## An Invitation to Cognitive Science
Daniel N. Osherson, Series Editor

Volume 1: Language
Edited by Lila R. Gleitman and Mark Liberman

Volume 2: Visual Cognition
Edited by Stephen M. Kosslyn and Daniel N. Osherson

Volume 3: Thinking
Edited by Edward E. Smith and Daniel N. Osherson

Volume 4: Conceptual Foundations
Edited by Saul Sternberg and Donald Scarborough

# Thinking

## An Invitation to Cognitive Science
Second Edition

Volume 3

edited by Edward E. Smith and Daniel N. Osherson

A Bradford Book
The MIT Press
Cambridge, Massachusetts
London, England

Tourangeau, R. (1984). Cognitive science and survey methods: A cognitive perspective. In T. Jabine, M. Straf, J. Tanur, and R. Tourangeau, eds., *Cognitive aspects of survey methodology: Building a bridge between disciplines*, 73–100. Washington, DC: National Academy Press.

Tourangeau, R., and K. A. Rasinski (1988). Cognitive processes underlying context effects in attitude measurement. *Psychological Bulletin* 103, 299–314.

Tulving, E., and Z. Pearlstone (1966). Availability versus accessibility of information in memory for words. *Journal of Verbal Learning and Verbal Behavior* 5, 381–391.

Tversky, A., and D. Griffin (1991). On the dynamics of hedonic experience: Endowment and contrast in judgments of well-being. In F. Strack, M. Argyle, and N. Schwarz, eds., *Subjective well-being*. Oxford: Pergamon.

Tversky, A., and D. Kahneman (1973). Availability: A heuristic for judging frequency and probability. *Cognitive Psychology* 5, 207–232.

Wänke, M., N. Schwarz, and H. Bless (in press). The availability heuristic revisited: Experienced ease of retrieval in mundane frequency estimates. *Acta Psychologica*.

Wyer, R. S., and T. K. Srull (1989). *Memory and cognition in its social context*. Hillsdale, NJ: Erlbaum.

Wyer, R. S., and T. K. Srull, eds. (1994). *Handbook of social cognition*. 2nd ed., 2 vols. Hillsdale, NJ: Erlbaum.

# Chapter 11
# The Mind as the Software of the Brain

*Ned Block*

Cognitive scientists often say that the mind is the software of the brain. This chapter is about what this claim means.

## 11.1 Machine Intelligence

In this section, we start with an influential attempt to define "intelligence," and then we consider how human intelligence is to be investigated on the machine model. In the last part of the section we discuss the relation between the mental and the biological.

### 11.1.1 The Turing Test

One approach to the mind has been to avoid its mysteries by simply *defining* the mental in terms of the behavioral. This approach has been popular among thinkers who fear that acknowledging mental states that do not reduce to behavior would make psychology unscientific because unreduced mental states are not intersubjectively accessible in the manner of the entities of the hard sciences. "Behaviorism," as the attempt to reduce the mental to the behavioral is called, has often been regarded as refuted, but it periodically reappears in new forms.

Behaviorists don't define the mental in terms of just plain *behavior*, for after all something can be intelligent even if it has never had the chance to exhibit its intelligence. Behaviorists define the mental not in terms of behavior, but rather behavioral *dispositions*, the tendency to emit certain behaviors given certain stimuli. It is important that the stimuli and the behavior be specified nonmentalistically. Thus, intelligence could not be defined in terms of the disposition to give sensible responses to questions, because that would be to define a mental notion in terms of another mental notion (indeed, a closely related one). To see the difficulty of behavioristic analyses, one has to appreciate how mentalistic our ordinary behavioral descriptions are. Consider, for example, *throwing*. A series of motions that constitute throwing if produced by one mental cause might be a dance to get the ants off if produced by another.

An especially influential behaviorist definition of intelligence was put forward by A. M. Turing (1950). Turing, one of the mathematicians who cracked the German code during World War II, formulated the idea of the universal Turing machine, which contains, in mathematical form, the essence of the programmable digital computer. Turing wanted to define intelligence in a way that applied to both men and machines, and indeed, to anything that is intelligent. His version of behaviorism formulates the issue of whether machines could think or be intelligent in terms of whether they could pass this test: A judge in one room communicates by teletype (this was 1950!) with a computer in a second room and a person in a third room for some specified period (let's say an hour). The computer is intelligent if and only if the judge cannot tell the difference between the computer and the person. Turing's definition finessed the difficult problem of specifying nonmentalistically the behavioral dispositions that are characteristic of intelligence by bringing in the discrimination behavior of a human judge. And the definition generalizes. *Anything* is intelligent if, and only if, it can pass the Turing test.

Turing suggested that we replace the concept of intelligence with the concept of passing the Turing test. But what is the replacement *for*? If the purpose of the replacement is practical, the Turing test is not a big success. If one wants to know if a machine does well at playing chess or diagnosing pneumonia or planning football strategy, it is better to see how the machine performs in action than to subject it to a Turing test. For one thing, what we care about is that it do well at detecting pneumonia, not that it do so in a way indistinguishable from the way a person would do it. And so if it does the job, who cares if it doesn't pass the Turing test?

A second purpose might be utility for theoretical purposes. But machines that can pass the Turing test, such as Weizenbaum's ELIZA (see below), have been dead ends in artificial intelligence research, not exciting beginnings. (See "Mimicry versus Exploration" in Marr 1977, and Shieber 1994.)

A third purpose, the one that comes closest to Turing's intentions, is the purpose of *conceptual clarification*. Turing was famous for having formulated a precise mathematical concept that he offered as a replacement for the vague idea of mechanical computability. The precise concept (computability by a Turing machine) did everything one would want a precise concept of mechanical computability to do. No doubt, Turing hoped that the Turing-test conception of intelligence would yield everything one would want from a definition of intelligence without the vagueness of the ordinary concept.

Construed as a proposal about how to make the concept of intelligence precise, there is a gap in Turing's proposal: we are not told how the judge is to be chosen. A judge who was a leading authority on genuinely intelligent machines might know how to tell them apart from people. For example, the expert may know that current intelligent machines get certain problems right that people get wrong. Turing acknowledged this point by jettisoning the claim that being able to pass the Turing test is a necessary condition of intelligence, weakening his claim to: passing the Turing test is a sufficient condition for intelligence. He says, "May not machines carry out something which ought to be described as thinking but which is very different from what a man does? This objection is a very strong one, but at least we can say that if, nevertheless, a machine can be constructed to play the imitation game satisfactorily, we need not be troubled by this objection" (p. 435). In other words, a machine that *does pass* is necessarily intelligent, even if some intelligent machines fail.

But the problem of how to specify the qualities of the judge goes deeper than Turing acknowledges, and ruins the Turing test as a sufficient condition too. A stupid judge, or one who has had no contact with technology, might think that a radio was intelligent. People who are naive about computers are amazingly easy to fool, as demonstrated in the First Turing Test at the Boston Computer Museum in 1991. (See Shieber 1994.) A version of Weizenbaum's ELIZA (described in the next paragraph) was classified as human by five of ten judges. The test was "restricted" in that the computer programmers were given specific topics that their questions would be restricted to, and the judges were forbidden to ask "tricky" questions. For example, if the topic were Washington, D.C., a judge was not supposed to ask questions like, "Is Washington, D.C. bigger than a breadbox?" However, the winning program's topic was "whimsical conversation," a "smart-aleck" way of interacting that all the judges fell in with immediately, and one that would have had the same effect, even without set topics. Further, the restrictions to nontricky questions weren't enforced. (I speak as one of the referees who failed to enforce them.) For the most part, the computer-naive judges didn't really know how to formulate a tricky question. Thus the variant of ELIZA may be regarded as having done well in an unrestricted Turing test. Variants on ELIZA (all written by Joseph Weintraub) also were victorious in the Second and Third Turing Tests. The upshot is that a completely stupid computer program has *already* been shown to pass a reasonable facsimile of a Turing test.

ELIZA is a simple program written by Joseph Weizenbaum that in one version imitates a psychiatrist. See Weizenbaum (1976). It employs a group of simple but effective strategies. For example, it looks for "key words" on a list supplied by the programmer, for example, "I," "you," "alike," "father," and "everybody." The words are ordered; for example, "father" comes before "everybody," and so if you type in "My father is afraid of everybody," the machine will spit back one of its "father" responses, such as "WHAT ELSE COMES TO MIND WHEN YOU THINK OF YOUR FATHER?" If you type in "Everybody laughed at me," you will get one of its responses to "everybody," such as "WHO IN PARTICULAR ARE YOU THINKING OF?" It also has techniques that simultaneously transform "you" into "I," and "me" into "you" so that if you type in "You don't agree with me," it can reply: "WHY DO YOU THINK THAT I DON'T AGREE WITH YOU?" It also *stores* sentences including certain key words such as "my." If your *current* input includes no key words, but if you earlier said, "My boyfriend made me come here," the program will say "DOES THAT HAVE ANYTHING TO DO WITH THE FACT THAT YOUR BOYFRIEND MADE YOU COME HERE?" It also has stock responses to "Parlez-vous francais?" (or any other sentence with "francais" in it), namely "I SPEAK ONLY ENGLISH." If all else fails, the machine has a list of last-ditch responses such as "WHO IS THE PSYCHIATRIST HERE, YOU OR ME?"

This type of program is easy to trip up if you know how it works. For example, in the First Turing Test, one judge said, "You're trying to make this difficult for me, aren't you?" and the program replied roughly as follows: "Would you prefer it if it was not true that you're trying to make this difficult for you aren't you?" As Shieber points out, the pronoun-transposition rules can't handle "tag questions" that end with, for example, "aren't you?"

The point that this program illustrates is that a simple program can be extraordinarily successful in activities akin to passing the Turing test. Weizenbaum's program is not sophisticated or complex by current standards (there is a PC version that is only 200 lines in BASIC) yet this type of program is better at passing the Turing test than anything else written to date, as shown by the three victories in a row in the Turing tests mentioned above. Imagine how convincing a program would be produced if the Defense budget were devoted to this task for a year! But even if a high-budget government initiative produced a program that was good at passing the Turing test, if the program was just a bundle of tricks like the Weizenbaum program, with question types all thought of in advance, and canned responses placed in the machine, the machine would not be intelligent.

One way of dealing with the problem of specifying the judge is to make some characterization about the judge's mental qualities part of the formulation of the Turing test. For example, one might specify that the judge be moderately knowledgeable about computers and good at thinking, or better, good at thinking about thinking. But including a specification of the judge's mental qualities in the description of the test will ruin the test as a way of *defining* the concept of intelligence in nonmentalistic terms. Further, if we are going to specify that the judge must be good at thinking about

thinking, we might just as well give up on having the judge judge which contestants are human beings or machines and just have the judge judge which contestants *think*. And then the idea of the Turing test would amount to: A machine thinks if our best thinkers (about thinking) think it thinks. Although this sounds like a platitude, it is actually false. For even our best thinkers are fallible. The most that can be claimed is that if our best thinkers think that something thinks, then it is rational for us to believe that it does.

I've made much of the claim that judges can be fooled by a mindless machine that is just a bag of tricks. "But," you may object, "how do we know that *we* are not just a bag of tricks?" Of course, in a sense perhaps we are, but that isn't the sense relevant to what is wrong with the Turing test. To see this point, consider the ultimate in unintelligent Turing test passers, a hypothetical machine that contains *all conversations of a given length* in which the machine's replies make sense. Let's stipulate that the test lasts one hour. Because there is an upper bound on how fast a human typist can type, and because there are a finite number of keys on a teletype, there is an upper bound on the "length" of a Turing test conversation. Thus there are a finite (though more than astronomical) number of different Turing test conversations, and there is no contradiction in the idea of *listing them all.*

Let's call a string of characters that can be typed in an hour or less a "typable" string. In principle, all typable strings could be generated, and a team of intelligent programmers could throw out all the strings that cannot be interpreted as a conversation in which at least one party (say the second contributor) is making sense. The remaining strings (call them the sensible strings) could be stored in a hypothetical computer (say, with marks separating the contributions of the separate parties), which works as follows. The judge types in something. Then the machine locates a string that starts with the judge's remark, spitting back its next element. The judge then types something else. The machine finds a string that begins with the judge's first contribution, followed by the machine's, followed by the judge's next contribution (the string will be there because all sensible strings are there), and then the machine spits back its fourth element, and so on. (We can eliminate the simplifying assumption that the judge speaks first by recording *pairs* of strings; this would also allow the judge and the machine to talk at the same time.) Of course, such a machine is only logically possible, not physically possible. The number of strings is too vast to exist, and even if they could exist, they could never be accessed by any sort of a machine in anything like real time. But because we are considering a proposed definition of intelligence that is supposed to capture the *concept* of intelligence, conceptual possibility will do the job. If the concept of intelligence is supposed to be exhausted by the ability to pass the Turing test, then even a universe in which the laws of physics are very different from ours should contain exactly as many unintelligent Turing-test passers as married bachelors, namely zero.

Notice that the choice of one hour as a limit for the Turing test is of no consequence, for the procedure just described works for *any* finite Turing test.

The following variant of the machine may be easier to grasp. The programmers start by writing down all typable strings, call them $A_1 \ldots A_n$. Then they think of *just one* sensible response to each of these, which we may call $B_1 \ldots B_n$. (Actually, there will be fewer *B*'s than *A*'s because some of the *A*'s will take up the entire hour.) The programmers may have an easier time of it if they think of themselves as simulating some definite personality, say my Aunt Bubbles, and some definite situation, say Aunt Bubbles being brought into the teletype room by her strange nephew and asked to answer questions for an hour. Thus each of the B's will be the sort of reply Aunt Bubbles would give to the preceding A. For example, if $A_{73}$ is "Explain general relativity," $B_{73}$ might be "Ask my nephew, he's the professor." What about the judge's replies to each of the B's? The judge can give any reply up to the remaining length limit, and so below each of the B's, there will sprout a vast number of C's (vast, but fewer than the number of B's, for the time remaining has decreased). The programmers' next task is to produce just one D for each of the C's. Thus if the B just mentioned is followed by a C that is "xyxyxyxyxyxyxy!" (Remember, the judge doesn't have to make sense), the programmers might make the following D "My nephew warned me that you might type some weird messages."

Think of conversations as paths downward through a tree, starting with an $A_i$ from the judge, a reply, $B_i$ from the machine, and so on. See figure 11.1. For each $A_i$–$B_i$–$C_j^i$ that is a beginning to a conversation, the programmers must produce a D that makes sense given the A, B, and C that precede it.

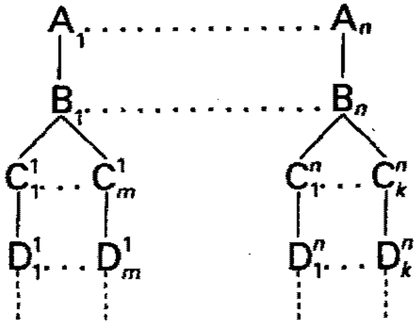


**Figure 11.1**
A conversation is any path from the top to the bottom.

The machine works as follows. The judge goes first. Whatever the judge types in (typos and all) is one of $A_1 \ldots A_n$. The machine locates the particular A, say $A_{2398}$, and then spits back $B_{2398}$, a reply chosen by the programmers to be appropriate to $A_{2398}$. The judge types another message, and the machine again finds it in the list of C's that sprout below $B_{2398}$, and then spits back the prerecorded reply (which takes into account what was said in $A_{2398}$ and $B_{2398}$). And so on. Though the machine can do as well in the one-hour Turing test as Aunt Bubbles, it *has the intelligence of a jukebox*. Every clever remark it produces was specifically thought of by the programmers as a response to the previous remark of the judge in the context of the previous conversation.

Though this machine is too big to exist, there is nothing incoherent or contradictory about its specification, and so it is enough to refute the behaviorist interpretation of the Turing test that I have been talking about.[1]

Notice that there is an upper bound on how long any particular Aunt Bubbles machine can go on in a Turing test, a limit set by the length of the strings it has been given. Of course *real people* have their upper limits too, given that real people will eventually quit or die. However, there is a very important difference between the Aunt Bubbles machine and a real person. We can define "competence" as idealized performance. Then, relative to appropriate idealizations, it may well be that real people have an infinite competence to go on. That is, if human beings were provided with unlimited memory and with motivational systems that gave passing the Turing test infinite weight, they could go on forever (at least according to conventional wisdom in cognitive science). This is definitely not the case for the Aunt Bubbles machine. But this difference provides no objection to the Aunt Bubbles machine as a refutation of the Turing-test conception of intelligence, because the notion of competence is not behavioristically acceptable, requiring as it does for its specification a distinction among components of the mind. For example, the mechanisms of thought must be distinguished from the mechanisms of memory and motivation.

"But," you may object, "isn't it rather chauvinist to assume that a machine must process information in just the way *we* do to be intelligent?"

[1] The Aunt Bubbles machine refutes something stronger than behaviorism, namely the claim that the mental "supervenes" on the behavioral; that is, that there can be no mental difference without a behavioral difference. (Of course, the behavioral dispositions are finite —see the next paragraph in the text.) I am indebted to Stephen White for pointing out to me that the doctrine of the supervenience of the mental on the behavioral is widespread among thinkers who reject behaviorism, such as Donald Davidson. The Aunt Bubbles machine is described and defended in detail in Block (1978, 1981a), and was independently discovered by White (1982). It has been dubbed the "Blockhead" in Jackson (1993).

Answer: Such an assumption would indeed be chauvinist, but I am not assuming it. The point against the Turing-test conception of intelligence is not that the Aunt Bubbles machine wouldn't process information in the way we do, but rather that the way in which it does process information is unintelligent despite its performance in the Turing test.

Ultimately, the problem with the Turing test for theoretical purposes is that it focuses on performance rather than on competence. Of course, performance is evidence for competence, but the core of our understanding of the mind lies with mental competence, not behavioral performance. The behaviorist cast of mind that leads to the Turing-test conception of intelligence also leads to labeling the sciences of the mind as "the behavioral sciences." But as Chomsky (1959) has pointed out, that is like calling physics the science of meter readings.

### 11.1.2 Two Kinds of Definitions of Intelligence

We have been talking about an attempt to define intelligence using the resources of the Turing test. However, there is a very different approach to defining intelligence.

To explain this approach, it will be useful to contrast two kinds of definitions of water. One might be better regarded as a definition of the word *'water'*. The word might be defined as the colorless, odorless, tasteless liquid that is found in lakes and oceans. In this sense of "definition," the definition of "water" is available to anyone who speaks the language, even someone who knows no science. But one might also define water by saying what water really is—that is, by saying what physicochemical structure in fact makes something pure water. The answer to this question involves its chemical constitution: $H_2O$. Defining a **word** is something we can do in our armchair, by consulting our linguistic intuitions about hypothetical cases, or, bypassing this process, by simply stipulating a meaning for a word. Defining (or explicating) the *thing* is an activity that involves empirical investigation into the nature of something in the world.

What we have been discussing so far is the first kind of definition of intelligence, the definition of the word, not the thing. Turing's definition is not the result of an empirical investigation into the components of intelligence of the sort that led to the definition of water as $H_2O$. Rather, he hoped to avoid muddy thinking about machine intelligence by stipulating that the word "intelligent" should be used in a certain way, at least with regard to machines. Quite a different way of proceeding is to investigate intelligence *itself* as physical chemists investigate water. We consider how this might be done in the next section, but first we should recognize a complication.

There are two kinds (at least) of kinds: *structural* kinds such as *water* or *tiger*, and *functional* kinds such as *mousetrap* or *gene*. A structural kind has a "hidden compositional essence"; for water, the compositional essence is a matter of its molecules consisting of two hydrogen molecules and one oxygen molecule. Functional kinds, by contrast, have no essence that is a matter of composition. A certain sort of function, a causal role, is the key to being a mousetrap or a carburetor. (The full story is quite complex: Something can be a mousetrap because it is made to be one even if it doesn't fulfill that function very well.) What makes a bit of DNA a gene is its function with respect to mechanisms that can read the information that it encodes and use this information to make a biological product.

Now the property of being intelligent is no doubt a functional kind, but it still makes sense to investigate it experimentally, just as it makes sense to investigate genes experimentally. One topic of investigation is the role of intelligence in problem solving, planning, decision making, and so on. Just what functions are involved in a functional kind is often a difficult and important empirical question. The project of Mendelian genetics has been to investigate the function of genes at a level of description that does not involve their molecular realizations. A second topic of investigation is the nature of the realizations that have the function in us, in humans: DNA in the case of genes. Of course, if there are Martians, their genes may not be composed of DNA. Similarly, we can investigate the functional details and physical basis of human intelligence without attention to the fact that our results will not apply to other mechanisms of other hypothetical intelligences.

### 11.1.3 Functional Analysis

Both types of projects just mentioned can be pursued via a common methodology, which is sometimes known as *functional analysis.* Think of the human mind as represented by an intelligent being in the head, a "homunculus." Think of this homunculus as composed of smaller and stupider homunculi, and each of these being composed of still smaller and still stupider homunculi until you reach a level of completely mechanical homunculi. (This picture was first articulated in Fodor 1968; see also Dennett 1974 and Cummins 1975.)

Suppose one wants to explain how we understand language. Part of the system will recognize individual words. This word-recognizer might be composed of three components, one of which has the task of fetching each incoming word, one at a time, and passing it to a second component. The second component includes a dictionary, that is, a list of all the words in the vocabulary, together with syntactic and semantic information about each word. This second component compares the target word with words in the vocabulary (perhaps executing many such comparisons simultaneously) until it gets a match. When it finds a match, it sends a signal to a third component whose job it is to retrieve the syntactic and semantic

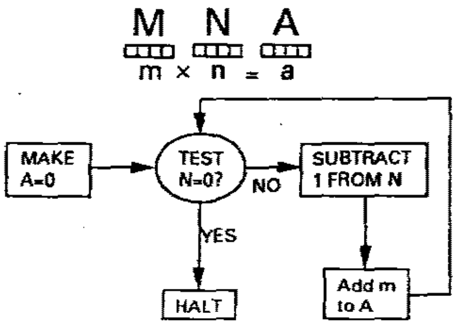


**Figure 11.2**
Program for multiplying. The user begins the multiplication by putting representations of *m* and *n*, the numbers to be multiplied, in registers M and N. At the end of the computation, the answer will be found in register A. See the text for a description of how the program works.

information stored in the dictionary. This speculation about how a model of language understanding works is supposed to illustrate how a cognitive competence can be explained by appeal to simpler cognitive competences, in this case the simple mechanical operations of fetching and matching.

The idea of this kind of explanation of intelligence comes from attention to the way in which computers work. Consider a computer that multiplies *m* times *n* by adding *m* to zero *n* times. Here is a program for doing this computation. Think of *m* and *n* as represented in the registers M and N in figure 11.2. Register A is reserved for the answer, *a*. First, a representation of 0 is placed in the register A. Second, register N is examined to see if it contains (a representation of) 0. If the answer is yes, the program halts and the correct answer is 0. (If $n = 0$, *m* times $n = 0$.) If no, N is decremented by 1 (and so register N now contains a representation of $n - 1$), and (a representation of) *m* is added to the answer register, A. Then the procedure loops back to the second step: register N is checked once again to see if its value is 0; if not, it is again decremented by 1, and again *m* is added to the answer register. This procedure continues until N finally has the value 0, at which time *m* will have been added to the answer register exactly *n* times. At this point, the answer register contains a representation of the answer.

This program multiplies via a "decomposition" of multiplication into other processes, namely addition, subtraction of 1, setting a register to 0, and checking a register for 0. Depending on how these things are themselves done, they may be further decomposable, or they may be the fundamental bottom-level processes, known as *primitive processes*.

The cognitive-science definition or explication of intelligence is analogous to this explication of multiplication. Intelligent capacities are understood via decomposition into a network of less intelligent capacities, ultimately grounded in totally mechanical capacities executed by primitive processors.

The concept of a primitive process is very important; the next section is devoted to it.

### 11.1.4 Primitive Processors

What makes a processor primitive? One answer is that for primitive processors, the question, "How does the processor work?" *is not a question for cognitive science to answer*. The cognitive scientist answers "How does the multiplier work?" for the multiplier described above by giving the program or the information-flow diagram for the multiplier. But if components of the multiplier, say the gates of which the adder is composed, are primitive, then it is not the cognitive scientist's business to answer the question of how such a gate works. The cognitive scientist can say, "That question belongs in another discipline, electronic circuit theory." Distinguish the question of *how something works* from the question of *what it does*. The question of *what* a primitive processor does is part of cognitive science, but the question of *how* it does it is not.

This idea can be made a bit clearer by looking at how a primitive processor actually works. The example involves a common type of computer adder, simplified so as to add only single digits.

To understand this example, you need to know these simple facts about binary notation:[2] 0 and 1 are represented alike in binary and normal (decimal) notation, but the binary representation that corresponds to decimal '2' is '*10*'. Our adder will solve these four problems:

$$0 + 0 = 0$$

$$1 + 0 = 1$$

$$0 + 1 = 1$$

$$1 + 1 = 10$$

The first three equations are true in both binary and decimal, but the last is true only if understood in binary.

The second item of background information is the notion of a gate. An AND gate is a device that accepts two inputs, and emits a single output. If both inputs are '*1*'s, the output is a '*1*'; otherwise, the output is a '*0*'. An EXCLUSIVE-OR (either but not both) gate is a "difference detector": it

[2] The rightmost digit in binary (as in familiar decimal) is the 1s place. The second digit from the right is the 2s place (corresponding to the 10s place in decimal). Next is the 4s place (that is, 2 squared), just as the corresponding place in decimal is the 10 squared place.

emits a '*0*' if its inputs are the same (that is, '*1*'/'*1*' or '*0*'/'*0*'), and it emits a '*1*' if its inputs are different (that is, '*1*'/'*0*' or '*0*'/'*1*').

This talk of '*1*' and '*0*' is a way of thinking about the "bistable" states of computer representers. These representers are made so that they are always in one or the other of two states, and only momentarily in between. (This is what it is to be bistable.) The states might be a 4-volt and a 7-volt potential. If the two input states of a gate are the same (say 4 volts), and the output is the same as well (4 volts), and if every other combination of inputs yields the 7-volt output, then the gate is an AND gate, and the 4-volt state realizes '*1*'. (Alternatively, if the 4-volt state is taken to realize '*0*', the gate is an "inclusive or" (either or both) gate.) A different type of AND gate might be made so that the 7-volt state realized '*1*'. The point is that '*1*' is conventionally assigned to *whatever* bistable physical state of an AND gate it is that has the role mentioned, that is, '*1*' is conventionally assigned to whatever state it is such that two of them as inputs yield another one as output, and nothing else yields that output. And all that counts about an AND gate from a computational point of view is its input-output function, not how it works or whether 4 volts or 7 volts realizes a '*1*'. Notice the terminology I have been using: one speaks of a physical state (4-volt potential) as "realizing" a computational state (having the value '*1*'). This distinction between the computational and physical levels of description will be important in what follows, especially in section 11.3.

Here is how the adder works. The two digits to be added are connected both to an AND gate and to an EXCLUSIVE-OR gate, as illustrated in figures 11.3a and 11.3b. Let's look at 11.3a first. The digits to be added are '*1*' and '*0*', and they are placed in the input register, which is the top pair of boxes. The EXCLUSIVE-OR gate, which, you recall, is a difference detector, sees different things, and so outputs a '*1*' to the rightmost box of the answer register, which is the bottom pair of boxes. The AND gate outputs a '*0*' except when it sees two '*1*'s, and so it outputs a '*0*'. In this way, the circuit computes '*1 + 0 = 1*'. For this problem, as for '*0 + 1 = 1*' and '*0 + 0 = 0*', the EXCLUSIVE-OR gate does all the real work. The role of the AND gate in this circuit is *carrying*, and that is illustrated in figure 11.3b. The digits to be added, '*1*' and '*1*', are placed in the top register again. Now, both inputs to the AND gate are '*1*'s, and so the AND gate outputs a '*1*' to the leftmost box of the answer (bottom) register. The EXCLUSIVE-OR gate puts a '*0*' in the rightmost box, and so we have the correct answer, '*10*'.

The borders between scientific disciplines are notoriously fuzzy. No one can say *exactly* where chemistry stops and physics begins. Because the line between the upper levels of processors and the level of primitive processors is the same as the line between cognitive science and one of the

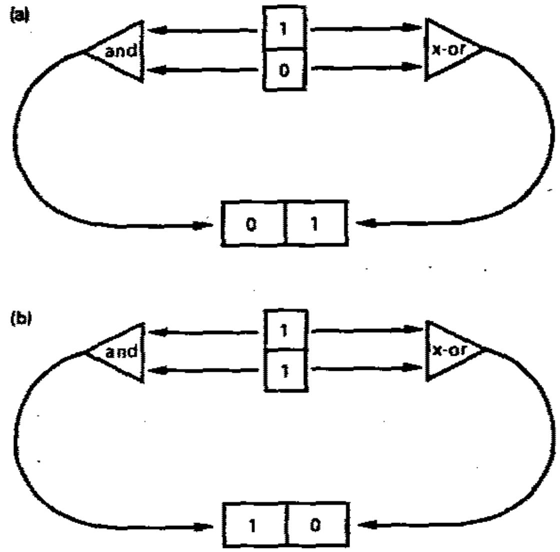


**Figure 11.3**
(a) Adder doing 1 + 0 = 1 (b) Adder doing 1 + 1 = 10.

"realization" sciences such as electronics or physiology, the boundary between the levels of complex processors and the level of primitive processors will have the same fuzziness. Nonetheless, in this example we should expect that the gates are the primitive processors. If they are made in the usual way, they are the largest components whose operation must be explained, not in terms of cognitive science, but rather in terms of electronics or mechanics or some other realization science. Why the qualification, "If they are made in the usual way"? It would be *possible* to make an adder each of whose gates were *whole computers,* with their own multipliers, adders, and normal gates. (We could even make an adder whose gates were people!) It would be silly to waste a whole computer (or a person) on such a simple task as that of an AND gate, but it could be done. In that case, the real level of primitives would not be the gates of the original adder, but rather the (normal) gates of the component computers.

Primitive processors are the only computational devices for which *behaviorism is true.* Two primitive processors (such as gates) count as computationally equivalent if they have the same input-output function, that is, the same actual and potential behavior, even if one works hydraulically and the other electrically. But computational equivalence of *non*primitive devices is not to be understood in this way. Consider two multipliers that work via different programs. Both accept inputs and emit outputs only in decimal notation. One of them converts inputs to binary, does the computation in binary, and then converts back to decimal. The other does the computation directly in decimal. These are not computationally equivalent multipliers despite their identical input-output functions.

If the mind is the software of the brain, then we must take seriously the idea that the functional analysis of human intelligence will bottom out in primitive processors in the brain.

### 11.1.5 The Mental and the Biological

One type of electrical AND gate consists of two circuits with switches arranged as in figure 11.4. The switches on the left are the inputs. When only one or neither of the input switches is closed, nothing happens, because the circuit on the left is not completed. Only when both switches are closed does the electromagnet go on, and that pulls the switch on the right closed, thereby turning on the circuit on the right. (The circuit on the right is only partially illustrated.) In this example, a switch being closed realizes '1'; it is the bistable state that obtains as an output if and only if two of them are present as an input.

Another AND gate is illustrated in figure 11.5. If neither of the mice on the left is released into the right-hand part of their cages, or if only one of the mice is released, the cat does not strain hard enough to pull the leash. But when both are released, and are therefore visible to the cat, the cat strains enough to lift the third mouse's gate, letting it into the cheesy part of its box. And so we have a situation in which a mouse getting cheese is output if and only if two cases of mice getting cheese are input.

The point illustrated here is the irrelevance of hardware realization to computational description. These gates work in very different ways, but they are nonetheless computationally equivalent. And of course it is possible to think of an indefinite variety of other ways of making a primitive AND gate. How such gates work is no more part of the domain of cognitive science than is the nature of the buildings that hold computer factories. This question reveals a sense in which the computer model of the mind is profoundly *unbiological*. We are beings who *have* a useful and interesting biological level of description, but the computer model of the mind aims for a level of description of the mind that abstracts away from the biological realizations of cognitive structures. As far as the computer model goes, it does not matter whether our gates are realized in gray matter, switches, or cats and mice.

Of course, this is not to say that the computer model is in any way incompatible with a biological approach. Indeed, cooperation between the biological and computational approaches is vital to *discovering* the program of the brain. Suppose one were presented with a computer of alien design and set the problem of ascertaining its program by any means possible. Only a fool would choose to ignore information to be gained by opening the computer up to see how its circuits work. One would want to put information at the program level together with information at the electronic level, and likewise, in finding the program of the human mind, one can expect biological and cognitive approaches to complement each other.

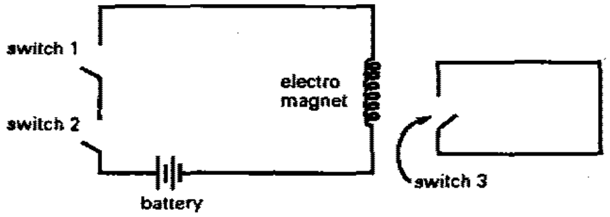


**Figure 11.4**
Electrical AND gate.
*Open = 0, closed = 1*

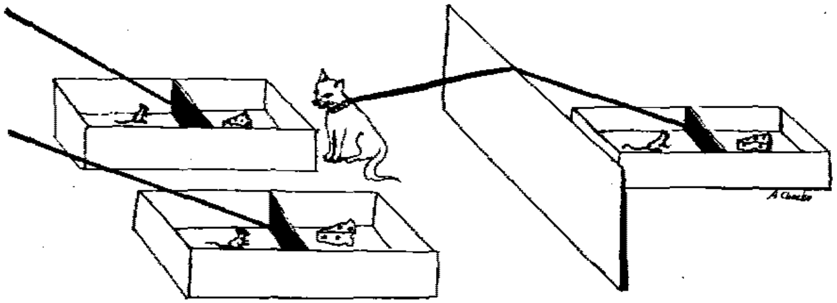


**Figure 11.5**
Cat and mouse AND gate.
*Hungry mouse = 0, mouse fed = 1*

Nonetheless, the computer model of the mind has a built-in antibiological bias, in the following sense. If the computer model is right, we should be able to create intelligent machines in our image—our *computational* image, that is. And the machines we create in our computational image may not be biologically similar to us. If we can create machines in our computational image, we will naturally feel that the most compelling theory of the mind is one that is general enough to apply to both them and us, and this will be a computational theory, not a biological theory. A biological theory of the *human* mind will not apply to these machines, though the biological theory will have a complementary advantage: namely, that such a biological theory will encompass us together with our less intelligent biological cousins, and thus provide a different kind of insight into the nature of human intelligence. Both approaches can accommodate evolutionary considerations, though for the computational paradigm, evolution is no more relevant to the nature of the mind than the programmer's intentions are to the nature of a computer program.

Some advocates of connectionist computer models of the mind claim that they are advocating a biological approach. But examining typical connectionist models shows that they are firmly within the computationalist paradigm. For example, the popular backpropagation models allow weights of network connections to shift between positive and negative, something that cannot happen with neural connections. It would be a simple matter to constrain the network models so that connections could not change between excitatory and inhibitory, but no one wants to do this because then the models would not work. Thus, although these models are biologically inspired, biological fidelity is rejected when it does not make computational sense.

## 11.2 Intelligence and Intentionality

Our discussion so far has centered on the computational approach to one aspect of the mind, intelligence. But the mind has a different aspect that we have not yet discussed, one that has a very different relation to computational ideas, namely *intentionality*.

For our purposes, we can take intelligence to be a capacity for various intelligent activities such as solving mathematics problems, deciding whether to go to graduate school, and figuring out how spaghetti is made. (Notice that this analysis of intelligence as a capacity to solve, figure out, decide, and the like, is a mentalistic, not a behaviorist analysis.)

Intentionality is aboutness. Intentional states represent the world as being a certain way. The thought that the moon is full and the perceptual state of seeing that the moon is full are both about the moon and they both represent the moon as being full. Thus both are intentional states. (See volume 2, chapter 9.) We say that the *intentional content* of both the thought and the perceptual state is *that the moon is full*. A single intentional content can have very different behavioral effects, depending on its relation to the person who has the content. For example, the fear that there will be nuclear war might inspire one to work for disarmament, but the belief that there will be nuclear war might influence one to emigrate to Australia. (Don't let the spelling mislead you: intending is only one kind of intentional state. Believing and desiring are others.) Intentionality is an important feature of many mental states, but many philosophers believe it is not "the mark of the mental." There are bodily sensations, the experience of orgasm, for example, that are genuine mental states but have no intentional content. (Well, maybe there is a bit of intentional content to this experience, for example, locational content, but the phenomenal content of the experience, what it is like to have it, is clearly not exhausted by that intentional content.)

The features of thought just mentioned are closely related to features of language. Thoughts represent, are about things, and can be true or false; and the same is true of *sentences*. The sentence "Bruce Springsteen was born in the USSR" is about Springsteen, represents him as having been born in the Soviet Union, and is false. It would be surprising if the intentional content of thought and of language were independent phenomena, and so it is natural to try to reduce one to the other or to find some common explanation for both. We pursue this idea below, but before we go any further, let's try to get clearer about just what the difference is between intelligence and intentionality.

One way to get a handle on the distinction between intelligence and intentionality is to recognize that in the opinion of many writers on this topic you can have intentionality without intelligence. Thus John McCarthy (creator of the artificial-intelligence language LISP) holds that thermostats have intentional states in virtue of their capacity to represent and control temperature (McCarthy 1980). And there is a school of thought that assigns content to tree rings in virtue of their representing the age of the tree (see the references below in section 11.3). But no school of thought holds that the tree rings are actually intelligent. An intelligent system must have certain intelligent capacities, capacities to *do* certain sorts of things, and tree rings can't do these things. Less controversially, words on a page and images on a television screen have intentionality. For example, my remark earlier in this paragraph to the effect that McCarthy created LISP is about McCarthy. But words on a page have no intelligence. Of course, the intentionality of words on a page is only derived intentionality, not original intentionality. (See Searle 1980 and Haugeland 1980.) Derived intentional content is inherited from the original intentional contents of intentional systems such as you and me. We have a great deal of freedom in giving symbols their derived intentional content. If we want to, we can decide that "McCarthy" will now represent Minsky or Chomsky. Original intentional contents are the intentional contents that the representations of an intentional system have *for* that system. Such intentional contents are not subject to our whim. Words on a page have derived intentionality, but they do not have any kind of intelligence, not even derived intelligence, whatever that would be.

Conversely, there can be intelligence without intentionality. Imagine that an event with negligible (but, and this is important, nonzero) probability occurs: In their random movement, particles from the swamp come together and by chance result in a molecule-for-molecule duplicate of your brain. The swamp brain is arguably intelligent, because it has many of the same capacities that your brain has. If we were to hook it up to the right inputs and outputs and give it an arithmetic problem, we would get an intelligent response. But there are reasons for denying that it has the intentional states that you have, and indeed, for denying that it has any intentional states at all. For because we have not hooked it up to input devices, it has never had any information from the world. Suppose the swamp brain and your brain go through an identical process, which in your case is the thinking of the thought that Bernini vandalized the Pantheon. The identical process in the swamp-brain has the phenomenal features of that thought, in the sense of "phenomenal content" indicated in the discussion of orgasm above. What it is like for you to think the thought is just what it is like for the swamp-brain. But, unlike you, the swamp-brain has no idea who Bernini was, what the Pantheon is, or what vandalizing is. No information about Bernini has made any kind of contact with the swamp-brain; no signals from the Pantheon have reached it, either. Had it a mouth, it would merely be mouthing words. Thus no one should be happy with the idea that the swamp-brain is thinking the thought that Bernini vandalized the Pantheon.

The upshot: what makes a system intelligent is what it can do, what it has the capacity to do. Thus intelligence is future oriented. What makes a system an intentional system, by contrast, is in part a matter of its causal history; it must have a history that makes its states represent the world, that is, have *aboutness*. Intentionality has a past-oriented requirement. A system can satisfy the future-oriented needs of intelligence while flunking the past-oriented requirement of intentionality. (Philosophers disagree about just *how* future-oriented intentionality is, whether thinking about something requires the ability to "track" it; but there should be little disagreement that there is *some* past-oriented component.)

Now let's see what the difference between intelligence and intentionality has to do with the computer model of the mind. Notice that the method of functional analysis that explains intelligent processes by reducing them to unintelligent mechanical processes *does not explain intentionality*. The parts of an intentional system can be just as intentional as the whole system. (See Fodor 1981.) In particular, the component processors of an intentional system can manipulate symbols that are about just the same things that the symbols manipulated by the whole system are about. Recall that the multiplier of figure 11.2 was explained via decomposition into devices that add, subtract, and the like. The multiplier's states were intentional in that they were about numbers. The states of the adder, subtractor, and so on, are also about numbers and are thus similarly intentional.

There is, however, an important relation between intentionality and functional decomposition that is explained in the next section. As you will see, though the multiplier's and the adder's states are about numbers, the gate's representational states represent *numerals*, and in general the subject matter of representations shifts as we cross the divide from complex processors to primitive processors.

### 11.2.1 The Brain as a Syntactic Engine Driving a Semantic Engine

To see the idea of the brain as a syntactic engine it is important to see the difference between the number 1 and the symbol (in this case, a numeral or digit) '*1*'. (The convention in this book is that italics indicate symbols, but in this chapter, numerals are indicated by italics plus single quotation marks.) Certainly, the difference between the city, Boston, and the word '*Boston*' is clear enough. The former has bad drivers in it; the latter has no people or cars at all, but does have six letters. No one would confuse a city with a word, but it is less obvious what the difference is between the number 1 and the numeral '*1*'. The point to keep in mind is that many different symbols, such as '*II*' (in Roman numerals), and '*two*' (in alphabetical writing) denote the same number, and one symbol, such as '*10*', can denote different numbers in different counting systems (as '*10*' denotes one number in binary and another in decimal).

With this distinction in mind, one can see an important difference between the multiplier and the adder discussed earlier. The algorithm used by the multiplier in figure 11.2 is notation *in*dependent: *Multiply n by m by adding n to zero m times* works in any notation. And the program described for implementing this algorithm is also notation independent. As we saw in the description of this program in section 11.1.3, the program depends on the properties of the numbers represented, not the representations themselves. By contrast, the internal operation of the adder described in figures 11.3A and 11.3B depends on binary notation, and its description in section 11.1.4 speaks of numerals (notice the quotation marks and italics) rather than numbers. Recall that the adder exploits the fact that an EXCLUSIVE-OR gate detects symbol differences, yielding a '*1*' when its inputs are different digits, and a '*0*' when its inputs are the same digits. This gate gives the right answer all by itself so long as no carrying is involved. The trick used by the EXCLUSIVE-OR gate depends on the fact that when you add two digits of the same type ('*1*' and '*1*' or '*0*' and '*0*') the rightmost digit of the answer is the same. This result is true in binary, but not in other standard notations. For example, it is not true in familiar decimal notation. (1 + 1 = 2, but 0 + 0 = 0.)

The inputs and outputs of both the multiplier and the adder must be seen as referring to numbers. One way to see this fact is to notice that otherwise one could not see the multiplier as exploiting an algorithm involving multiplying numbers by adding numbers. What are multiplied and added are numbers. But once we go *inside* the adder, we must see the binary states as referring to *symbols themselves.* For as just pointed out, the algorithms are notation dependent. This change of subject matter is even more dramatic in some computational devices, in which there is a level of processing in which the algorithms operate over *parts* of decimal numerals. Consider, for example, a calculator, in which the difference between an '*8*' and a '*3*' is a matter of two small segments on the left of the '*8*' being turned off to make a '*3*'. In calculators, there is a level at which the algorithms concern these segments.

This fact gives us an interesting additional characterization of primitive processors. Typically, as we functionally decompose a computational system, we reach a point where there is a shift of subject matter from abstractions like numbers or from things in the world to the symbols themselves. The inputs and outputs of the adder and multiplier refer to numbers, but the inputs and outputs of the gates refer to numerals. Typically, this shift occurs when we have reached the level of primitive processors. The operation of the higher-level components such as the multiplier can be explained in terms of a program or algorithm that is manipulating numbers. But the operation of the gates cannot be explained in terms of number manipulation; it must be explained in symbolic terms (or at lower levels, for example in terms of electromagnets). At the most basic computational level, computers are symbol crunchers, and for this reason the computer model of the mind is often described as the symbol-manipulation view of the mind.

Seeing the adder as a syntactic engine driving a semantic engine requires noticing two functions: one maps numbers onto other numbers, and the other maps symbols onto other symbols. The symbol function is concerned with the numerals as symbols—without attention to their meanings. Here is the symbol function:

'*0*', '*0*' → '*0*'

'*0*', '*1*' → '*1*'

'*1*', '*0*' → '*1*'

'*1*', '*1*' → '*10*'

The idea is that we interpret something physical in a machine or its outputs as symbols, and some other physical aspect of the machine as indicating that the symbols are inputs or outputs. Then given that interpretation, the machine's having some symbols as inputs causes it to have other symbols as outputs. For example, having the pair '*0*', '*0*' as inputs causes having '*0*' as an output. And so the symbol function is a matter of the causal structure of the machine under an interpretation.

This symbol function is mirrored by a function that maps the numbers represented by the numerals on the left onto the numbers represented by the numerals on the right. This function will thus map numbers onto numbers. We can speak of this function that maps numbers onto numbers as the *semantic* function (semantics being the study of meaning), because it is concerned with the meanings of the symbols, not the symbols themselves. (It is important not to confuse the notion of a semantic function in this sense with a function that maps symbols onto what they refer to; the semantic function maps numbers onto numbers, but the function just mentioned, which often goes by the same name, would map symbols onto numbers.) Here is the semantic function (in decimal notation—you must choose *some* notation to express a semantic function):

0, 0 → 0

0, 1 → 1

1, 0 → 1

1, 1 → 2

Notice that the two specifications just given differ in that the first maps quoted entities onto other quoted entities. The second has no quotes. The first function maps symbols onto symbols; the second function maps the numbers referred to by the arguments of the first function onto the numbers referred to by the values of the first function. (A function maps arguments onto values.) The first function is a kind of linguistic "reflection" of the second.

The key idea behind the adder is that of an isomorphism between these two functions. The designer has found a machine that has physical aspects that can be interpreted symbolically, and under that symbolic interpretation, there are symbolic regularities: some symbols in inputs result in other symbols in outputs. These symbolic regularities are isomorphic to rational relations among the semantic values of the symbols of a sort that are useful to us, in this case the relation of addition. It is the *isomorphism between these two functions* that explains how it is that a device that manipulates symbols manages to add numbers.

Now the idea of the brain as a syntactic engine driving a semantic engine is just a generalization of this picture to a wider class of symbolic activities, namely the symbolic activities of human thought. The idea is that we have symbolic structures in our brain, and that nature (evolution and learning) has seen to it that there are correlations between causal interactions among these structures and rational relations among the meanings of the symbolic structures. A crude example: the way in which we avoid swimming in shark-infested water is that the brain-symbol structure *'shark'* engenders the brain-symbol structure *'danger'*. (What makes *'danger'* mean *danger* is discussed below.)

The primitive mechanical processors "know" only the "syntactic" forms of the symbols they process (for example, what strings of zeroes and ones they see), and not what the symbols mean. Nonetheless, these meaning-blind primitive processors control processes that "make sense"—processes of decision, problem solving, and the like. In short, there is a correlation between the meanings of our internal representations and their forms. And this explains how it is that our syntactic engine can drive our semantic engine.[3]

In the last paragraph I mentioned a correlation between causal interactions among symbolic structures in our brain and rational relations among the meanings of the symbol structures. This way of speaking can be misleading if it encourages the picture of the neuroscientist opening the brain, just *seeing* the symbols, and then figuring out what they mean. Such a picture inverts the order of discovery, and gives the wrong impression of what makes something a symbol.

The way to discover symbols in the brain is first to map out rational relations among states of mind, and then identify aspects of these states that can be thought of as symbolic in virtue of their functions. Function is what gives a symbol its identity, even the symbols in English orthography, though this relation can be hard to appreciate because these functions have been rigidified by habit and convention. In reading unfamiliar handwriting we may notice an unorthodox symbol, someone's weird way of writing a letter of the alphabet. How do we know which letter of the alphabet it is? By its function! Th% function of a symbol is som%thing on% can appr%ciat% by s%%ing how it app%ars in s%nt%nc%s containing familiar words whos% m%anings w% can gu%ss. You will have little trouble figuring out, on this basis, what letter in the last sentence was replaced by '%'.

### 11.2.2 Is a Wall a Computer?

John Searle (1990a) argues against the computationalist thesis that the brain is a computer. He does not say that the thesis is false, but rather that it is trivial, because, he suggests, everything is a computer; indeed, everything is *every* computer. In particular, his wall is a computer computing Wordstar. (See also Putnam 1988, for a different argument for a similar conclusion.) The points in the preceding section allow easy understanding of the motivation for this claim and what is wrong with it. In that section we saw that the key to computation is an isomorphism. We arrange things so that, if certain physical states of a machine are understood as symbols, then causal relations among those symbol-states mirror useful rational relations among the meanings of those symbols. The mirroring is an isomorphism. Searle's claim is that this sort of isomorphism is cheap. We can regard two aspects of the wall at time *t* as the symbols *'0'*and *'1'*, and then we can regard an aspect of the wall at time *t + 1* as *'1'*, and so the wall just computed 0 + 1 = 1. Thus, Searle suggests, everything (or rather everything that is big or complex enough to have enough states) is every computer, and the claim that the brain is a computer has no bite.

The problem with this reasoning is that the isomorphism that makes a syntactic engine drive a semantic engine is more full-bodied than Searle acknowledges. In particular, the isomorphism has to include not just a particular computation that the machine *does perform*, but also all the computations that the machine *could have* performed. The point can be made clearer by a look at figure 11.6, a type of X-OR gate. (See O'Rourke and Shattuck, forthcoming.)

The numerals at the beginnings of arrows represent inputs. The computation of 1 + 0 = 1 is represented by the path $A \rightarrow C \rightarrow E$. The computation of 0 + 1 = 1 is represented by the path $A \rightarrow B \rightarrow E$, and so on. Now here is the point. In order for the wall to be this computer, it isn't enough for it to have states that correspond to *'0'* and *'1'* followed by a state that corresponds to *'1'*. It must also be such that *had* the *'1'* input been replaced by a *'0'* input, the *'1'* output *would have been* replaced by the *'0'* output. In

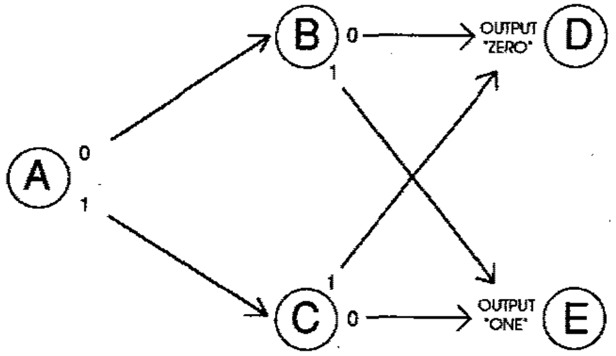


**Figure 11.6**
The numerals at the beginnings of arrows indicate inputs.

[3] The idea described here was first articulated to my knowledge in Fodor (1975, 1980); see also Dennett (1981) to which the terms "semantic engine" and "syntactic engine" are due, and Newell (1980). More on this topic can be found in Dennett (1987) by looking up "syntactic engine" and "semantic engine" in the index.

other words, it has to have symbolic states that satisfy not only the *actual* computation, but also the *possible* computations that the computer *could* have performed. And this is nontrivial.

Searle (1992, 209) acknowledges this point, but insists nonetheless that there is no fact of the matter of whether the brain is a specific computer. Whether something is a computer, he argues, depends on whether we decide to interpret its states in a certain way, and that is up to us. "We can't, on the one hand, say that anything is a digital computer if we can assign a syntax to it, and then suppose there is a factual question intrinsic to its physical operation whether or not a natural system such as the brain is a digital computer." Searle is right that whether something is a computer and what computer it is is *in part* up to us. But what the example just given shows is that it is not *totally* up to us. A rock, for example, is not an X-OR gate. We have a great deal of freedom as to how to interpret a device, but there are also very important restrictions on this freedom, and that is what makes it a substantive claim that the brain is a computer of a certain sort.

## 11.3 Functionalism and the Language of Thought

Thus far we have (1) considered functional analysis, the computer model of the mind's approach to intelligence, (2) distinguished intelligence from intentionality, and (3) considered the idea of the brain as a syntactic engine. The idea of the brain as a syntactic engine explains how it is that symbol-crunching operations can result in a machine "making sense." But so far we have encountered nothing that could be considered the computer model's account of intentionality. It is time to admit that although the computer model of the mind has a natural and straightforward account of intelligence, there is no account of intentionality that comes along for free.

We will not survey the field here. Instead, let us examine a view that represents a kind of orthodoxy, not in the sense that most researchers believe it, but in the sense that the other views define themselves in large part by their response to it.

The basic tenet of this orthodoxy is that our intentional contents are simply meanings of our internal representations. As mentioned earlier, there is something to be said for regarding the content of thought and language as a single phenomenon, and this is a quite direct way of so doing. There is no commitment in this orthodoxy on the issue of whether our internal language, the language in which we think, is the same or different from the language with which we speak. Further, there is no commitment as to a direction of reduction, that is, as to which is more basic, mental content or meanings of internal symbols.

For concreteness, let us talk in terms of Fodor's (1975) doctrine that the meaning of external language derives from the content of thought, and the content of thought derives from the meaning of elements of the language of thought. (See also Harman 1973.) According to Fodor, believing or hoping that grass grows is a state of being in one or another computational relation to an internal representation that means that grass grows. This doctrine can be summed up in a set of slogans: believing that grass grows is having 'Grass grows.' in the Belief Box, desiring that grass grows is having this sentence (or one that means the same) in the Desire Box, and so on.

Now, if all content and meaning derives from meaning of the elements of the language of thought, we immediately want to know how the mental symbols get their meaning.[4] This is a question that gets wildly different answers from different philosophers, all equally committed to the cognitive-science point of view. We briefly look at two of them. The first point of view, mentioned earlier, takes as a kind of paradigm those cases in which a symbol in the head might be said to covary with states in the world in the way that the number of rings in a tree trunk correlates with the age of the tree. (See Dretske 1981, Stampe 1977, Stalnaker 1984, and Fodor 1987, 1990.) On this view, the meaning of mental symbols is a matter of the correlations between these symbols and the world.

One version of this view (Fodor 1990) says that T is the truth condition of a mental sentence M if and only if: M is in the Belief Box if and only if T, in ideal conditions. That is, what it is for 'Grass is green' to have the truth condition that grass be green is for 'Grass is green' to appear in the Belief Box just in case grass really is green (and conditions are ideal). The idea behind this theory is that there are cognitive mechanisms that are designed to put sentences in the Belief Box when and only when they are true, and if those cognitive mechanisms are working properly and the environment cooperates (no mirages, no Cartesian evil demons), these sentences will appear in the Belief Box when and only when they are true.

One problem with this idea is that even if this theory works for "observation sentences" such as 'This is yellow', it is hard to see how it could work for "theoretical sentences." A person's cognitive mechanisms could be working fine, and the environment could contain no misleading evidence, and still one might not believe that space is Riemannian or that some quarks have charm or that one is in the presence of a magnetic field.

[4] In one respect, the meanings of mental symbols cannot be semantically more basic than meanings of external symbols. The name "Aristotle" has the reference it has because of its causal connection (via generations of speakers) to a man who was called by a name that was an ancestor of our external term 'Aristotle.' And so the term in the language of thought that corresponds to 'Aristotle' will certainly derive its reference from and thus will be semantically less basic than the public-language word.

For theoretical ideas, it is not enough to have one's nose rubbed in the evidence: you also have to have the right theoretical idea. And if the analysis of ideal conditions includes "has the right theoretical idea," that would make the analysis circular because having the right theoretical idea amounts to "comes up with the true theory." And appealing to truth in an analysis of 'truth' is to move in a very small circle. (See Block 1986, 657–660.)

The second approach is known as functionalism (actually, "functional role semantics" in discussions of meaning) in philosophy, and as procedural semantics in cognitive psychology and computer science. Functionalism says that what gives internal symbols (and external symbols too) their meanings is how they function. To maximize the contrast with the view described in the last two paragraphs, it is useful to think of the functionalist approach with respect to a symbol that doesn't (on the face of it) have *any* kind of correlation with states of the world, say the symbol 'and'. Part of what makes 'and' mean what it does is that if we are sure of 'Grass is green and grass grows', we find the inference to 'Grass is green' and also 'Grass grows' compelling. And we find it compelling "in itself," not because of any other principle (see Peacocke 1993). Or if we are sure that one of the conjuncts is false, we find compelling the inference that the conjunction is false too. What it is to mean *and* by 'and' is to find such inferences compelling in this way, and so we can think of the meaning of 'and' as a matter of its behavior in these and other inferences. The functionalist view of meaning applies this idea to all words. The picture is that the internal representations in our heads have a function in our deciding, deliberating, problem solving—indeed in our thought in general—and that is what constitutes their meanings.

This picture can be bolstered by considering what happens when one first learns Newtonian mechanics. In my own case, I heard a large number of unfamiliar terms more or less all at once: *'mass'*, *'force'*, *'energy'*, and the like. I never was told definitions of these terms in terms I already knew. (No one has ever come up with definitions of such "theoretical terms" in observation language.) What I did learn was how to *use* these terms in solving homework problems, making observations, explaining the behavior of a pendulum, and the like. In learning how to use the terms in thought and action (and perception as well, though its role there is less obvious), I learned their meanings, and this fits with the functionalist idea that the meaning of a term just *is* its function in perception, thought, and action. (See chapter 4 for a discussion of the restructuring of concepts that goes on in learning a new theory.) A theory of what meaning is can be expected to jibe with a theory of what it is to acquire meanings, and so considerations about acquisition can be relevant to semantics.

An apparent problem arises for such a theory in its application to the meanings of numerals. After all, it is a mathematical fact that truths in the familiar numeral system *'1'*, *'2'*, *'3'* are preserved, even if certain nonstandard interpretations of the numerals are adopted (so long as nonstandard versions of the operations are adopted too). For example, *'1'* might be mapped onto 2, *'2'* onto 4, *'3'* onto 6, and so on. That is, the numerals, both "odd" and "even," might be mapped onto the *even* numbers. Because *'1'* and *'2'* can have the *same* functional role in different number systems and still designate the very numbers they usually designate in normal arithmetic, how can the functional role of *'1'* determine whether *'1'* means 1 or 2? It seems that all functional role can do is "cut down" the number of possible interpretations, and if there are still an infinity left after the cutting down, functional role has gained nothing.

A natural functionalist response would be to emphasize the *input* and *output* ends of the functional roles. We say "two cats" when confronted with a pair of cats, not when confronted with one or five cats, and our thoughts involving the symbol *'3'* affect our actions toward triples in an obvious way in which these thoughts do not affect our actions toward octuples. The functionalist can avoid nonstandard interpretations of *internal* functional roles by including in the semantically relevant functional roles external relations involving perception and action (Harman 1973). In this way, the functionalist can incorporate the insight of the view mentioned earlier that meaning has something to do with covariation between symbols and the world.

The picture of how cognitive science can handle intentionality should be becoming clear. Transducers at the periphery and internal primitive processors produce and operate on symbols so as to give them their functional roles. In virtue of their functional roles (both internal and external), these symbols have meanings. The functional role perspective explains the mysterious correlation between the symbols and their meanings. It is the activities of the symbols that give them their meanings, and so it is no mystery that a syntax-based system should have rational relations among the meanings of the system's symbols. Intentional states have their relations by virtue of these symbolic activities, and the contents of the intentional states of the system, thinking, wanting, and so on, are inherited from the meanings of the symbols. This is the orthodox account of intentionality for the computer model of the mind. It combines functionalism with commitment to a language of thought. Both views are controversial, the latter both in regard to its truth and its relevance to intentionality even if true. Notice, incidentally, that in this account of intentionality, the source of intentionality is computational structure, independently of whether the computational structure is produced by software or hardware. Thus the title of this chapter, in indicating that the mind is the software of the brain, has the potential to mislead. If we think of the computational structure of a computer as coming entirely from a program put into a structureless general-purpose machine, we are very far from the facts about the human brain—which is not such a general-purpose machine.

At the end of this chapter we discuss Searle's famous Chinese Room argument, which is a direct attack on this theory. The next two sections are devoted to arguments for and against the language of thought.

### 11.3.1 Objections to the Language-of-Thought Theory

Many objections have been raised to the language-of-thought picture. Let us briefly look at three objections made by Dennett (1975).

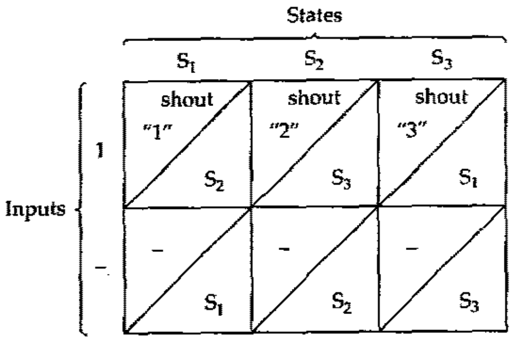
The first objection is that we all have an infinity of beliefs (or at any rate a very large number of them). For example, we believe that trees do not light up like fireflies, and that this book is probably closer to your eyes than the President's left shoe is to the ceiling of the Museum of Modern Art gift shop. But how can it be that so many beliefs are all stored in the rather small Belief Box in your head? One line of response to this objection involves distinguishing between the *ordinary* concept of belief and a *scientific* concept of belief toward which one hopes cognitive science is progressing. For scientific purposes, we home in on cases in which our beliefs *cause* us to *do* something, say throw a ball or change our mind, and cases in which beliefs are caused by something, as when perception of a rhinoceros causes us to believe that there is a rhinoceros in the vicinity. Science is concerned with causation and causal explanation, and so the protoscientific concept of belief is the concept of a *causally active* belief. It is only for these beliefs that the language-of-thought theory is committed to sentences in the head. This idea yields a very simple answer to the infinity objection, namely that on the protoscientific concept of belief, most of us did not *have* the belief that trees do not light up like fireflies until they read this paragraph.

Beliefs in the protoscientific sense are explicit, that is, recorded in storage in the brain. For example, you no doubt were once told that the sun is 93 million miles away from the earth. If so, perhaps you have this fact explicitly recorded in your head, available for causal action, even though until reading this paragraph, this belief hadn't been conscious for years. Such explicit beliefs have the potential for causal interaction, and thus must be distinguished from cases of belief in the ordinary sense (if they are beliefs at all), such as the belief all normal people have that trees do not light up like fireflies.

Being explicit is to be distinguished from other properties of mental states, such as being conscious. Theories in cognitive science tell us of mental representations about which no one knows from introspection, such as mental representations of aspects of grammar. If this is right, there is much in the way of mental representation that is explicit but not conscious, and thus the door is opened to the possibility of belief that is explicit but not conscious.

It is important that the language-of-thought theory is not meant to be a theory of all possible believers, but rather only of *us*. The language-of-thought theory allows creatures who can believe without explicit representation at all, but the claim of the language-of-thought theory is that they aren't us. A digital computer consists of a central processing unit (CPU) that reads and writes explicit strings of zeroes and ones in storage registers. One can think of this memory as in principle unlimited, but of course any actual machine has a finite memory. Now any computer with a finite amount of explicit storage can be simulated by a machine with a much larger CPU and *no* explicit storage, that is, no registers and no tape. The way the simulation works is by using the extra states as a form of implicit memory. So, in principle, we could be simulated by a machine with no explicit memory at all.

Consider, for example, the finite automaton diagrammed in figure 11.7. The table shows it as having three states. The states, $S_1$, $S_2$, and $S_3$, are listed across the top. The inputs are listed at the left side. Each box is in a column and a row that specifies what the machine does when it is in the state named at the top of the column, and when the input is the one listed at the side of the row. The top part of the box names the output, and the bottom part of the box names the next state. This is what the table says: when the machine is in $S_1$, and it sees a '*1*', it says "1", and goes to $S_2$. When it is in $S_2$, if it sees a '*1*' it says "2" and goes into the next state, $S_3$. In that state, if it sees a '*1*' it says "3" and goes back to $S_1$. When it sees nothing, it says nothing and stays in the same state. This automaton counts "modulo" three, that is, you can tell from what it says how many



| | | States | | |
|---|---|---|---|---|
| | | $S_1$ | $S_2$ | $S_3$ |
| Inputs | 1 | shout "1"<br>$S_2$ | shout "2"<br>$S_3$ | shout "3"<br>$S_1$ |
| | – | –<br>$S_1$ | –<br>$S_2$ | –<br>$S_3$ |

**Figure 11.7**

ones it has seen since the last multiple of three. But what the machine table makes clear is that this machine need have no memory of the sort that involves writing anything down. It can "remember" solely by changing state. Some theories based on neural-network models assume that we are such machines.

Suppose, then, that we are digital computers with explicit representations. We could be simulated by finite automata that have many more states and no explicit representations. The simulators will have just the same beliefs as we do, but no explicit representations (unless the simulators are just jukeboxes of the type of the Aunt Bubbles machine described in section 11.1.1). The machine in which remembered items are recorded explicitly has an advantage over a computationally equivalent machine that "remembers" by changing state, namely that the explicit representations can be part of a combinatorial system. This point is explained in the next section.

Time to sum up. The objection was that an infinity of beliefs cannot be written down in the head. My response was to distinguish between a loose and ordinary sense of 'belief' in which it may be true that we have an infinity of beliefs, and a protoscientific sense of 'belief' in which the concept of belief is the concept of a causally active belief. In the latter sense, I claimed, we do not have an infinity of beliefs.

Even if you agree with this response to the infinity objection, you may still feel dissatisfied with the idea that, because the topic has never crossed their minds, most people don't believe that zebras don't wear underwear in the wild. Perhaps it will help to say something about the relation between the protoscientific concept of belief and the ordinary concept. It is natural to want some sort of reconstruction of the ordinary concept in scientific terms, a reconstruction of the sort we have when we define the ordinary concept of the weight of a person as the force exerted on the person by the earth at the earth's surface. To scratch this itch, we can give a first approximation to a definition of a belief in the ordinary sense as anything that is either (1) a belief in the protoscientific sense, or (2) naturally and easily follows from a protoscientific belief.

A second objection to the language-of-thought theory is provided by Dennett's example of a chess-playing program that "thinks" it should get its queen out early, even though there is no explicitly represented rule that says anything like "Get your queen out early." The fact that it gets its queen out early is an "emergent" consequence of an interaction of a large number of rules that govern the details of play. But now consider a human analog of the chess-playing machine. Shouldn't we say that she believes she should get her queen out early despite her lack of any such explicit representation?

The reply to this challenge to the language-of-thought theory is that in the protoscientific sense of belief, the chess player simply does not believe that she should get her queen out early. If this idea seems difficult to accept, notice that there is no additional predictive or explanatory force to the hypothesis that she believes she should get her queen out early beyond the predictive or explanatory force of the explicitly represented strategies from which getting the queen out early emerges. (Though there is no additional predictive force, there may be some additional predictive utility, just as there is utility in navigation to supposing that the sun goes around the earth.) Indeed, the idea that she should get her queen out early can actually conflict with her deeply held chess principles, despite being an emergent property of her usual tactics. We could suppose that if you point out to her that her strategies have the consequence of getting her queen out early, she says, "Oh no, I'd better revise my usual strategies." Thus, postulating that she believes that she should get her queen out early could lead to mistaken predictions of her behavior. In sum, the protoscientific concept of a causally active belief can be restricted to the strategies that really are explicitly represented.

Perhaps there is a quasi-behaviorist ordinary sense of belief in which it is correct to ascribe the belief that the queen should come out early simply because she behaves as if she believes it. Even if we agree to recognize such a belief, it is not one that ever causally affects any other mental states or any behavior, and so it is of little import from a scientific standpoint.

A third objection to the language-of-thought theory is provided by the "opposite" of the "queen out early" case—Dennett's sister in Cleveland case. Suppose that a neurosurgeon operates on someone's Belief Box, inserting the sentence, "I have a sister in Cleveland." When the patient wakes up, the doctor says "Do you have a sister?" "Yes," the patient says, "In Cleveland." Doctor: "What's her name?" Patient: "Gosh, I can't think of it." Doctor: "Older or younger?" Patient: "I don't know, and by golly I'm an only child. I don't know why I'm saying that I have a sister at all." Finally, the patient concludes that she never really believed she had a sister in Cleveland, but rather was a victim of some sort of compulsion to speak as if she did. The upshot is supposed to be that the language-of-thought theory is false because you can't produce a belief just by inserting a sentence in the Belief Box.

The objection reveals a misleading aspect of the "Belief Box" slogan, not a problem with the doctrine that the slogan characterizes. According to the language-of-thought theory, believing that one has a sister in Cleveland is a computational relation to a sentence, but this computational relation shouldn't be thought of as simply *storage*. Rather, the computational relation must include some specification of relations to other sentences to which one also has the same computational relation, and in that sense the computational relation must be holistic. This point holds both for the ordinary notion of belief and the protoscientific notion. It holds for the ordinary notion of belief because we don't count someone as believing just because she mouths words as our neurosurgery victim mouthed the words, "I have a sister in Cleveland." And it holds for the protoscientific notion of belief because the unit of explanation and prediction is much more likely to be groups of coherently related sentences in the brain than single sentences all by themselves. If one is going to retain the "Belief Box" way of talking, one should say that for a sentence in the Belief Box to count as a belief, it should cohere sufficiently with other sentences so as not to be totally unstable, disappearing on exposure to the light.

## 11.4 Arguments for the Language of Thought

So it seems that the language-of-thought hypothesis can be defended from these a priori objections. But is there any positive reason to believe it? One such reason is that it is part of a reasonably successful research program. But there are challengers (mainly, some versions of the connectionist program mentioned earlier), and so a stronger case will be called for if the challengers' research programs also end up being successful.[5]

A major rationale for accepting the language of thought has been one form or another of *productivity* argument, stemming from Chomsky's work (see Chomsky 1975). The idea is that people are capable of thinking vast numbers of thoughts that they have not thought before—and indeed that no one may ever have thought before. Consider, for example, the thought mentioned earlier that this book is closer to you than the President's shoe is to the museum gift-shop ceiling. The most obvious explanation of how we can think such new thoughts is the same as the explanation of how we can frame the sentences that express them: namely, via a combinatorial system in which we think. Indeed, abstracting away from limitations on memory, motivation, and length of life, there may be no upper bound on the number of thinkable thoughts. The number of sentences in the English language is certainly infinite (see volume 1). But what does it mean to say that sentences containing millions of words are "in principle" thinkable?

[5] Notice that the type of success is important to whether connectionism is really a rival to the language-of-thought point of view. Connectionist networks have been successful in various pattern-recognition tasks, for example discriminating mines from rocks. Of course, even if these networks could be made to do pattern-recognition tasks much better than we can, that wouldn't suggest that these networks can provide models of higher cognition. Computers that are programmed to do arithmetic in the classic symbol-crunching mode can do arithmetic much better than we can, but no one would conclude that therefore these computers provide models of higher cognition.

Those who favor productivity arguments say this: The explanation for the fact that we cannot actually think sentences containing millions of words would have to appeal to such facts as that, were we to try to think sufficiently long or complicated thoughts, our attention would flag, or our memory would fail us, or we would die. They think that we can idealize away from these limitations, because the mechanisms of thought themselves are unlimited. But this claim that if we abstract away from memory, mortality, motivation, and the like, our thought mechanisms are unlimited, is a doctrine for which there is no *direct* evidence. The perspective from which this doctrine springs has been fertile, but it is an open question what aspect of the doctrine is responsible for its success.

After all, we might be finite beings, essentially. Not all idealizations are equally correct, and contrary to widespread assumption in cognitive science, the idealization to the unboundedness of thought may be a bad one. Consider a finite automaton naturally described by the table in figure 11.7.[6] Its only form of memory is change of state. If you want to get this machine to count to 4 instead of just to 3, you can't just add more memory, you have to give it another state by changing the way the machine is built. Perhaps we are like this machine.

An extension of the productivity argument to deal with this sort of problem has recently been proposed by Fodor (1987), and Fodor and Pylyshyn (1988). Fodor and Pylyshyn point out that it is a fact about human beings that, if someone can think the thought that Mary loves John, then she can also think the thought that John loves Mary. And likewise for a vast variety of pairs of thoughts that involve the same conceptual constituents, but are put together differently. There is a *systematicity* relation among many thoughts that begs for an explanation in terms of a combinatorial system. The conclusion is that human thought operates in a medium of "movable type."

However, the most obvious candidate for the elements of such a combinatorial system in many areas are the *external* symbol systems themselves. Perhaps the most obvious case is arithmetical thoughts. If someone is capable of thinking the thought that $7 + 16$ is not 20, then presumably she is capable of thinking the thought that $17 + 6$ is not 20. Indeed, someone who has mastered the ten numerals plus other basic symbols of Arabic notation and their rules of combination can think any arithmetical thought that is expressible in a representation that he or she can read. (Notice that false propositions can be thinkable—one can think the thought that $2 + 2 = 5$, if only to think that it is false.)

[6] This table could be used to describe a machine that does have a memory with explicit representation. I say "naturally described" to indicate that I am thinking of a machine that does not have such a memory, a machine for which the table in figure 11.7 is an apt and natural description.

One line of a common printed page contains eighty symbols. There are a great many different arithmetical propositions that can be written on such a line—about as many as there are elementary particles in the universe. Though almost all of them are false, all are arguably thinkable with some work. Starting a bit smaller, try to entertain the thought that 695,302,222,387,987 + 695,302,222,387,986 = 2. How is it that we have so many possible arithmetical thoughts? The obvious explanation is that we can string together—either in our heads or on paper—the symbols (numerals, pluses, and so on) themselves, and simply read the thought off the string of symbols. Of course, this does not show that the systematicity argument is *wrong*. Far from it, for it shows *why* it is right. But this point does *threaten the value* of the systematicity argument considerably. For it highlights the possibility that the systematicity argument may apply only to *conscious* thought, and not to the rest of the iceberg of unconscious thought processes that cognitive science is mainly about. Thus Fodor and Pylyshyn are right that the systematicity argument shows that there is a language of thought. And they are right that if connectionism is incompatible with a language of thought, so much the worse for connectionism. But where they are wrong is with respect to an unstated assumption: that the systematicity argument shows that languagelike representations pervade cognition.

To see this point, notice that much of the success in cognitive science has been in our understanding of perceptual and motor modules (see volume 2). The operation of these modules is neither introspectible—accessible to conscious thought—nor directly influencible by conscious thought. These modules are "informationally encapsulated" (see Pylyshyn 1984, and Fodor 1983). The productivity in conscious thought that is exploited by the systematicity argument certainly does not demonstrate productivity in the processing inside such modules. True, if someone can think that if John loves Mary, then he can think that Mary loves John. But we don't have easy access to such facts about pairs of representations of the kind involved in unconscious processes. Distinguish between the conclusion of an argument and the argument itself. The conclusion of the systematicity argument may well be right about unconscious representations. That is, *systematicity itself* may well obtain in these systems. My point is that the systematicity *argument* shows little about encapsulated modules and other unconscious systems.

The weakness of the systematicity argument is that, resting as it does on facts that are so readily available to conscious thought, its application to unconscious processes is more tenuous. Nonetheless, as the reader can easily see by looking at the other chapters in these volumes, the symbol-manipulation model has been quite successful in explaining aspects of perception thought and motor control. So, although the systematicity argument is limited in its application to unconscious processes, the model it supports for conscious processes appears to have considerable application to unconscious processes nonetheless.

To avoid misunderstanding, I add that the point just made does not challenge all the thrust of the Fodor and Pylyshyn critique of connectionism. Any neural-network model of the mind will have to accommodate the fact of our use of a systematic combinatorial symbol system in conscious thought. It is hard to see how a neural-network model could do this without being in part an implementation of a standard symbol-crunching model.

In effect, Fodor and Pylyshyn (1988, 44) counter the idea that the systematicity argument depends entirely on conscious symbol manipulating by saying that the systematicity argument applies to animals. For example, they argue that the conditioning literature has no cases of animals that *can* be trained to pick the red thing rather than the green one, but *cannot* be trained to pick the green thing rather than the red one.

This reply has some force, but it is uncomfortably anecdotal. The data a scientist collects depend on his or her theory. We cannot rely on data collected in animal conditioning experiments run by behaviorists—who, after all, were notoriously opposed to theorizing about internal states.

Another objection to the systematicity argument derives from the distinction between linguistic and pictorial representation that plays a role in the controversies over mental imagery. (See chapter 7 in volume 2.) Many researchers think that we have two different representational systems, a languagelike system—thinking in words—and a pictorial system—thinking in pictures. If an animal that can be trained to pick red instead of green can also be trained to pick green instead of red, that may reflect the properties of an imagery system shared by human beings and animals, not a properly languagelike system. Suppose Fodor and Pylyshyn are right about the systematicity of thought in animals. That may reflect only a combinatorial pictorial system. If so, it would suggest (though it wouldn't show) that human beings have a combinatorial pictorial system too. But the question would still be open whether humans have a *languagelike* combinatorial system that is used in unconscious thought. In sum, the systematicity argument certainly applies to conscious thought, and it is part of a perspective on unconscious thought that has been fertile, but there are difficulties in its application to unconscious thought.

## 11.5 Explanatory Levels and the Syntactic Theory of the Mind

In this section, let us assume that the language-of-thought hypothesis is correct in order to ask another question: Should cognitive-science explanations appeal only to the syntactic elements in the language of thought (the

*'0'*s and *'1'*s and the like), or should they also appeal to the content of these symbols? Stich (1983) has argued for the "syntactic theory of mind," a version of the computer model in which the language of thought is construed in terms of uninterpreted symbols, symbols that may *have* contents, but whose contents are irrelevant for the purposes of cognitive science. I shall put the issue in terms of a critique of a simplified version of the Stich (1983) argument.

Let us begin with Stich's case of Mrs. T, a senile old lady who answers, "What happened to McKinley?" with "McKinley was assassinated," but cannot answer questions like, "Where is McKinley now?" "Is he alive or dead?" and the like. Mrs. T's logical facilities are fine, but she has lost most of her memories, and virtually all the concepts that are normally connected to the concept of assassination, such as the concept of death. Stich sketches the case so as to persuade us that though Mrs. T may know that something happened to McKinley, she doesn't have any real grasp of the concept of assassination, and thus cannot be said to believe that McKinley was assassinated.

The argument that I will critique concludes that purely syntactic explanations undermine content explanations because a syntactic account is superior to a content account. The syntactic approach is said to be superior in two respects. First, the syntactic account can handle Mrs. T, who has little in the way of intentional content, but plenty of internal representations whose interactions can be used to explain and predict what she does, just as the interactions of symbol structures in a computer can be used to explain and predict what it does. And the same holds for very young children, people with weird psychiatric disorders, and denizens of exotic cultures. In all these cases, cognitive science can (at least potentially) assign internal syntactic descriptions and use them to predict and explain, but there are problems with content ascriptions (though, in the last case at least, the problem is not that these people have no contents, but just that their contents are so different from ours that we cannot assign contents to them in *our terms*). In sum, the first type of superiority of the syntactic perspective over the content perspective is that it allows for the psychology of the senile, the very young, the disordered, and the exotic, and thus, it is alleged, the syntactic perspective is far more *general* than the content perspective.

The second respect of superiority of the syntactic perspective is that it allows more *fine-grained* predictions and explanations than the content perspective. To take a humdrum example, the content perspective allows us to predict that if someone believes that all men are mortal, and that he is a man, he can conclude that he is mortal. But suppose that the way in which this person represents the generalization that all men are mortal to himself is via a syntactic form of the type "All nonmortals are nonmen"; then the inference will be harder to draw than if he had represented it without the negations. In general, what inferences are hard rather than easy, and what sorts of mistakes are likely will be better predictable from the syntactic perspective than from the content perspective, in which all the ways of representing one belief are lumped together.

The upshot of this argument is supposed to be that because the syntactic approach is more general and more fine-grained than the content approach, content explanations are therefore undermined and shown to be defective. And so cognitive science would do well to scrap attempts to explain and predict in terms of content in favor of appeals to syntactic form alone.

But there is a fatal flaw in this argument, one that applies to many reductionist arguments. The fact that syntactic explanations are better than content explanations in some respects says nothing about whether content explanations are not *also* better than syntactic explanations in some respects. A dramatic way of revealing this fact is to notice that if the argument against the content level were correct, *it would undermine the syntactic approach itself*. This point is so simple, fundamental, and widely applicable that it deserves a name: let's call it the Reductionist Cruncher. Just as the syntactic objects on paper can be described in molecular terms, for example as structures of carbon molecules, so too the syntactic objects in our heads can be described in terms of the viewpoint of chemistry and physics. But a physicochemical account of the syntactic objects in our head will be more general than the syntactic account in just the same way as the syntactic account is more general than the content account. There are possible beings, such as Mrs. T, who are similar to us syntactically but not in intentional contents. Similarly, there are possible beings who are similar to us in physicochemical respects, but not syntactically. For example, creatures could be like us in physicochemical respects without having physicochemical parts that function as syntactic objects—just as Mrs. T's syntactic objects don't function so as to confer content upon them. If neural-network models of the sort that antilanguage-of-thought theorists favor could be bioengineered, they would fit this description. The bioengineered models would be like us and like Mrs. T in physicochemical respects, but unlike us and unlike Mrs. T in syntactic respects. Further, the physicochemical account will be more fine-grained than the syntactic account, just as the syntactic account is more fine-grained than the content account. Syntactic generalizations will fail under some physicochemically specifiable circumstances, just as content generalizations fail under some syntactically specifiable circumstances. I mentioned that content generalizations might be compromised if the syntactic realizations included too many syntactic negations. The present point is that syntactic generalizations might fail when syntactic objects interact on the basis of certain physicochemical properties. To take a slightly silly example, if a token of '*s*' and a token of '*s* → *t*' are both positively charged so that they repel each other, that could prevent logic processors from putting them together to yield a token of '*t*'.

In sum, if we could refute the content approach by showing that the syntactic approach is more general and fine-grained than the content approach, then we could also refute the syntactic approach by exhibiting the same deficiency in it relative to a still deeper theory. The Reductionist Cruncher applies even within physics itself. For example, anyone who rejects the explanations of thermodynamics in favor of the explanations of statistical mechanics will be frustrated by the fact that the explanations of statistical mechanics can themselves be "undermined" in just the same way by quantum mechanics.

The same points can be made in terms of the explanation of how a computer works. Compare two explanations of the behavior of the computer on my desk, one in terms of the programming language, and the other in terms of what is happening in the computer's circuits. The latter level is certainly more general in that it applies not only to programmed computers, but also to nonprogrammable computers that are electronically similar to mine—for example, certain calculators. Thus the greater generality of the circuit level is like the greater generality of the syntactic perspective. Further, the circuit level is more fine-grained in that it allows us to predict and explain computer failures that have nothing to do with program glitches. Circuits will fail under certain circumstances (for example, overload, excessive heat or humidity) that are not characterizable in the vocabulary of the program level. Thus the greater predictive and explanatory power of the circuit level is like the greater power of the syntactic level to distinguish cases of the same content represented in different syntactic forms that make a difference in processing.

However, the computer analogy reveals a flaw in the argument that the "upper" level (the program level in this example) explanations are defective and should be scrapped. The fact that a "lower" level like the circuit level is superior in some respects does not show that "higher" levels such as the program levels are not themselves superior in other respects. Thus the upper levels are not shown to be *dispensable*. The program level has *its own* type of greater generality—namely, it applies to computers that use the same programming language, but are built in different ways, even computers that don't have circuits at all (but, say, work via gears and pulleys). Indeed, there are many predictions and explanations that are simple at the program level, but would be absurdly complicated at the circuit level. Further (and here is the Reductionist Cruncher again), if the program level could be shown to be defective by the circuit level, then the circuit level could itself be shown to be defective by a deeper theory, such as the quantum field theory of circuits.

The point here is *not* that the program level is a convenient fiction. On the contrary, the program level is just as *real* and *explanatory* as the circuit level.

Perhaps it will be useful to see the matter in terms of an example from Putnam (1975). Consider a rigid, round peg 1 inch in diameter and a square hole in a rigid board with a 1-inch diagonal. The peg won't fit through the hole for reasons that are easy to understand via a little geometry. (The side of the hole is 1 divided by the square root of 2, which is a number substantially less than 1.) Now if we went to the level of description of this apparatus in terms of the molecular structure that makes up a specific solid board, we could explain the rigidity of the materials, and we would have a more fine-grained understanding, including the ability to predict the incredible case where the alignment and motion of the molecules is such as to allow the peg to actually go through the board. But the "upper"-level account in terms of rigidity and geometry nonetheless provides correct explanations and predictions, and applies more generally to *any* rigid peg and board, even one with quite a different sort of molecular constitution, say one made of glass—a supercooled liquid—rather than a solid.

It is tempting to say that the account in terms of rigidity and geometry is only an approximation, the molecular account being the really correct one. (See Smolensky 1988 for a dramatic case of yielding to this sort of temptation.) But the cure for this temptation is the Reductionist Cruncher: the reductionist will also have to say that an elementary-particle account shows the molecular account to be only an approximation. And the elementary-particle account itself will be undermined by a still deeper theory. The point of a scientific account is to cut nature at its joints, and nature *has real joints* at many levels, each of which requires its own kind of idealization.

Further, those which are counted as elementary particles today may be found to be composed of still more elementary particles tomorrow, and so on, ad infinitum. Indeed, contemporary physics allows this possibility of an infinite series of particles within particles. (See Dehmelt 1989.) If such an infinite series obtains, the reductionist would be committed to saying that there are no genuine explanations because for any explanation at any given level, there is always a deeper explanation that is more general and more fine-grained that undermines it. But the existence of genuine explanations surely does not depend on this recondite issue in particle physics!

I have been talking as if there is just one content level, but actually there are many. Marr distinguished among three levels: the computational level, the level of representation and algorithm, and the level of implementation. At the computational or formal level, the multiplier discussed earlier is to be understood as a function from pairs of numbers to their products, for example, from {7,9} to 63. The most abstract characterization at the level of representation and algorithm is simply the algorithm of the multiplier, namely: multiply *n* by *m* by adding *m* *n* times. A less abstract characterization at this middle level is the program described earlier, a sequence of operations including subtracting 1 from the register that initially represents *n* until it is reduced to zero, adding *m* to the answer register each time. (See figure 11.2.) *Each of these levels is a content level rather than a syntactic level.* There are many types of multipliers whose behavior can be explained (albeit at a somewhat superficial level) simply by referring to the fact that they are multipliers. The algorithm mentioned gives a deeper explanation, and the program—one of many programs that can realize that algorithm—gives a still deeper explanation. However, when we break the multiplier down into parts such as the adder of figures 11.3a and 11.3b, we explain its internal operation in terms of gates that operate on syntax, that is, in terms of operations on numerals. Now it is crucial to realize that the mere possibility of a *description* of a system in a certain vocabulary does not by itself demonstrate the existence of a genuine explanatory level. We are concerned here with cutting nature at its joints, and *talking* as if there is a joint does not make it so. The fact that it is good *methodology* to look first for the function, then for the algorithm, then for the implementation, does not by itself show that these inquiries are inquiries at different levels, as opposed to different ways of approaching the same level. The crucial issue is whether the different vocabularies correspond to genuinely distinct laws and explanations, and in any given case, this question will be answerable only empirically. However, we already have good empirical evidence for the reality of the content levels just mentioned—as well as the syntactic level. The evidence is to be found in this very book, where we see genuine and distinct explanations at the level of function, algorithm, and syntax.

A further point about explanatory levels is that it is legitimate to use different and even *incompatible* idealizations at different levels. See Putnam (1975). It has been argued that because the brain is analog, the digital computer must be incorrect as a model of the mind. But even digital computers are analog at one level of description. For example, gates of the sort described earlier in which 4 volts realizes '*1*' and 7 volts realizes '*0*' are understood from the digital perspective as always representing either '*0*' or '*1*'. But an examination at the electronic level shows that values intermediate between 4 and 7 volts appear momentarily when a register switches between them. We abstract from these intermediate values for the purposes of one level of description, but not another.

## 11.6 Searle's Chinese Room Argument

As we have seen, the idea that a certain type of symbol processing can be what *makes* something an intentional system is fundamental to the computer model of the mind. Let us now turn to a flamboyant frontal attack on this idea by John Searle (1980, 1990b; Churchland and Churchland 1990; the basic idea of this argument stems from Block 1978). Searle's strategy is one of avoiding quibbles about specific programs by imagining that cognitive science in the distant future can come up with the program of an actual person who speaks and understands Chinese, and that this program can be implemented in a machine. Unlike many critics of the computer model, Searle is willing to grant that perhaps this can be done so as to focus on his claim that *even if this can be done, the machine will not have intentional states.*

The argument is based on a thought experiment. Imagine yourself given a job in which you work in a room (the Chinese Room). You understand only English. Slips of paper with Chinese writing on them are put under the input door, and your job is to write sensible Chinese replies on other slips, and push them out under the output door. How do you do it? You act as the CPU (central processing unit) of a computer, following the computer program mentioned above that describes the symbol processing in an actual Chinese speaker's head. The program is printed in English in a library in the room. This is how you follow the program. Suppose the latest input has certain unintelligible (to you) Chinese squiggles on it. There is a blackboard on a wall of the room with a "state" number written on it; it says '*17*'. (The CPU of a computer is a device with a finite number of states whose activity is determined solely by its current state and input, and because you are acting as the CPU, your output will be determined by your input and your "state." The '*17*' is on the blackboard to tell you what your "state" is.) You take book 17 out of the library, and look up these particular squiggles in it. Book 17 tells you to look at what is written on your scratch pad (the computer's internal memory), and given both the input squiggles and the scratch-pad marks, you are directed to change what is on the scratch pad in a certain way, write certain other squiggles on your output pad, push the paper under the output door; and finally, change the number on the state board to '*193*'. As a result of this activity, speakers of Chinese find that the pieces of paper you slip under the output door are sensible replies to the inputs.

But you know nothing of what is being said in Chinese; you are just following instructions (in English) to look in certain books and write certain marks. According to Searle, because you don't understand any Chinese, the system of which you are the CPU is a mere Chinese simulator, not a real Chinese understander. Of course, Searle (rightly) rejects the Turing test for understanding Chinese. His argument, then, is that because the program of a real Chinese understander is not sufficient for understanding Chinese, no symbol-manipulation theory of Chinese understanding (or any other intentional state) is correct about what *makes* something a

Chinese understander. Thus the conclusion of Searle's argument is that the fundamental idea of thought as symbol processing is wrong even if it allows us to build a machine that can duplicate the symbol processing of a person and thereby duplicate a person's behavior.

The best criticisms of the Chinese Room argument have focused on what Searle—anticipating the challenge—calls the systems reply. (See the responses following Searle 1980, and the comment on Searle in Hofstadter and Dennett 1981.) The systems reply has a positive and a negative component. The negative component is that we cannot reason from "Bill has never sold uranium to North Korea" to "Bill's company has never sold uranium to North Korea." Similarly, we cannot reason from "Bill does not understand Chinese" to "The system of which Bill is a part does not understand Chinese." (See Copeland 1993b.) Hence there is a gap in Searle's argument. The positive component goes further, saying that the whole system—man + program + board + paper + input and output doors—does understand Chinese, even though the man who is acting as the CPU does not. If you open up your own computer, looking for the CPU, you will find that it is just one of the many chips and other components on the mother board. The systems reply reminds us that the CPUs of the thinking computers we hope to have someday will not *themselves* think—rather, they will be *parts* of thinking systems.

Searle's clever reply is to imagine the paraphernalia of the "system" *internalized* as follows. First, instead of having you consult a library, we are to imagine you *memorizing* the whole library. Second, instead of writing notes on scratch pads, you are to memorize what you would have written on the pads, and you are to memorize what the state blackboard would say. Finally, instead of looking at notes put under one door and passing notes under another door, you just use your *own body* to listen to Chinese utterances and produce replies. (This version of the Chinese Room has the additional advantage of generalizability so as to involve the complete behavior of a Chinese-speaking system instead of just a Chinese note exchanger.) But as Searle would emphasize, when you seem to Chinese speakers to be conducting a learned discourse with them in Chinese, all you are aware of doing is thinking about what noises the program tells you to make next, given the noises you hear and what you've written on your mental scratch pad.

I argued above that the CPU is just one of many components. If the whole system understands Chinese, that should not lead us to expect the CPU to understand Chinese. The effect of Searle's internalization move—the "new" Chinese Room—is to attempt to destroy the analogy between looking inside the computer and looking inside the Chinese Room. If one looks inside the computer, one sees many chips in addition to the CPU. But if one looks inside the "new" Chinese Room, all one sees is *you*, for you have memorized the library and internalized the functions of the scratch pad and the blackboard. But the point to keep in mind is that although the non-CPU components are no longer easy to see, they are not gone. Rather, they are internalized. If the program requires the contents of one register to be placed in another register, and if you would have done so in the original Chinese Room by copying from one piece of scratch paper to another, in the new Chinese Room you must copy from one of your mental analogs of a piece of scratch paper to another. You are implementing the system by doing what the CPU would do and you are simultaneously simulating the non-CPU components. Thus if the positive side of the systems reply is correct, the total system that you are implementing does understand Chinese.

"But how can it be," Searle would object, "that you implement a system that understands Chinese even though *you* don't understand Chinese?" The systems-reply rejoinder is that you implement a Chinese understanding system without yourself understanding Chinese or necessarily even being aware of what you are doing under that description. The systems reply sees the Chinese Room (new and old) as an English system implementing a Chinese system. What you are aware of are the thoughts of the English system, for example your following instructions and consulting your internal library. But in virtue of doing this Herculean task, you are also implementing a real, intelligent Chinese-speaking system, and so your body houses two genuinely distinct intelligent systems. The Chinese system also thinks, but though you implement this thought, you are not aware of it.

The systems reply can be backed up with an addition to the thought experiment that highlights the division of labor. Imagine that you take on the Chinese simulating as a 9-to-5 job. You come in Monday morning after a weekend of relaxation, and you are paid to follow the program until 5:00 P.M. When you are working, you concentrate hard on working, and so instead of trying to figure out the meaning of what is said to you, you focus your energies on working out what the program tells you to do in response to each input. As a result, during working hours you respond to everything just as the program dictates, except for occasional glances at your watch. (The glances at your watch fall under the same category as the noises and heat given off by computers: aspects of their behavior that are not part of the machine description but are due rather to features of the implementation.) If someone speaks to you in English, you say what the program (which, you recall, describes a real Chinese speaker) dictates. So if during working hours someone speaks to you in English, you respond with a request in Chinese to speak Chinese, or even an inexpertly pronounced "No speak English," which was once memorized by the Chinese speaker being simulated, and which you the English-speaking system may even fail to recognize as English. Then, come 5:00 P.M., you stop working and react to Chinese talk just as any monolingual English speaker would.

Why is it that the English system implements the Chinese system rather than, say, the other way around? Because you (the English system whom I am now addressing) are following the instructions of a program in English to make Chinese noises and not the other way around. If you decide to quit your job to become a magician, the Chinese system disappears. However, if the Chinese system decides to become a magician, he will make plans that he would express in Chinese, but then when 5:00 P.M. rolls around, you quit for the day, and the Chinese system's plans are on the shelf until you come back to work. And of course you have no commitment to doing *whatever* the program dictates. If the program dictates that you make a series of movements that leads you to a flight to China, you can drop out of the simulating mode, saying "I quit!" The Chinese speaker's existence and the fulfillment of his plans depends on your work schedule and your plans, not the other way around.

Thus, you and the Chinese system cohabit one body. In effect, Searle uses the fact that you are not aware of the Chinese system's thoughts as an argument that it has no thoughts. But this is an invalid argument. Real cases of multiple personalities are often cases in which one personality is unaware of the others.

It is instructive to compare Searle's thought experiment with the string-searching Aunt Bubbles machine described at the beginning of this paper. This machine was used against a behaviorist proposal of a behavioral *concept* of intelligence. But the symbol-manipulation view of the mind is not a proposal about our everyday concept. To the extent that we think of the English system as implementing a Chinese system, that will be because we find the symbol-manipulation theory of the mind plausible as an empirical theory.

There is one aspect of Searle's case with which I am sympathetic. I have my doubts as to whether there is anything "it is like" to be the Chinese system, that is, whether the Chinese system is a *phenomenally conscious* system. My doubts arise from the idea that perhaps consciousness is more a matter of implementation of symbol processing than of symbol processing itself. Though surprisingly Searle does not mention this idea in connection with the Chinese Room, it can be seen as the argumentative heart of his position. Searle has argued independently of the Chinese Room (Searle 1992, ch. 7) that intentionality requires consciousness. (See the replies to Searle (1990c) in *Behavioral and Brain Sciences* 13, 1990.) But this doctrine, if correct, can shore up the Chinese Room argument. For if the Chinese system is not conscious, then, according to Searle's doctrine, it is not an intentional system, either.

Even if I am right about the failure of Searle's argument, it does succeed in sharpening our understanding of the nature of intentionality and its relation to computation and representation.

## Suggestions for Further Reading

For book-length treatments of the computer model of the mind, see Pylyshyn (1984) and Copeland (1993a). There are a number of excellent anthologies: Block (1980, 1981b), Haugeland (1981), Lycan (1990), Rosenthal (1991), Beakley and Ludlow (1992), and Goldman (1993). These anthologies include many of the papers discussed here. In addition, I recommend Guttenplan (1994), which has short summaries of the state of play on these and many other issues about the mind.

On internal representation and its semantics, Fodor (1985) provides a good but dated guide. Cummins (1989) and Sterelny (1990) are readable books with chapters on all the major views. Haugeland (1990) looks at the major views from a phenomenological perspective. See also Dreyfus (1979). Putnam (1988) is a critique of the notions of meaning and content implicit in the computer model. See also the articles in the first section of Block (1980), especially Field (1978), which is the classic article on the relation between internal representation and functionalism. This book, and Lycan (1990) as well, have sections on imagery that cover the issue of whether mental images are coded in representations like the ‘*1*’s and ‘*0*’s in computers, or in representations that are "pictorial." For a book-length treatment, see Tye (1991). Stich and Warfield (1994) covers a wide range of issues about mental representation.

The best books to date on philosophical issues surrounding connectionism are Clark (1993) and Ramsey, Stich, and Rumelhart (1991). See also Smolensky (1988).

The discussion here of issues of reduction and levels of description is very one-sided. The best presentation of the other side is Kim (1992). See also Churchland (1986).

Philosophers are increasingly concerned with the reality of content and its role in cognitive science. The place to start is with Churchland (1981) and Stich (1983). Sterelny (1985) is an interesting review of Stich. Dennett (1987) takes a view somewhere between Churchland's eliminativism and the realism espoused here. The case against content from the point of view of issues in the philosophy of language is discussed in detail in Schiffer (1987), a difficult work. Horwich (1990) argues that deflationary views of truth are irrelevant to these issues about content.

Discussions of the Turing test are to be found in Moor (1987) and Block (1981a).

The discussion in this chapter has omitted all mention of consciousness except at the very end. This neglect reflects limitations of space, not ideology. See the entries on "Consciousness" and "Qualia" in Guttenplan (1994) and Davies and Humphreys (1993). See also Block (1995).

## Problem

11.1 Schwartz (1988) argues for connectionism and against standard computational models on the ground that the brain is slow, squishy, and error-prone, whereas computers execute an intricate parallel dance of interlocking reliable processes. The conclusion is that computer models of the mind cannot be right, and that we must seek biological approaches to the mind. Can you find anything wrong with this argument?

## Questions

11.1 Recall the example of the molecule-for-molecule duplicate of your brain that happened by chance to come together from molecules from the swamp. You disapprove of the

Supreme Court, and the swamp-brain, if hooked up to a mouth, mouths all the same anti–Supreme Court slogans as you do. But as I argued earlier, the swamp-brain, never having read about the Supreme Court or heard anyone talk about it or anything of the kind, should not be regarded as having any intentional states that are *about* it. Still, all the swamp-brain's states are the same as yours "from the inside," and so the question naturally arises: "Is there *some* kind of content you share with him (it)?" Philosophers who answer Yes have named the kind of content you (putatively) share with the swamp-brain "narrow" content, and there is a raging debate about whether there is such a thing. The case against it is presented in Burge (1979, 1986), Putnam (1988), and Pettit and McDowell (1986). A defense is to be found in Fodor (1987), ch. 2.

11.2 Our beliefs certainly influence what we do, and it seems that our beliefs do so by virtue of their content. If my belief that the American political system is rotten caused me to speak out, my action was due to the content of my belief. Had I believed our political system was great, I wouldn't have spoken. But how can content be causally efficacious when the primitive processors in our heads are sensitive only to the syntactic properties of representations, and not their semantic properties? The issue is further discussed in Lepore and Loewer (1987), Dretske (1988), and Block (1990b).

11.3 Suppose you had an identical twin raised from birth with color-"inverting" lenses in his eyes. Isn't it possible that things that you both call "green" look to him the way things you both call "red" look to you? If this sort of spectrum inversion is possible, does it show that there can be no computer model of this "qualitative" content? A good case for the possibility of spectrum inversion is provided by Shoemaker (1981). For the opposing view, see Dennett (1988) and Harman (1989); Block (1990a) is a reply to Harman. Comprehensive studies are Lycan (1987) and Dennett (1991).

16.4 Many philosophers have followed Dennett (1969) in adopting an *evolutionary* approach to intentionality. Papineau (1984) and Millikan (1984) have argued that what makes the frog's fly representation represent flies is that this representation fulfills its biological function when flies are present. But if evolution is essential to intentionality, how could a computer, being a nonevolved device, have intentionality?[7]

## References

Beakeley, B., and P. Ludlow, eds. (1992). *Philosophy of mind: Classical problems/contemporary issues.* Cambridge, MA: MIT Press.

Block, N. (1978). Troubles with functionalism. In C. W. Savage, ed., *Minnesota studies in philosophy of science,* IX, 26–325. Minneapolis, MN: University of Minnesota Press. Reprinted in Rosenthal (1991) and Lycan (1990).

Block, N. (1980, 1981b). *Readings in philosophy of psychology,* vols. 1, 2. Cambridge, MA: Harvard University Press.

Block, N. (1981a). Psychologism and behaviorism. *The Philosophical Review* 90, 1, 5–43.

Block, N. (1986). Advertisement for a semantics for psychology. In P. A. French et al., eds., *Midwest studies in philosophy,* vol. X. Minneapolis, MN: University of Minnesota Press, 615–678.

Block, N. (1990a). Inverted earth. In J. Tomberlin, ed., *Philosophical perspectives,* vol. 4, Atascadero: Ridgeview.

Block, N. (1990b). Can the mind change the world? In G. Boolos, ed., *Essays in honor of Hilary Putnam.* Cambridge: Cambridge University Press.

[7] I am indebted to Ken Aizawa, George Boolos, Susan Carey, Willem DeVries, Jerry Fodor, and Steven White for comments on an earlier draft. This work was supported by the National Science Foundation (DIR8812559)

Block, N. (1995). On a confusion about a function of consciousness. *The Behavioral and Brain Sciences* 18, 227–247.

Block, N., O. Flanagan, and G. Guzeldere (1995). *The nature of consciousness.* Cambridge, MA: MIT Press.

Burge, T. (1979). Individualism and the mental. In P. A. French et al., eds., *Midwest studies in philosophy* IV, 73–121. Minneapolis, MN: University of Minnesota Press.

Burge, T. (1986). Individualism and psychology. *The Philosophical Review* 95, 1, 3–45.

Chalmers, D. (1994). On implementing a computation. *Minds and Machines* 4, 4, 391–402.

Chomsky, N. (1959). Review of B. F. Skinner's *Verbal Behavior. Language* 35, 1, 26–58.

Chomsky, N. (1975). *Reflections on language.* New York: Pantheon.

Churchland, P. M. (1981). Eliminative materialism and the propositional attitudes. *The Journal of Philosophy* 78, 67–90.

Churchland, P. M., and P. S. Churchland (1990). Could a machine think? *Scientific American* 262, 1, 26–31.

Churchland, P. S. (1986). *Neurophilosophy.* Cambridge, MA: MIT Press.

Clark, A. (1993). *Associative engines.* Cambridge, MA: MIT Press.

Copeland, J. (1993a). *Artificial intelligence: A philosophical introduction.* Oxford: Blackwell.

Copeland, J. (1993b). The curious case of the Chinese gym. *Synthese* 95, 173–186.

Cummins, R. (1975). Functional analysis. *Journal of Philosophy* 72, 741–765. Partially reprinted in Block (1980).

Cummins, R. (1989). *Meaning and mental representation.* Cambridge, MA: MIT Press.

Davies, M., and G. Humphreys (1993). *Consciousness.* Oxford: Blackwell.

Dehmelt, H. (1989). Triton, ... electron, ... cosmon, ...: An infinite regression? *Proceedings of the National Academy of Sciences* 86, 8618.

Dennett, D. (1969). *Content and consciousness.* London: Routledge and Kegan Paul.

Dennett, D. C. (1974). Why the law of effect will not go away. *Journal of the Theory of Social Behavior* 5, 169–187.

Dennett, D. C. (1975). Brain writing and mind reading. In K. Gunderson, ed., *Minnesota studies in philosophy of science,* vol. VII. Minneapolis, MN: University of Minnesota Press.

Dennett, D. C. (1981). Three kinds of intentional psychology. In R. Healy, ed., *Reduction, time and reality.* Cambridge: Cambridge University Press.

Dennett, D. C. (1987). *The intentional stance.* Cambridge, MA: MIT Press.

Dennett, D. C. (1988). Quining qualia. In A. Marcel and E. Bisiach, eds., *Consciousness in contemporary society.* Oxford: Oxford University Press.

Dennett, D. C. (1991). *Consciousness explained.* Boston: Little, Brown.

Dretske, F. (1981). *Knowledge and the flow of information.* Cambridge, MA: MIT Press.

Dretske, F. (1988). *Explaining behavior: Reasons in a world of causes.* Cambridge, MA: MIT Press.

Dreyfus, H. L. (1979). *What computers can't do.* New York: Harper and Row.

Field, H. (1978). Mental representation. *Erkenntnis* 13, 1, 9–61. Reprinted in Block (1980).

Fodor, J. (1968). The appeal to tacit knowledge in psychological explanation. *The Journal of Philosophy* 65, 20.

Fodor, J. (1975). *The language of thought.* New York: Crowell.

Fodor, J. (1980). Methodological solipsism considered as a research strategy in cognitive psychology. *The Behavioral and Brain Sciences* 3, 417–424. Reprinted in Haugeland (1981).

Fodor, J. (1981). Three cheers for propositional attitudes. In Fodor's *RePresentations.* Cambridge, MA: MIT Press.

Fodor, J. (1983). *The modularity of mind: An essay on faculty psychology.* Cambridge, MA: MIT Press.

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

Fodor, J. (1985). Fodor's guide to mental representation. *Mind* 94, 76–100.

Fodor, J. (1987). *Psychosemantics*. Cambridge, MA: MIT Press.

Fodor, J. (1990). Psychosemantics, or where do truth conditions come from? In Lycan (1990).

Fodor, J., and Z. Pylyshyn (1988). Connectionism and cognitive architecture: A critical analysis. *Cognition* 28, 3–71.

Goldman, A., ed. (1993). *Readings in philosophy and cognitive science*. Cambridge, MA: MIT Press.

Guttenplan, S., ed. (1994). *A companion to philosophy of mind*. Oxford: Blackwell.

Harman, G. (1973). *Thought*. Princeton, NJ: Princeton University Press.

Harman, G. (1990). The intrinsic quality of experience. In J. Tomberlin (1990).

Haugeland, J. (1978). The nature and the plausibility of cognitivism. *The Behavioral and Brain Sciences* 1, 215–226. Reprinted in Haugeland (1981).

Haugeland, J. (1980). Programs, causal powers, and intentionality. *The Behavioral and Brain Sciences* 3, 432–3.

Haugeland, J., ed. (1981). *Mind design*. Cambridge, MA: MIT Press.

Haugeland, J. (1990). The intentionality all-stars. In Tomberlin (1990).

Hofstadter, D., and D. Dennett (1981). *The mind's I: Fantasies and reflections on mind and soul*. New York: Basic Books.

Horwich, P. (1990). *Truth*. Oxford: Blackwell.

Jackson, F. (1993). Block's challenge. In J. Bacon, K. Campbell, and L. Reinhardt. *Ontology, causality and mind*. Cambridge, MA: Cambridge University Press, 235–246.

Kim, J. (1992). Multiple realization and the metaphysics of reduction. *Philosophy and Phenomenological Research* 52, 1.

LePore, E., and B. Loewer (1987). Mind matters. *The Journal of Philosophy* 84, 11, 630–641.

Lycan, W., ed. (1987). *Consciousness*. Cambridge, MA: MIT Press.

Lycan, W. (1990). *Mind and cognition*. Oxford: Blackwell.

Marr, D. (1977). Artificial intelligence: A personal view. *Artificial Intelligence* 9, 37–48. Reprinted in Haugeland (1981).

McCarthy, J. (1980). Beliefs, machines and theories. *The Behavioral and Brain Sciences* 3, 435.

Millikan, R. G. (1984). *Language, thought and other biological categories: New foundations for realism*. Cambridge, MA: MIT Press.

Moor, J. (1987). Turing Test. In S. Shapiro, ed., *Encyclopedia of artificial intelligence*. New York: John Wiley and Sons, 1126–1130.

Newell, A. (1980). Physical symbol systems. *Cognitive Science* 4, 2, 135–183.

O'Rourke, J., and J. Shattuck (1993). Does a rock realize every finite automaton? A critique of Putnam's theorem. TR #030 Smith College.

Papineau, D. (1984). Representation and explanation. *Philosophy of Science* 51, 4, 550–572.

Peacocke, C. (1993). *A study of concepts*. Cambridge, MA: MIT Press.

Pettit, P., and J. McDowell (1986). *Subject, thought, and context*. Oxford: Oxford University Press.

Putnam, H. (1975). Philosophy and our mental life. In *Mind, language and reality: Philosophical papers*, vol. 2. London: Cambridge University Press. Reprinted in Block (1980), and, in somewhat different form, in Haugeland (1981). Originally published in *Cognition* 2 (1973) with a section on IQ that is omitted from both of the reprinted versions.

Putnam, H. (1988). *Representation and reality*. Cambridge, MA: MIT Press.

Pylyshyn, Z. (1984). *Computation and cognition: Issues in the foundations of cognitive science*. Cambridge, MA: MIT Press.

Ramsey, W., S. Stich, and D. Rumelhart (1991). *Philosophy and connectionist theory*. Hillsdale, NJ: Erlbaum.

Rosenthal, D. M., ed. (1991). *The nature of mind*. Oxford: Oxford University Press.

Schiffer, S. (1987). *Remnants of meaning*. Cambridge, MA: MIT Press.

Schwartz, J. (1988). The new connectionism: Developing relationships between neuroscience and artificial intelligence. *Daedalus* 117, 1, 123–142.

Searle, J. (1980). Minds, brains, and programs. *The Behavioral and Brain Sciences* 3, 417–424. Reprinted in Haugeland (1981).

Searle, J. (1990a). Is the brain a digital computer? *Proceedings and Addresses of the American Philosophical Association* 64, 21–37.

Searle, J. (1990b). Is the brain's mind a computer program? *Scientific American* 262, 1, 20–25.

Searle, J. (1990c). Consciousness, explanatory inversion and cognitive science. *The Behavioral and Brain Sciences* 13: 4, 585–595.

Searle, J. (1992). *The rediscovery of the mind*. Cambridge, MA: MIT Press.

Shieber, S. (1994). Lessons from a restricted Turing test. *Communications of the ACM* 37, 6, 70–78.

Shoemaker, S. (1981). The inverted spectrum. *The Journal of Philosophy* 74, 7, 357–381.

Smolensky, P. (1988). On the proper treatment of connectionism. *Behavioral and Brain Sciences* 11, 1–23. See also the commentary that follows and the reply by the author.

Stalnaker, R. (1984). *Inquiry*. Cambridge, MA: MIT Press.

Stampe, D. W. (1977). Toward a causal theory of linguistic representation. In P. A. French et al., eds., *Midwest studies in philosophy* II. Minneapolis, MN: University of Minnesota Press, 42–46.

Sterelny, K. (1985). Review of Stich (1983). *Australasian Journal of Philosophy* 63, 4, 510–519.

Sterelny, K. (1990). *The representational theory of the mind*. Oxford: Blackwell.

Stich, S. (1983). *From folk psychology to cognitive science: The case against belief*. Cambridge, MA: MIT Press.

Stich, S., and T. Warfield (1994). *Mental representation: A reader*. Cambridge, MA: Blackwell.

Tomberlin, J. (1990). *Philosophical perspectives, IV: Philosophy of mind and action theory*. Atascadero, CA: Ridgeview Publishing Co.

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

Tye, M. (1991). *The imagery debate*. Cambridge, MA: MIT Press.

Weizenbaum, J. (1976). *Computer power and human reason*. San Francisco: W. H. Freeman.

White, S. (1982). Functionalism and propositional content. Doctoral dissertation, University of California, Berkeley.