. This fact has the effect of separating the high-frequency dynamics of a hierarchy—involving the internal structure of the components—from the low-frequency dynamics—involving interaction among components. (p. 106)

In a nearly decomposable system any component can be sensitive, in principle, to the contents of any other component, so long as the requirements for communication between components are small in comparison to the requirements for communication within components. In a suitably organized hierarchy the low-bandwidth intercomponent linkages will be designed to carry useful information between modules in a compact form, and this information may in some cases report broad-based "context" from the sending module.

This can be illustrated using a version of one of Fodor's own examples (which he in turn credits to Prof. Paul Casalegno) [1]. Consider a module consisting of

some number N of binary bits, and the property *ODD* that is true of this module just in case an odd number of the N bits are currently in the *ON* position. The *ODD* property is clearly sensitive to the complete state of the module (the local context), as any change to any bit in the module can change the value of *ODD*. Just as clearly, this *ODD* property is insensitive to any external (global) context. Therefore computation of the *ODD* property will be expensive if N is large, but if we assume that modules are relatively small, then *ODD* will be cheap.

Now let us consider a system of M different modules, each containing $N+2$ bits. The first N bits of each module are to be strictly local, just as those considered above. The remaining two bits, which we will call "context bits," are either to be determined by the value of a binary property of some other module or left unused (*OFF*), with the choice made for each context bit when the system is constructed. Now let us ask about a global property, called *GODD* (global *ODD*), which is true of the entire system just in case an odd number of the total $M \times N$ non-context bits are currently in the *ON* position. As Fodor points out, the contribution that each module makes to *GODD* might be considered context-independent [2], but "*the result of its contributing what it contributes*" is surely context sensitive (p. 27). Can this global, context sensitive property be maintained with reasonable cost (in time and space) even if M, the number of modules, is very large?

The answer is "yes," and the algorithm is a trivial application of the concept of hierarchy. Designate one of the modules (call it the *GODD* module) to compute *GODD* simply by computing *ODD* (on the basis of all $N+2$ of its bits), and specify that its two context bits are determined by the *ODD* property of two other modules. The context bits of these two other modules should likewise be determined by the *ODD* property of *other* modules, and so on recursively until there are no remaining modules (at which point all remaining context bits will remain unused = *OFF*). The resulting pattern of connectivity forms a pyramid-like hierarchy with the *GODD* module at the top.

Notice that the *ODD* property will be true of the *GODD* module in exactly those cases in which the *GODD* property is true of the global system. Notice also that the amount of communication required to re-compute the *GODD* property after a bit-flip anywhere in the system is proportionate only to $\log(M)$, not to M. This means that we can scale M up exponentially while suffering only a linear increase in the time to re-compute the global property, making the maintenance of *GODD* tractable as long as the local computations of *ODD* are tractable, even though it is globally context sensitive [3]. Notice also that other *GODD*-like properties can be simultaneously maintained over the same system for the modest cost of a few extra bits per module, and that the hierarchies for different globally sensitive properties may run in different directions. It is true that it's a long distance from *GODD* to the properties like centrality and simplicity that most trouble Fodor, and it's true that there are many *other* unsolved problems regarding centrality and simplicity, but the access to global information required for maintaining all of these properties can be achieved by means of the same hierarchical techniques.

Nothing in the computational theory of mind is in conflict with the notion that globally context sensitive properties are implemented hierarchically, as the *GODD*

property above, and indeed most work in artificial intelligence assumes hierarchical organization as a matter of course. Hierarchy can, in principle, efficiently support the global context sensitivity that Fodor worries will undermine the computational theory of mind. There is much more work to be done in showing how minds are *really* organized, and in particular in determining the extent to which they make use of representation hierarchies. And Fodor is certainly right that most of the interesting questions about how human minds came to be organized in the ways that they are in fact organized are still wide open. But a simple analysis of the properties of representation hierarchies shows that global context sensitivity is not incompatible, after all, with massive modularity.

How can Fodor have missed this? He does briefly entertain a hierarchy-based escape from one version of his dilemma but he dismisses it with the argument that it requires that "each computational mechanism presupposes computational mechanisms less modular than itself" (p. 73), thereby producing a regress. But to the extent that one can say this of the *GODD* system above, one would also have to admit that the "less modular" modules (those higher in the hierarchy) are less specific (containing only *summaries* of the other modules, not their entire contents); they may therefore be as compact as the "more modular" modules and the feared intractability problems need not arise.

The failure to appreciate the computational significance of hierarchy may run deeper, to an underlying lack of familiarity with basic theoretical computer science. Several statements in the book reveal what is at best sloppy thinking about complexity and computability theory; even if it would be unfair to lay the blame for this at Fodor's feet (because, according to one reviewer, others in the field routinely make the same mistakes), any philosopher making arguments grounded in computational complexity theory would be well advised to study the theory and to use its terms carefully. For example, to anyone familiar with the proofs of computational equivalence of Turing machines and recurrent neural networks Fodor's discussion of connectionism will seem rather strange. How could connectionists be in even *worse* shape than proponents of symbolic theories *with respect to computational power* when their preferred mechanisms have the *same* computational power (Siegelmann & Sontag, 1991)? The answer has to do with explanatory (rather than computational) power, but the distinction is not clear in Fodor's text and most of his discussion focuses on apparent computational limits. Similarly, Fodor makes statements like "Turing machines can compute *anything* that's syntactic" (p. 30, emphasis in the original). This is simply false, as evidenced by the many volumes of algorithmically unsolvable problems (that is, problems that cannot be solved by Turing machines) that can be found on the shelves of any technical library. One might argue that some of these problems (for example, Turing's halting problem) involve semantic rather than syntactic properties, but it's hard to imagine any useful concept of "syntactic" that doesn't cover the uncomputable string-rewriting problems that can be found in many texts on computability theory (see e.g. Floyd & Beigel, 1994). An example is the problem of determining if a particular string can be generated by a particular set of unrestricted grammar rules; if this is not a syntactic property, then what is? Another example is the Post correspondence

problem: "Given two lists, each containing the same number of strings, is some non-trivial [non-empty] concatenation of strings from the first list equal to the concatenation of the *corresponding* strings from the second list?" (Floyd & Beigel, 1994, p. 508). Surely this is a syntactic property and yet one can prove that there is no Turing machine that can solve it in general. One is left with the impression that Fodor's grounding in theoretical computer science is not sufficient to mount an attack on the computational theory of mind based on alleged computational inefficiencies of particular architectures.

Where does this leave us with respect to psychological Darwinism? The first point to note is that Fodor's own arguments that psychological Darwinism follows from massive modularity now come back into play; if indeed the mind is massively modular (and this is no longer ruled out by context sensitivity), then natural selection may be the best explanation of the origins of human cognitive processes. But we can do better than this, again on the basis of purely computational considerations. Suppose we could show, through the use of computational simulations (if not through pure mathematical analysis), that standard Darwinian mechanisms lead naturally to the evolution of modular architectures? This would reinforce the new synthesis picture in a new way: if one accepts Darwinism more broadly, then one would expect, on the basis of such evidence, for minds to evolve with modular architectures.

Such evidence is already in hand. Darwinian simulations are widespread in computer science and "genetic and evolutionary computation" is now a large field to which several journals, annual conferences, and book series are devoted. In one of the larger areas of the field, called genetic programming, standard Darwinian mechanisms of selection and recombination are used to evolve computer programs for cognitive tasks ranging from artistic production to scientific discovery (and including many engineering problems). In 1994 John Koza published extensive evidence demonstrating that genetic programming, when provided with the raw materials for modularity, will automatically exploit modularity to produce better programs in less evolutionary time (Koza, 1994). For example, a program evolved to compute the parity (evenness or oddness) of some number of bits, when evolved in a context in which modules are an option, will often be composed of modules that themselves compute the parity of smaller numbers of bits; the modular solution is "better" in several respects (for example, it is more compact) and it emerges from the genetic programming system more quickly than do non-modular solutions. It can even be shown that modularity arises naturally in genetic programming systems that are neutral with respect to architecture—that is, in systems in which the mechanisms to support modularity must be built from more primitive computational elements and in which the "decision" to use modularity at all must emerge by natural selection (Spector, 2001; Spector & Robinson, 2002). The notion of modularity employed here is slightly different from Fodor's (it is the notion of a subroutine or macro in computer science), but encapsulation is among its defining features.

It is worth noting that the power of Darwinian evolution can itself be seen as

due to the computational properties of hierarchies. In arguing that psychological Darwinism follows from massive modularity Fodor writes:

> To get the feel of the thing, imagine cutting up the Manhattan telephone directory and then pairing all the numbers with all the names at random. How often do you suppose the number thus assigned to someone would be the number that he actually has? (pp. 93–94)

He argues that only some sort of "instructional" mechanism, sensitive to facts about the world, could produce the large numbers of innate true beliefs that humans apparently have [4]. But how plausible is it that natural selection could produce the required large numbers of innate true beliefs in a reasonable span of evolutionary time? To return to Fodor's example, how long would it take natural selection to unscramble the Manhattan telephone directory? Not as long as one might think.

Suppose that we have a population of randomly scrambled directories and that at least one of these directories has at least one correct name/number pair. If that directory is reproductively successful (which it ought to be, as it is better adapted to the environment—on rare occasions the call reaches the right person), then the "allele" for this correct pair will spread through the population. When this happens the amount of error in the average individual will already have decreased substantially by some measures; the number of possible permutations of a set increases dramatically (factorially) with the number of elements in the set, so each "repaired" pair will reduce the remaining number of permutations dramatically. Natural selection can do even better than this when aided by sexual recombination. Two parents each having one (different) correct pair will have a fair chance of producing a child with *both* correct pairs, thereby producing a substantial improvement in one generation, and the potential gains increase as the number of correct pairs in the parents increase. Of course children will also sometimes be *worse* than their parents, and the overall progress of the population will depend on several factors including selection pressure, mutation rates, and the "fitness landscape" of the underlying computational problem. Nonetheless, the potential for rapid evolution clearly exists.

The connection to hierarchical organization is that the primary mathematical apparatus that has been developed to analyze the dynamics of genetic algorithms, as described in the previous paragraph, involves a *hierarchy* of "schemata" describing sets of possible individuals. At the top of the hierarchy is a completely open-ended schema that describes any possible individual. Each step down in the hierarchy represents a commitment to a particular allele at a particular location. At the bottom of the hierarchy are the fully specified individuals, of which there are usually vastly too many to examine exhaustively. As first elucidated by Holland (1992; the original edition was published in 1975) and subsequently elaborated by many others (e.g. Poli, 2001), the power of the genetic algorithm (and more generally of natural selection combined with recombination) appears to derive from the way in which the properties of the schema hierarchy are leveraged to obtain maximal information about the search space while processing a minimal number of actual individuals.

The message of this digression on schema theory is not that evolution is particularly good at unscrambling phone directories; a custom-crafted algorithm

could probably do the job more quickly. The message is rather that (as Fodor acknowledges) Darwinian processes like those that we know to be operating in nature can also solve problems like this [5], that (as he may not fully appreciate) they can do it faster than one might guess, and that (to return to the theme of this article) their efficiency seems to be related to the way that an implicit genetic hierarchy is navigated by the mechanisms of recombination under the pressure of natural selection.

None of this will be surprising to students of computer science, all of whom learn to use hierarchies to simplify computing tasks ranging from number guessing games (first determine which half of the range contains the number, then determine which half of that half contains the number, etc.) to alphabetizing lists of names (for which hierarchical solutions have complexity of order $n \times \log(n)$ rather than the much worse n^2 of naive non-hierarchical approaches). But perhaps it will be surprising to others, and in any event it is important that the properties of hierarchies be better appreciated in discussions of the computational complexity of cognition.

In conclusion, hierarchical organization schemes are powerful tools that in some cases provide surprising computational efficiencies. Hierarchical organization solves the problem that Fodor believes refutes the massive modularity thesis. Because of this the computational theory of mind emerges from the attack unscathed, as do Darwinian explanations of the origins of minds. Indeed, recent work involving computational simulations provides new evidence that the modules required by the computational theory of mind can and do arise by strictly Darwinian mechanisms, thus bolstering the case for new synthesis cognitive science.

Acknowledgements

Jay Garfield, Joe Cruz, Ernie Alleva, Murray Kiteley, Harold Skulsky, Laura Sizer, Rebecca S. Neimark, and two anonymous reviewers provided comments that helped to improve this work. This effort was sponsored by the Defense Advanced Research Projects Agency (DARPA) and Air Force Research Laboratory, Air Force Materiel Command, USAF, under agreement number F30502–00–2–0611. This research was also made possible by generous funding from Hampshire College to the Institute for Computational Intelligence.

Notes

[1] I have cast the example in terms of bits rather than words for computational explicitness.

[2] Certainly the contribution made by modules with unused context bits is context-independent. For other modules the situation is not so clear.

[3] The context sensitivity really is "global" here; the *GODD* property depends on every bit in the entire system.

[4] Natural selection counts as an "instructional" mechanism in this context.

[5] Code that demonstrates this can be found at <http://hampshire.edu/lspector/unscramble.lisp>.

References

FLOYD, R.W. & BEIGEL, R. (1994). *The language of machines: an introduction to computability and formal languages*. New York: Computer Science Press, W.H. Freeman.

FODOR, J. (1983). *The modularity of mind*. Cambridge, MA: MIT Press.

FODOR, J. (2000). *The mind doesn't work that way*. Cambridge, MA: MIT Press.

HOLLAND, J.H. (1992). *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press.

KOZA, J.R. (1994). *Genetic programming II: automatic discovery of reusable programs*. Cambridge: MIT Press.

POLI, R. (2001). Exact schema theory for genetic programming and variable-length genetic algorithms with one-point crossover. *Genetic Programming and Evolvable Machines*, *2*, 123–163.

SIEGELMANN, H.T. & SONTAG, E.D. (1991). Turing computability with neural nets. *Applied Mathematics Letters*, *4*, 77–80.

SIMON, H.A. (1969). *The sciences of the artificial*. Cambridge, MA: MIT Press.

SPECTOR, L. (2001). Autoconstructive evolution: Push, PushGP, and Pushpop. In L. SPECTOR, E. GOODMAN, A. WU, W.B. LANGDON, H.-M. VOIGT, M. GEN, S. SEN, M. DORIGO, S. PEZESHK, M. GARZON & E. BURKE (Eds) *Proceedings of the Genetic and Evolutionary Computation Conference, GECCO-2001*. San Francisco, CA: Morgan Kaufmann.

SPECTOR, L. & ROBINSON, A. (2002). Genetic programming and autoconstructive evolution with the Push programming language. *Genetic Programming and Evolvable Machines*, *3*, 7–40.