COGNITIVE SCIENCE
A Multidisciplinary Journal

Cognitive Science 34 (2010) 966–971
Copyright © 2010 Cognitive Science Society, Inc. All rights reserved.
ISSN: 0364-0213 print / 1551-6709 online
DOI: 10.1111/j.1551-6709.2010.01120.x

# Computing Machinery and Understanding

Michael Ramscar

*Department of Psychology, Stanford University*

Received 11 March 2010; received in revised form 13 March 2010; accepted 22 March 2010

**Abstract**

How are natural symbol systems best understood? Traditional ‘‘symbolic’’ approaches seek to understand cognition by analogy to highly structured, prescriptive computer programs. Here, we describe some problems the traditional computational metaphor inevitably leads to, and a very different approach to computation (Ramscar, Yarlett, Dye, Denny, & Thorpe, 2010; Turing, 1950) that allows these problems to be avoided. The way we conceive of natural symbol systems depends to a large degree on the computational metaphors we use to understand them, and machine learning suggests an understanding of symbolic thought that is very different to traditional views (Hummel, 2010). The empirical question then is: Which metaphor is best?

*Keywords:* Computational metaphors; Natural language; Symbolic models; Prediction

> In considering the functions of the mind or the brain, we find certain operations which we can explain in purely mechanical terms. This we say does not correspond to the real mind: It is a sort of [onion] skin which we must strip off if we are to find the real mind. But then in what remains we find a further skin to be stripped off, and so on. Proceeding in this way do we ever come to the ‘‘real’’ mind, or do we eventually come to the skin which has nothing in it? In the latter case the whole mind is mechanical
>
> – Turing, 1950, pp. 454–455

The model of symbolic thought asserted by Hummel (2010) assumes that symbols are understood by reference to propositions (Fodor, 1981). This account of meaning is incompatible, in principle, with the constraints imposed by Feature-Label-Order on symbolic learning (Ramscar, Yarlett, Dye, Denny, & Thorpe, 2010). FLO constraints are a formal consequence of a kind of learning (which is basic to humans) and a kind of natural symbol (the kind basic to language), given elementary principles of computation and coding. What

Correspondence should be sent to Michael Ramscar, Department of Psychology, Stanford University, 450 Serra Mall, Stanford, CA 94305. E-mail: ramscar@stanford.edu

is worrying about Hummel's response to our formalization of these constraints is that it suggests that while we may be right about symbols, he may be right about how they are understood.

We are optimistic that this is not the case: The logic of FLO does not rule out meaning; it simply recasts understanding as a probabilistic, predictive process. Rather, we suspect Hummel's confusion is tied up with an expectation that the mind will conform to his particular model. In this, he is not alone. Belief in *the* ''computational metaphor of mind'' (singular) is widespread. We believe in a computational metaphor, too. But like Turing (1950), we think there is more than one way of conceiving of computation, and more than one computational metaphor.

In what follows, we briefly describe some problems with the picture of symbolic thought painted by Hummel. We then describe the approach to computation embodied in our own work, and the very different metaphor of mind and language it gives rise to. Our goal is not to advocate a model of computation for its own sake; it is to better understand the workings of natural symbol systems (i.e., natural language), the nature of which is still very much an open question.

Critically, we think that mapping natural language expressions onto the promiscuous relations Hummel describes is harder than he does. Far harder: We think you cannot do it. Consider the relation, *love* (Hummel, 2010): It does not limit itself to relating George to Mary, and Sally to Mary: One might reasonably say *George loves a good argument*, or even, *a good argument loves George*. However, ''*loves*'' means something else in these last two examples. It denotes different relationships. This poses a problem: Make *love* too promiscuous (Hummel & Holyoak, 1997), and it treats all *loves* relations identically. Hummel and Holyoak argue this is good, because it captures the ''similarities'' between *love* applied to arguments and people. However, it also ignores all of the differences. If a promiscuous *love* relation treats *Mary loves George* and *George loves arguments* identically, and if people's use of these two *loves* is not semantically identical—as Hummel and Holyoak admit—then two things follow: First, promiscuous relations fail to capture an essential feature of natural symbol systems, because people use the same words to describe different relations, while promiscuous relations do not. Second, as a consequence, promiscuous relational systems and meaningful natural symbol systems are *different things.*

Usually, one of two moves is made at this point. The first relegates the problems we pose about *love* to ''pragmatics'' (or *performance*) and avers that *competence*—whatever promiscuous relations *can* capture—is the thing to explain (Chomsky, 1965). However, if the meanings of symbols are a matter of pragmatics, this begs the question of what promiscuous relational systems actually *add* to our understanding of natural symbol systems.

Should we eschew the pragmatics option, we find ourselves with a bigger problem: Explaining how promiscuous relations can express different senses of *loves*. The usual way to do this is to assume that the different senses of *loves* refer to different *''concepts.''* The meaning of a *word* is represented by the concept WORD, which gets its meaning from OTHER WORDS, which in turn get their meanings from YET MORE WORDS, until it becomes WORDS ALL THE WAY DOWN (they are not words, of course, they are CONCEPTS—hence the capitals—and though no-one knows how they solve these

problems, people in this tradition seem happy enough crossing their fingers and hoping that someday, someone will; Hummel & Holyoak, 1997; Fodor, 1998; Chomsky, 2000; Doumas et al., 2006). Once this leap of faith is made, shades of meaning can be coded as different, less promiscuous relations: *LOVES-1* (Mary and George), LOVES-2 (arguments), LOVES-*n,* etc., and as long as one gets all this coded structure *exactly* right, you might get a model that *appears* sensitive to the different shades of meaning for *loves*.

The problem with this is that *everything* in the model's worldview has to be coded by hand, meaning that we can discover little about meaningful natural symbol systems from this ''turtles all the way down'' (see Hawking, 1988) approach to computation. This brings us to the heart of the matter: Many scientists working in the cognitive tradition believe that what we described so far is the limit of what computation—and modeling—can be: You take a computer and an algorithm, hand-code some representations ''just so,'' and—*voi-la!*—you have a cognitive simulation. (Just do not push the idea of ''cognitive'' or ''simu-late'' too hard.)

Given that theorizing about the mind is dominated by this computational metaphor, it is no surprise that cognitive science is in turn dominated by theories that assume the mind *must* come with many of its representations built in (while being vague to the point of opacity when it comes to specifying the computational content of these representations; Scholz & Pullum, 2006).

So much for the bad news. The good news is that hand-coded systems are not the be all and end all of computation and computational modeling. There is an alternative, first described by Turing (1950) in his remarkable paper, *Computing machinery and intelligence* (which also gave us the ''Turing Test''). Turing argued that there was little chance of computers simulating human intelligence if every conceivable eventuality had to be worked out in advance by a programmer (i.e., hand coded). A better strategy, Turing suggested, lay in *learning machines:* ''Instead of trying to produce a programme to simulate the adult mind, why not rather try to produce one which simulates the child's?'' (Turing, 1950, p. 456).

We can illustrate how learning machines work by considering the representations employed in the FLO simulations. One might reasonably ask, ''*weren't they hand coded too*?'' The answer is, ''not really.'' While we hand-structured the representations in the FLO simulations to make them accessible as illustrations, there was nothing in that structure that was not a straightforward reflection of the world. Because the world (the fribbles) and the labels covaried systematically, any algorithm that learned from the distribution of the fribble features and the distribution of error those features generated would have ended up with representations informationally equivalent to those in our simulations—even if, for example, the inputs to the FLO simulations had been vectors of pixel values from a digital camera. Why? Because there was structure in the relationships between the labels and fribbles, and because learning machines are able to *discover* that structure for themselves (Drucker, Burges, Kaufman, Smola, & Vapnik, 1997; Galistel, 2003; Rescorla & Wagner, 1972).

Learning machines present a paradox: Presented on a scale small enough to comprehend (our simulations), they appear unimpressive, and yet scaled up, they have revolutionized modern life. As metaphors, they can revolutionize our understanding of the mind and natural symbol systems.

We can illustrate this rather bold claim by comparing a learning machine (a search engine) to a classical symbolic program (a spreadsheet). Spreadsheets are rule governed: They provide a user with a set of functions that have been coded in by their programmers. Once a user has learned these functions, and the types of input symbols they are compatible with, spreadsheets are semantically *determinate*: Tell a spreadsheet to add 1, 2, and 5, and 8 will invariably appear. Inadvertently type ''q'' instead of ''1,'' and the program will fail (''#WTF?#'').

Spreadsheets typify the traditional symbolic approach to computation, and metaphorically, they naturally give rise to the views of mind, symbols, and natural language described by Hummel (2010).

Search engines engage users with a fundamentally different paradigm for computation. While a spreadsheet's power lies in the functions programmed into it, search engines offer few obvious ''functions.'' Rather, their power comes from their ability to *learn* from the masses of data distributed about the Web. Similarly, using search engines does not involve memorizing rules from manuals; rather, users learn from experience the kind of things that make good search terms. Search is semantically *probabilistic* for both the user and the machine, and it is driven by expectation and optimization, not hard and fast rules. When you enter a term into a search engine, no specific output is guaranteed. Type ''michl rmscr,'' for example, and you may still get a link to the author.

While spreadsheets place the burden of structuring information on the user (in return for *determinate* output), search engines do the opposite: They *discover* structure for the user—*probabilistically*. This fundamental contrast distinguishes between two different computational paradigms, and two different computational metaphors of mind. It also lies at the heart of our disagreement with Hummel: Because there is more than one computational metaphor of mind, it makes no sense to assume, as Hummel (2010) does, that whatever your metaphor has, the mind has too. We are faced with the task of discovering *which* paradigm provides the right metaphor for natural computation. This is an empirical matter. It cannot be resolved by fiat, or through vague analogies to first- and second-order logics.

Turing (1950)—the father of classical symbolic computation—thought that learning machines offered the best model for human intelligence. History supports this. Where computers have to interact with humans and natural symbol systems (rather than *vice versa*), learning machines have outperformed classic programming approaches. Consider the proceedings of the 1989 meeting of the Association for Computational Linguistics (Hirschberg, 1989): It is dominated by hand-coded, ''symbolic'' programs. Move forward to the present (Lee & im Walde, 2009), and almost every system presented employs some form of learning machine (in 1989, automatic speech recognition was in its infancy; 20 years later, one despairs of ever reaching a human on business calls).

While our understanding of machine learning has grown in recent years, its *theoretical* impact on our understanding of the nature of mind has been limited. Debate in cognitive science has focused on the representation of symbolic thought (is it ''symbolic'' or ''subsymbolic?''). Meanwhile, the *nature* of mental computation is largely taken for granted. This needs remedy. The different computational paradigms we describe present *alternative*, not complementary methods. (e.g., while it is good for a search engine to sug-

gest *ramscar* given ''rmscr,'' it is not so good for a spreadsheet to sum ''3,3,3,3,4'' to ''15'' because it expected ''3'' rather than ''4'' as the last digit).

Our analysis of FLO (Ramscar et al., 2010) arose out of the idea that the mind can be modeled as a kind of learning machine. This computational-level assumption (Marr, 1982) in turn led to formal, testable hypotheses and an account of symbolic thought *very* different to the one Hummel assumes (see Levy, 2008; Jaeger, 2010, for similar approaches). FLO does not constrain ''the symbol grounding problem'' (turtles all the way down) as Hummel claims, because this ''problem'' only exists under Hummel's analysis, which assumes a determinate relationship between symbols and their meanings, and which violates basic principles of coding and computation. Our account tries to respect these principles, which is why we favor a predictive, probabilistic model of communication.

Hummel's approach leads inevitably to an impasse, and there appears to be no way around it. Vague appeals to analogies between logics, learning, and symbols do not solve this problem, and it is not clear how mixing different computational paradigms is supposed to help: Obscuring the computational properties of models (Hummel & Holyoak, 1997, 2003; Doumas et al., 2008; see also Rogers & McClelland, 2004; Goodman, Tenenbaum, Feldman, & Griffiths, 2008) simply obscures the conceptual confusions they embody. Ultimately, the only real option is to peel away some more onion skin, because none of the above can substitute for conceptual analysis.

If we are to improve the quality of our conceptual analyses of language and cognition, it is essential that we first improve our understanding of the strengths and weakness and compatibilities and incompatibilities of different paradigms and methods for computation. Much like the imaginary spreadsheet, it is likely that many of the assumptions and intuitions that inform our current understanding of the mind will never quite add up. Computational metaphors may be our best tools for isolating cherished but flawed ideas about mental processes, and clarifying our intuitive confusions. The challenge is deciding which metaphor works best.

## Acknowledgments

This material is based on work supported by the National Science Foundation under grant nos. 0547775 and 0624345. I thank Melody Dye, Brad Love, Joe Dougherty, and Art Markman for comments.

## References

Chomsky, N. (1965). *Cartesian linguistics*. New York: Harper and Row.

Chomsky, N. (2000). *New horizons in the study of language and mind*. Cambridge, England: Cambridge University Press.

Doumas, L. A. A., Holyoak, K. J., & Hummel, J. E. (2006). The Problem with using associations to carry binding information. *Behavioral and Brain Sciences*, *29*, 38–39.

Doumas, L. A. A., Hummel, J. E., & Sandhofer, C. M. (2008). A theory of the discovery and predication of relational concepts. *Psychological Review*, *115*, 1–43.

Drucker, H., Burges, C. J. C., Kaufman, L., Smola, A., & Vapnik, V. (1997). *Support vector regression machines*. Advances in Neural Information Processing Systems 9, NIPS 1996 (pp. 155–161). Cambridge, MA: The MIT Press.

Fodor, J. A. (1981). *Representations: Philosophical essays on the foundations of cognitive science*. Cambridge, MA: The MIT Press.

Fodor, J. (1998). *Concepts: Where cognitive science went wrong*. New York: Oxford University Press.

Gallistel, C. R. (2003). Conditioning from an information processing perspective. *Behavioral Process*, *62*, 89–101.

Goodman, N. D., Tenenbaum, J. B., Feldman, J., & Griffiths, T. L. (2008). A rational analysis of rule-based concept learning. *Cognitive Science*, *32*(1), 108–154.

Hawking, S. (1988). *A brief history of time*. New York: Bantam Books.

Hirschberg, J. (Ed.) (1989) *Proceedings of the 27th annual meeting of the association for computational linguistics*, Vancouver, British Columbia, June 1989. Association for Computational Linguistics.

Hummel, J. E. (2010) Symbolic vs. associative learning. *Cognitive Science*, *34*(7), 1.

Hummel, J. E., & Holyoak, K. J. (1997). Distributed representations of structure: A theory of analogical access and mapping. *Psychological Review*, *104*, 427–466.

Hummel, J. E., & Holyoak, K. J. (2003). A symbolic-connectionist theory of relational inference and generalization. *Psychological Review*, *110*, 220–264.

Jaeger, T. F. (2010). Redundancy and reduction: Speakers manage syntactic information density. *Cognitive Psychology*, *61*, 23–62.

Lee, G. G., & im Walde, S. S. (2009) Proceedings of the 47th annual meeting of the association for computational linguistics, Singapore, August 2009. Association for Computational Linguistics.

Levy, R. (2008). Expectation-based syntactic comprehension. *Cognition*, *106*(3), 1126–1177.

Marr, D. C. (1982). *Vision: A computational investigation into the human representation and processing of visual information*. New York: Freeman.

Ramscar, M., Yarlett, D., Dye, M., Denny, K., & Thorpe, K. (2010) Feature-label-order effects and their implications for symbolic learning. *Cognitive Science*, *34*(7).

Rescorla, R. A., & Wagner, A. R. (1972). A theory of Pavlovian conditioning: Variations in the effectiveness of reinforcement and nonreinforcement. In A. H. Black & W. F. Prokasy (Eds.), *Classical Conditioning II: Current research and theory* (pp. 64–99). New York: Appleton-Century-Crofts.

Rogers, T. T., & McClelland, J. L. (2004). *Semantic cognition: A parallel distributed processing approach*. Cambridge, MA: MIT Press.

Scholz, B. C., & Pullum, G. K. (2006). Irrational nativist exuberance. In R. Stainton (Ed.), *Contemporary debates in cognitive science* (pp. 59–80). Oxford, England: Basil Blackwel.

Turing, A. M. (1950). Computing machinery and intelligence. *Mind*, *59*, 433–460.